Doctoral Thesis

Knowledge Graph Mining with Latent Shape Graphs

Emir Muñoz

JULY 21, 2020

External Examiner	Supervisor	Internal Examiner
Prof. Dr Fabian Suchanek	Dr Matthias Nickles	Prof. Dr Mathieu d'Aquin
	Co-supervisor	
	Dr Vít Nováček	

DISSERTATION SUBMITTED IN PURSUANCE OF THE DEGREE OF DOCTOR OF PHILOSOPHY

Data Science Institute
National University of Ireland, Galway / Ollscoil na hÉireann, Gaillimh

The research presented in this dissertation has been supported by Fujitsu Laboratories Ltd., Japan and the Data Science Institute at National University of Ireland Galway, Ireland through the projects KI2NA (2014–2016) and TOMOE (2016–2018).

Abstract

Knowledge Graph Mining with Latent Shape Graphs

Knowledge graphs are graph-structured knowledge bases that have shown to be of great value in many Artificial Intelligence applications in academia and industry alike. They are typically generated automatically from un-/semi- structured data sources. The increasing popularity of knowledge graphs has been limited by multiple challenges given the size and quality of the information they contain. This thesis explores the relationship between the quality of knowledge graphs and machine learning technologies used to discover and extract knowledge from them. We focus on quality in terms of completeness and consistency.

Knowledge graphs provide the flexibility required for representing knowledge at different scales in open environments such as the Web. However, their versatility makes them have an ever-changing schema, which also makes them hard to summarize and understand their content. Moreover, they are typically never complete—even in very specific domains—and their consistency with respect to a given schema or ontology cannot be guaranteed without the corresponding validation. That lack of an accurate schema has shown to be problematic in use cases where applications might need to rely on the fact that data satisfy a set of constraints.

The contribution of this thesis is twofold. Firstly, we propose a scalable data-driven method to exhibit the actual (latent) shape of graph data. We introduce an algorithm for mining relation cardinality bounds and building so-called shapes that exhibit important aspects of the structure (or topological information) of entities and relations in a knowledge graph. Latent shapes also allow us to formalise an approximate algorithm for validating the structure of knowledge graphs. Secondly, we exploit the latent shapes of entities and relations to enhance the performance of machine learning models aimed to predict missing links and complete knowledge graphs. We use local patterns information and graph-based feature models in the Bioinformatics domain for improving the prediction of adverse drug reactions achieving new state-of-the-art results. Finally, we extend latent feature models by encoding the cardinality of relations as a regularisation term used to learn semantic embeddings that improve the precision of downstream prediction tasks in benchmark datasets.

Acknowledgements

A self that goes on changing is a self that goes on living
— Virginia Woolf

This journey started a while back with me leaving home to a short adventure in Ireland for three months. That time recently became eight years! I would like to thank my mother Rosa, sisters (Daniela, Yessenia), nieces (Thaira, Brenda, Sophia), and nephews (Diego, Axel) for all their support and love in the distance.

From early in my career as a researcher, I have been privileged to work and collaborate with many people who taught me in their way what is to be a good researcher. First, I would like to thank Dr Flavio Ferrarotti and Dr Aidan Hogan from who I truly learnt a lot about Databases and the Semantic Web. Without knowing it, you really made me see research from different angles and guided me to follow the topic of this PhD thesis. Special thanks to my supervisor Dr Matthias Nickles and co-supervisor Dr Vít Nováček for all the guidance and support during the process of this thesis. The process was not always smooth but Matthias and Vít helped me to stay focused and advised me in the difficult times. I would also like to thank the members of my committee: Prof. Dr Fabian Suchanek and Prof. Dr Mathieu d'Aquin for their valuable feedback and guidance. Their valuable questions, comments and suggestions have helped to improve this document.

A lot of thanks go to the Data Science Institute (previously DERI, INSIGHT, and DSI) and the Fujitsu team (Luca, Pierre-Yves, Piero, Sameh, Pasquale, Anthony, and many others). Too many people to mention, but all of you made this journey very satisfactory. Thank you for the opportunities and for believing in me.

On a personal note, I would like to thank my friends in Chile: Davinia, Eduardo, Joaquin, and Rodrigo for keeping me updated with their lives and adventures. Many thanks to my friends I met in Ireland: Hugo, Daniela, Gabi, Erik, and many others. Last but most importantly, a special thank you goes to my wife Bianca who lifted me all these years and decided to join me in this madness. I love you very much.

Published Work

Parts of this thesis also appear in the following peer-reviewed publications:

International Journals.

- Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (2019). 'Facilitating prediction of adverse drug reactions by using knowledge graphs and multi-label learning models'. In: *Briefings in Bioinformatics* 20.1, pp. 190–202
- Emir Muñoz and Matthias Nickles (2018). 'Statistical Relation Cardinality Bounds in Knowledge Bases'. In: *T. Large-Scale Data- and Knowledge-Centered Systems* 39, pp. 67–97

PhD Symposium Papers.

- Emir Muñoz (2016). 'On Learnability of Constraints from RDF Data'. In: *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings*. Ed. by Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Simone Paolo Ponzetto, and Christoph Lange. Vol. 9678. Lecture Notes in Computer Science. Springer, pp. 834–844

International Conferences.

- Emir Muñoz, Pasquale Minervini, and Matthias Nickles (2019). 'Embedding cardinality constraints in neural link predictors'. In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, Limassol, Cyprus, April 8-12, 2019*. Ed. by Chih-Cheng Hung and George A. Papadopoulos. ACM, pp. 2243–2250
- Emir Muñoz and Matthias Nickles (2017). 'Mining Cardinalities from Knowledge Bases'. In: *Database and Expert Systems Applications - 28th International Conference, DEXA 2017, Lyon, France, August 28-31, 2017, Proceedings, Part I*. ed. by Djamal Benslimane, Ernesto Damiani, William I. Grosky, Abdelkader Hameurlain, Amit P. Sheth, and Roland R. Wagner. Vol. 10438. Lecture Notes in Computer Science. Springer, pp. 447–462
- Sameh K. Mohamed, Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (2017). 'Identifying Equivalent Relation Paths in Knowledge Graphs'. In: *Language, Data, and Knowledge - First International Conference, LDK 2017, Galway, Ireland, June 19-20, 2017, Proceedings*. Ed. by Jorge Gracia, Francis Bond, John P. McCrae, Paul Buitelaar, Christian Chiarcos, and Sebastian Hellmann. Vol. 10318. Lecture Notes in Computer Science. Springer, pp. 299–314

- Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (2016). 'Using Drug Similarities for Discovery of Possible Adverse Reactions'. In: *AMIA 2016, American Medical Informatics Association Annual Symposium, Chicago, IL, USA, November 12-16, 2016*. 2500122[PII]. AMIA, pp. 924–933

International Workshops.

- Sameh K. Mohamed, Vít Nováček, Pierre-Yves Vandenbussche, and Emir Muñoz (2019). 'Loss Functions in Knowledge Graph Embedding Models'. In: *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG2019) Co-located with the 16th Extended Semantic Web Conference 2019 (ESWC 2019), Portoroz, Slovenia, June 2, 2019*. Ed. by Mehwish Alam, Davide Buscaldi, Michael Cochez, Francesco Osborne, Diego Reforgiato Recupero, and Harald Sack. Vol. 2377. CEUR Workshop Proceedings. CEUR-WS.org, pp. 1–10
- Emir Muñoz (2014). 'Learning Content Patterns from Linked Data'. In: *Proceedings of the Second International Workshop on Linked Data for Information Extraction (LD4IE 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, October 20, 2014.* Ed. by Anna Lisa Gentile, Ziqi Zhang, Claudia d'Amato, and Heiko Paulheim. Vol. 1267. CEUR Workshop Proceedings. CEUR-WS.org, pp. 21–32

Other papers published during the course of the PhD but that are not included in this thesis.

- Pasquale Minervini, Luca Costabello, Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (2017). 'Regularizing Knowledge Graph Embeddings via Equivalence and Inversion Axioms'. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part I*. ed. by Michelangelo Ceci, Jaakko Hollmén, Ljupco Todorovski, Celine Vens, and Saso Dzeroski. Vol. 10534. Lecture Notes in Computer Science. Springer, pp. 668–683
- Semih Yumusak, Emir Muñoz, Pasquale Minervini, Erdogan Dogdu, and Halife Kodaz (2016). 'A Hybrid Method for Rating Prediction Using Linked Data Features and Text Reviews'. In: *Joint Proceedings of the 5th Workshop on Data Mining and Knowledge Discovery meets Linked Open Data and the 1st International Workshop on Completing and Debugging the Semantic Web (Know@LOD-2016, CoDeS-2016) co-located with 13th ESWC 2016, Heraklion, Greece, May 30th, 2016.* Ed. by Heiko Paulheim, Jens Lehmann, Vojtech Svátek, Craig A. Knoblock, Matthew Horridge, Patrick Lambrix, and Bijan Parsia. Vol. 1586. CEUR Workshop Proceedings. CEUR-WS.org
- Suad Aldarra and Emir Muñoz (2015). 'A Linked Data-Based Decision Tree Classifier to Review Movies'. In: *Proceedings of the 4th Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data co-located with 12th Extended Semantic Web Conference (ESWC 2015), Portoroz, Slovenia, May 31, 2015.* Ed. by Johanna Völker, Heiko Paulheim, Jens Lehmann, and Vojtech Svátek. Vol. 1365. CEUR Workshop Proceedings. CEUR-WS.org

- Bo Liu, Sebastian Link, and Emir Muñoz (2015). 'Validation of Expressive XML Keys with XML Schema and XQuery'. In: *11th Asia-Pacific Conference on Conceptual Modelling, APCCM 2015, Sydney, Australia, January 2015*. Ed. by Motoshi Saeki and Henning Köhler. Vol. 165. CRPIT. Australian Computer Society, pp. 81–90
- Suad Aldarra, Emir Muñoz, Pierre-Yves Vandenbussche, and Vít Nováček (2014). 'SemanTex: Semantic Text Exploration Using Document Links Implied by Conceptual Networks Extracted from the Texts'. In: *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.* Ed. by Matthew Horridge, Marco Rospocher, and Jacco van Ossenbruggen. Vol. 1272. CEUR Workshop Proceedings. CEUR-WS.org, pp. 345–348
- Emir Muñoz, Luca Costabello, and Pierre-Yves Vandenbussche (2014). 'μRaptor: A DOM-based System with Appetite for hCard Elements'. In: *Proceedings of the Second International Workshop on Linked Data for Information Extraction (LD4IE 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, October 20, 2014.* Ed. by Anna Lisa Gentile, Ziqi Zhang, Claudia d'Amato, and Heiko Paulheim. Vol. 1267. CEUR Workshop Proceedings. CEUR-WS.org, pp. 67–71
- Emir Muñoz, Aidan Hogan, and Alessandra Mileo (2014). 'Using linked data to mine RDF from wikipedia's tables'. In: *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*. Ed. by Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler. ACM, pp. 533–542

Patent applications submitted over the course of the PhD (not necessarily related).

- Pasquale Minervini, Luca Costabello, Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (May 2018). *Method and apparatus for completing a knowledge graph*. US Patent App. 15/821,088
- Emir Muñoz, Suad Aldarra, and Luca Costabello (Jan. 2017). *A method, computer program and filter for protecting a computer device against malicious RDF content in Linked Data*. EPO Patent App. EP20170153795
- Emir Muñoz, Ahmed Abdelrahman, Vít Nováček, and Pierre-Yves Vandenbussche (Nov. 2016). *Apparatus and a system for calculating similarities between drugs and using the similarities to extrapolate side effects*. US Patent App. 15/087,902
- Roger Menday, Luca Costabello, Jürgen Umbrich, Pierre-Yves Vandenbussche, Emir Muñoz, and Vít Nováček (Oct. 2016). *Query mediator, a method of querying a polyglot data tier and a computer program executable to carry out a method of querying a polyglot data tier*. US Patent App. 15/084,293

Contents

Declaration

I declare that this thesis, titled "*Knowledge Graph Mining with Latent Shape Graphs*", is composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

July 21, 2020
Galway, Ireland

EMIR MUÑOZ

List of Figures

List of Tables

CHAPTER 1

Introduction

Contents

1.1 Motivation

THE functioning of the human brain is still largely mysterious for scientists. Our brain allows us to store (an enormous amount of) data and perform fast reasoning on top of it in almost no time. Thanks to our brain we are able to recognise entities (e.g., people, locations, objects) in images and text, and identify the relationships that can exist between entities according to our domains of knowledge. Although our knowledge about how the brain works is highly incomplete, we have developed several technologies that try, to some extent, to emulate its operation.

Artificial intelligence (AI) is a cross-disciplinary approach to understanding, modelling, and creating intelligent machines (systems). The aim of AI research and applications is, e.g., to understand speech or images, automate routine tasks (reducing chances of human error), speed up medicine and basic scientific research. These problems may be easy for (expert) people but hard for computers. First approaches in AI projects tried to hard-code the informal knowledge that a person has on any topic. However, such knowledge must be painstakingly handcrafted from scratch and it is mostly subjective and intuitive, which makes it difficult to formalise. These approaches are known as *knowledge base* approaches to artificial intelligence (Firebaugh, 1988).

Decades after with the added factor of massification of the Internet and the Web, we have an increasing number of knowledge bases created in academia and industry alike, generated automatically or manually, and comprising general or domain-specific knowledge. The availability of such a large amount of knowledge has attracted the attention of

AI researchers towards training machines to acquire their own knowledge, by generalising from experience, what is known as *machine learning*. Machine learning became an answer to the problem of requiring people to manually encode rules to make decisions. With machine learning those rules can be extracted automatically from a pile of data. In machine learning algorithms, objects are represented by features and mapped to an output, e.g., mapping the well-known Iris flowers to their specie (setosa, versicolor, virginica) using their representation in terms of sepal and petal length, and width (Fisher, 1936).

While the popularity and use cases in which knowledge bases can be used are increasing in both academic and industrial environments, creating or maintaining complete knowledge bases is still one of the main challenges for users and the community. Knowledge bases are never complete for large and open domains—and rarely complete even for very specific domains—making them error-prone and hard to exploit, let alone to reason with them (Motro, 1989; Drumond, Rendle, and Schmidt-Thieme, 2012; Razniewski, Suchanek, and Nutt, 2016). Since knowledge bases find wide applications in areas of life sciences, natural language processing, and financial sector, among others, the task of completion of knowledge bases has attracted much attention recently in the literature (Socher, D. Chen, Manning, et al., 2013; Gardner and Mitchell, 2015; Q. Wang, B. Wang, and L. Guo, 2015; Q. Wang, J. Liu, Y. Luo, et al., 2016). In this thesis, we focus on designing and applying machine learning techniques to deal with certain knowledge bases completion tasks.

Methods to help the exploitation of knowledge bases usually represent them as graphs, where entities are nodes connected by edges that represent relationships between entities, resulting in a so-called *knowledge graph* (Nickel, K. Murphy, Tresp, et al., 2016; Paulheim, 2017; Q. Wang, Mao, B. Wang, et al., 2017). While it is easy in some cases to identify relevant features in relational data and use them to apply machine learning algorithms, in knowledge graphs it becomes difficult to do the same due to the graph topology. Say you have a knowledge graph that contains information about articles and authors, you can easily think of relations such as title, authors, publisher and year of publication to be used as features. However, if a knowledge graph contains information about more abstract or complex domains such as life sciences, e.g., describing drugs and adverse drug reactions, it is harder to think in relevant features if one is not a domain expert. Nevertheless, even if one does not have an explicit schema at hand, there is always a latent schema inherently satisfied by the data (Hogan, 2018). An example is the cardinality information, which has shown to be a powerful tool to understand the data and its structure (Liddle, Embley, and Woodfield, 1993) even by non-expert users, but this information is almost never explicit in knowledge graphs (Mirza, Razniewski, Darari, et al., 2017; Muñoz and Nickles, 2017).

Approaches applying machine learning over knowledge graphs roughly fall into two categories: (a) observed feature models, and (b) latent feature models. Observed feature models generate graph-based features from a knowledge graph, such as subgraphs, connecting paths and neighbourhood information, which can be easily interpretable (Lao, Mitchell, and Cohen, 2011; Gardner and Mitchell, 2015; Ristoski and Paulheim, 2016b). On the other hand, latent feature models embed entities and relations into a latent vector space

learnt by minimising a loss function, and perform inference in that space (Bordes, Weston, Collobert, et al., 2011; Nickel, Tresp, and Kriegel, 2011). The former category of models is more suited for use cases that require interpretability of the predictions (e.g., in life sciences), while the latter models are seen as black boxes usually harder to interpret (Ribeiro, S. Singh, and Guestrin, 2016; Zilke, Loza Mencía, and Janssen, 2016). Yet latent feature models for knowledge graphs have shown to be more effective than observed feature models in several tasks (Nickel, K. Murphy, Tresp, et al., 2016; Q. Wang, Mao, B. Wang, et al., 2017), despite the fact that their assumption of a "complete" knowledge graph is never met.

Recent works in knowledge representation using symbolic logic and automated reasoning in knowledge graphs (Bordes, Usunier, García-Durán, et al., 2013; B. Yang, Yih, X. He, et al., 2015; Nickel, K. Murphy, Tresp, et al., 2016; Trouillon, Welbl, Riedel, et al., 2016) have proposed approaches where deep learning (Goodfellow, Bengio, and Courville, 2016), a subset of machine learning, is used to discover the mapping from representation to output along with the representation itself. In other words, the representation of the entities and relations in a knowledge graph is learnt during the training process. This approach is known as *representation learning* (Goodfellow, Bengio, and Courville, 2016). Thus, AI systems using knowledge graphs require little intervention and easily adapt to new tasks. Representation learning methods achieve remarkable better performance results than handcrafted features, but they usually require much larger amount of annotated data for training. They learn low-dimensional representations—so-called *vector representations* or *embeddings*—of entities and relations from the relational information contained in the knowledge graph.

Although the representation learning methods are trained to preserve as much information as possible compared with the input knowledge graph, the resulting embeddings are known to break some expected properties (Rocktäschel, S. Singh, and Riedel, 2015; Minervini, Costabello, Muñoz, et al., 2017). Latent feature approaches require of large datasets to learn good representations about entities and relations in a knowledge graph, and they tend to be inaccurate for relations with sparse data. Rocktäschel, S. Singh, and Riedel (2015) introduce a novel training paradigm for learning embeddings that combine matrix factorisation with first-order logic formulæ, thus, learning embeddings that satisfy logical background knowledge. Likewise, other works have proposed approaches to enhance embeddings learning by introducing logical rules (S. Guo, Q. Wang, L. Wang, et al., 2016), constraints (Minervini, Costabello, Muñoz, et al., 2017; Ding, Q. Wang, B. Wang, et al., 2018), and neighbourhoods information (D. Q. Nguyen, Sirts, L. Qu, et al., 2016), among others. (We refer the reader to Q. Wang, Mao, B. Wang, et al. (2017) for a survey of approaches learning knowledge graph embeddings with background knowledge.) However, there are still missing approaches that consider structural elements of knowledge graphs like the cardinality of relations.

To summarise, knowledge graphs are graph-structured knowledge bases, rich in information, where relationships or connections between entities are first-class citizens. The structure of knowledge graphs is not always explicitly stated and, thus, most machine learning algorithms ignore and do not considered the topology of data. Therefore, there

is a primary need for methods that expose the structure or shape that knowledge graphs data naturally exhibit in order to inform and bootstrap learning methods in novel areas such as Bioinformatics. If available such kind of information could be incorporated during the learning of distributed representations and used to improve the quality of downstream tasks over knowledge graphs (e.g., the completion task). Consequently, one would like to develop methods that incorporate the structure factors found in knowledge graphs to improving their completeness and representation of knowledge in real-world scenarios.

In the following, we formulate the research questions that we find are key to address current gaps in the state-of-the-art of knowledge graphs mining, namely, how to expose their schema structure and how to use such information in machine learning.

1.2 Research questions

The research carried out in this thesis is guided by four research questions defined around the study of consistency and completeness quality dimensions in knowledge graphs. The problem that we seek to address in this thesis is how to automatically extract structural information (shapes) from knowledge graphs, which can then be used to improve the performance of machine learning algorithms trying to complete the knowledge graph itself. More specifically, in this thesis we investigate the following research questions:

Research Question 1 *What type of patterns are able to expose the inherent shape information (topology) of a given knowledge graph, and how these patterns can be extracted efficiently when no schema or ontology is available?*

One of the oldest knowledge graphs is Cyc (Lenat, 1995) (started in 1984), attempting to assemble a comprehensive ontology for common sense knowledge describing how the world works, and thus enabling AI applications with human-like common sense reasoning. The level of description and curation required to build Cyc is enormous and hard to mimic—the initial estimated build time of Cyc was 5 years (350 person-years), but the effort has been prolonged and estimated already to be over 900 person-years back in 2009 (Sarjant, Legg, Robinson, et al., 2009). As this example shows, knowledge graphs are naturally schema-less giving them huge flexibility, but creating them requires a huge effort for defining their structure and constraints. To assess consistency, data generators only state basic and simple constraints that roughly cover entity types, possible relationships between entities, and their data types. Other type of constraints are rarely stated and left to be defined by the end-users (data consumers) depending on their specific use cases. Thus, it is hard for data consumers to know what is the shape of the information contained in a knowledge graph beforehand.

The most simple and effective way to understand the structure of a knowledge graph is analysing the relationships between entities, that is, edges or links in the graph (Hayes and Gutiérrez, 2004; Lao and Cohen, 2010; Gutiérrez, Hurtado, Mendelzon, et al., 2011;

Fernández, Martínez-Prieto, Fuente Redondo, et al., 2018). Although understanding the constraints at a relationship level is a critical aspect of data and knowledge modelling, the extraction of such constraints has not received much attention in knowledge graphs, mainly due to the complex structure and the semantics used to interpret the data (as we will se in Section 2.1.3). Therefore, there is a clear need for methods that unveil the structural patterns that entities and relationships naturally follow in a knowledge graph. We hypothesise that a notion of relation cardinality could help unveiling the structure of entities in the data. Such patterns could then be easily encoded using popular constraint languages and used by data consumers to determine the suitability of a knowledge graph for a given use case. This question focuses on finding scalable approaches for extracting cardinality constraint patterns in different sized datasets.

Research Question 2 *How to validate large-scale and noisy knowledge graphs using an approximate and more granular approach?*

RQ (1) assumes that no schema is provided for a knowledge graph, and thus its quality is most likely unknown. However, once a schema is defined, data consumers may require to know whether *all* the input data in a knowledge graph satisfies the given schema. This becomes challenging for large-scale knowledge graphs considering that, in practice, most of the validations are done using inference engines and ontology languages (such as the Web Ontology Language, OWL (Motik, Peter F. Patel-Schneider, and Parsia, 2012)), which have scalability issues. Traditionally, validation requires to take the closed-world and unique name assumptions; however, OWL is arguably not meant for that. The W3C community has been working on a few option to tackle this gap, and in 2017 released two popular alternatives: Shapes Constraint Language (SHACL) and Shape Expressions (ShEx). Even after data users define schemas using a constraint language, the consistency of knowledge graphs it is not static (i.e., cannot be ensured all the time), considering that knowledge graphs are dynamic and constantly enriched with new information. The only route to ensure that the data actually satisfy the restrictions is validating the data against a schema.

Strict and full validation of a knowledge graph is expensive and approximate solutions could help users to save a considerable amount of resources, i.e., computing time and memory. Assuming that knowledge graphs contain several link patterns, it may be argued that by knowing the validity of a small set of entities, the in-/ validity value can be passed to other entities with similar local patterns—directly related to the constraints in a schema. For instance, if one knows that entity e_1 is valid against a schema S, then we may assume that a very similar entity e_2 should also be valid w.r.t. S. As the structure of knowledge graphs may be quite complex, it has to be ensured that whatever are the patterns used for validation, they have to be (a) easy to extract in large graphs, and (b) dynamic to adapt to new triples that may be added in the future.

Research Question 3 *How to influence the performance of machine learning algorithms trying to complete a knowledge graph using cardinality and subgraph patterns present in the same*

knowledge graph?

Knowledge graphs are largely sparse and contain only positive information. The knowledge graph completion task deals with the automatic understanding of the structure of knowledge graphs to predict missing relationships or links. Due to the many applications of this task it has quickly become one of the main research areas in statistical relational learning (Getoor and Taskar, 2007). As we have mentioned previously, several large-scale open domain knowledge graphs are available on the Web such as Wikidata, DBpedia, NELL, among others (Vrandecic and Krötzsch, 2014; Mitchell, Cohen, Jr., et al., 2018). Among them, there is a good number of domain-specific knowledge graphs that aim to achieve a better completeness in a more bounded domain of knowledge like Bioinformatics (Dumontier, Callahan, Cruz-Toledo, et al., 2014; Law, Knox, Djoumbou, et al., 2014; Kim, Thiessen, Bolton, et al., 2015; Banda, Evans, Vanguri, et al., 2016). Knowledge graphs have different characteristics (e.g., size, license, etc.) and complex structure. Because of the previous and knowledge graphs' schema-less nature it is not trivial for machine learning models to exploit their content. Here we expect to build upon RQs (1) and (2) and naturally think of using the latent shapes in a knowledge graph to improve the performance of data mining and knowledge discovery algorithms, which are usually fed with relational data.

The challenge is then how to encode heterogeneous and complex graph data into features for machine learning. A few approaches have been proposed for combining semantic data and machine learning (Ristoski and Paulheim, 2016b). Features generated from the structure of knowledge graphs are easy to interpret by humans and help to validate prediction results. In particular, we would like to explore the Bioinformatics domain and discover missing links between drugs and side effects.

Research Question 4 *How to incorporate cardinality constraints as additional structural information into the process of learning latent representations from knowledge graphs?*

Link patterns studied in the above research questions can be categorised as *observable* patterns in the graph data, e.g., a path connecting two entities (Lao and Cohen, 2010). Recent works have shown, however, that *latent* feature models have the ability to learn better representations for entities and relations in a knowledge graph yielding better prediction results, but at the cost of interpretability (Nickel, Tresp, and Kriegel, 2012). The derived latent representations (a.k.a. vectors or embeddings) cannot be directly mapped to observable patterns in the data, thus, may not have a direct meaning for end users.

Knowledge graphs express data as a directed graph with labelled edges (relations) between nodes (entities). However, since entities and relations do not have their own embeddings, the core potential of reasoning and inference in knowledge graphs cannot be fully exploited (Socher, D. Chen, Manning, et al., 2013; S. Guo, Q. Wang, L. Wang, et al., 2016; Minervini, Costabello, Muñoz, et al., 2017; W. Chen, Xiong, Yan, et al., 2018). In addition to our contributions to traditional machine learning, we also aim to identify the benefits of using structural information in the learning of latent representations for a knowledge graph.

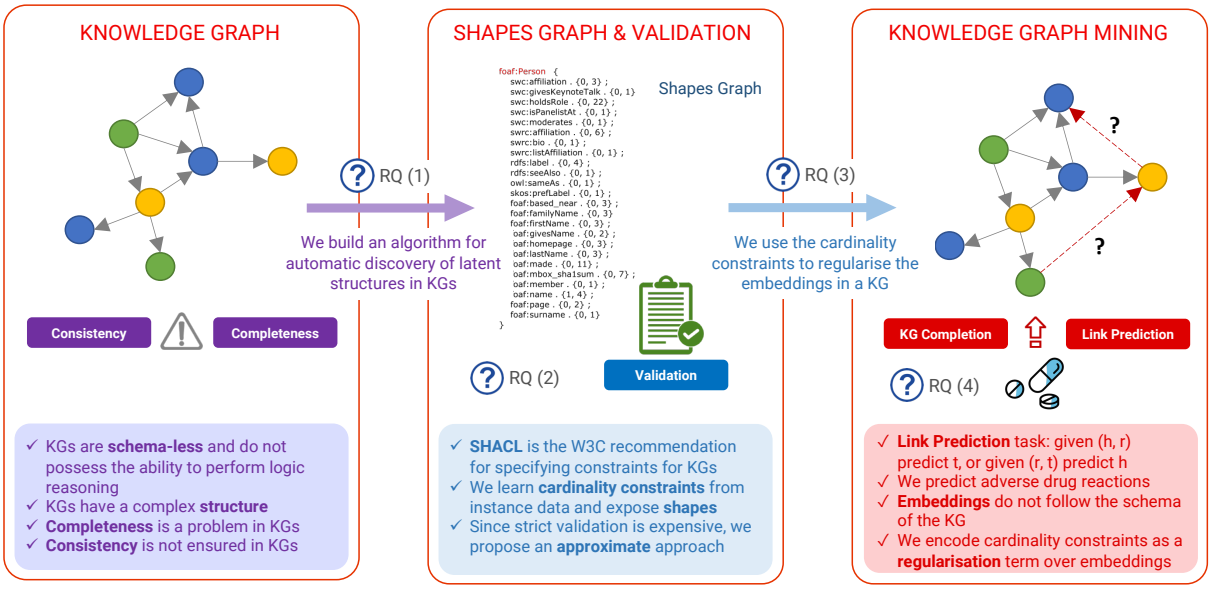


Figure 1.1: Overview of the proposed research and contributions of this thesis.

More specifically, we will attempt to enforce constraints satisfied by the knowledge graphs over the learnt embeddings. For example, if in the knowledge graph an entity has a relation of cardinality n, we expect this constraint to be satisfied by the embeddings.

Given the applicability of representation learning in knowledge graphs, contributions towards the improvement of latent representations become very relevant and applicable to different downstream tasks. This raises a sub-question, that aims to measure the effects of constraining the embeddings in the link prediction task over benchmark datasets.

4.1 *What are the effects in the performance of the link prediction task when using cardinality constraints as regularisation in knowledge graph embedding models?*

1.3 Contributions

Through this thesis, we combine several concepts from Graph Theory (e.g., paths, subgraphs), Databases (e.g., constraints, validation), and Machine Learning (e.g., classification, link prediction). We combine these concepts and define important aspects of the structure of knowledge graphs that can then be used in data mining and knowledge discovery. Figure 1.1 provides an overview of the concepts, research questions, and contributions made.

From our study of the four research questions stated above, we do a set of technical contributions. In particular, this thesis makes the following technical contributions:

- We provide an up-to-date overview of languages to define schemas for knowledge graphs (Section 3.1).

- We review approaches to inferring schema information from knowledge graphs (Section 3.2).
- We present an extensive review of the knowledge graph completion problem (Section 3.3) and the statistical relational learning models applied to the completion problem: graph-based feature approaches (Section 3.4.1) and latent feature approaches (Section 3.4.2).
- We introduce a principled approach using SPARQL query language for mining cardinality bounds from knowledge graphs, where cardinalities unveil the inherent structure of data, and show how to boost its performance by up to 40x using Apache Spark (Section 4.3).
- We show that local patterns for entities in the form of subgraphs or neighbourhoods can be efficient to extract and used to propose a novel approximate validation for large knowledge graphs using machine learning (Section 5.3).
- We present a novel approach to discovering adverse drug reactions using knowledge graphs and machine learning (Section 6.3).
- We empirically show the benefits of considering knowledge graphs for knowledge discovery in Bioinformatics obtaining new state-of-the-art results (Section 6.6).
- We present a novel regularisation term to encode cardinality information of relations into loss functions used to train knowledge graph embedding models (Section 7.3).
- We empirically analyse the benefits of our novel cardinality regulariser on state-of-the-art knowledge graph embedding models and benchmark datasets (Section 7.4).

1.4 Thesis structure

The content of this thesis is organised in four sequential parts that we illustrate in Figure 1.2 and summarise as follows.

Part I: Background. We dedicate the first part to present all fundamentals required to understand the content of this thesis. First, due to the different fields where knowledge bases, and most recently knowledge graphs, have been used we differentiate between semantic and non-semantic knowledge graphs from a data modelling perspective.

Following the scope defined in this introduction, in Chapter 2 we will review the fundamentals of machine learning and problems related to the quality analysis of knowledge graphs. We discuss the related work on two important quality dimensions for knowledge graphs, namely, consistency and completeness. Both of these dimensions have huge implications in tasks related to data mining and knowledge discovery in knowledge graphs.

Both consistency and completeness have their own challenges, but they are intertwined with one another in knowledge graphs. In Chapter 3, we explore the aspects related to *knowledge graph mining* and its connections with these quality dimensions. We review the main problems and approaches studied in literature around the topics of schema languages

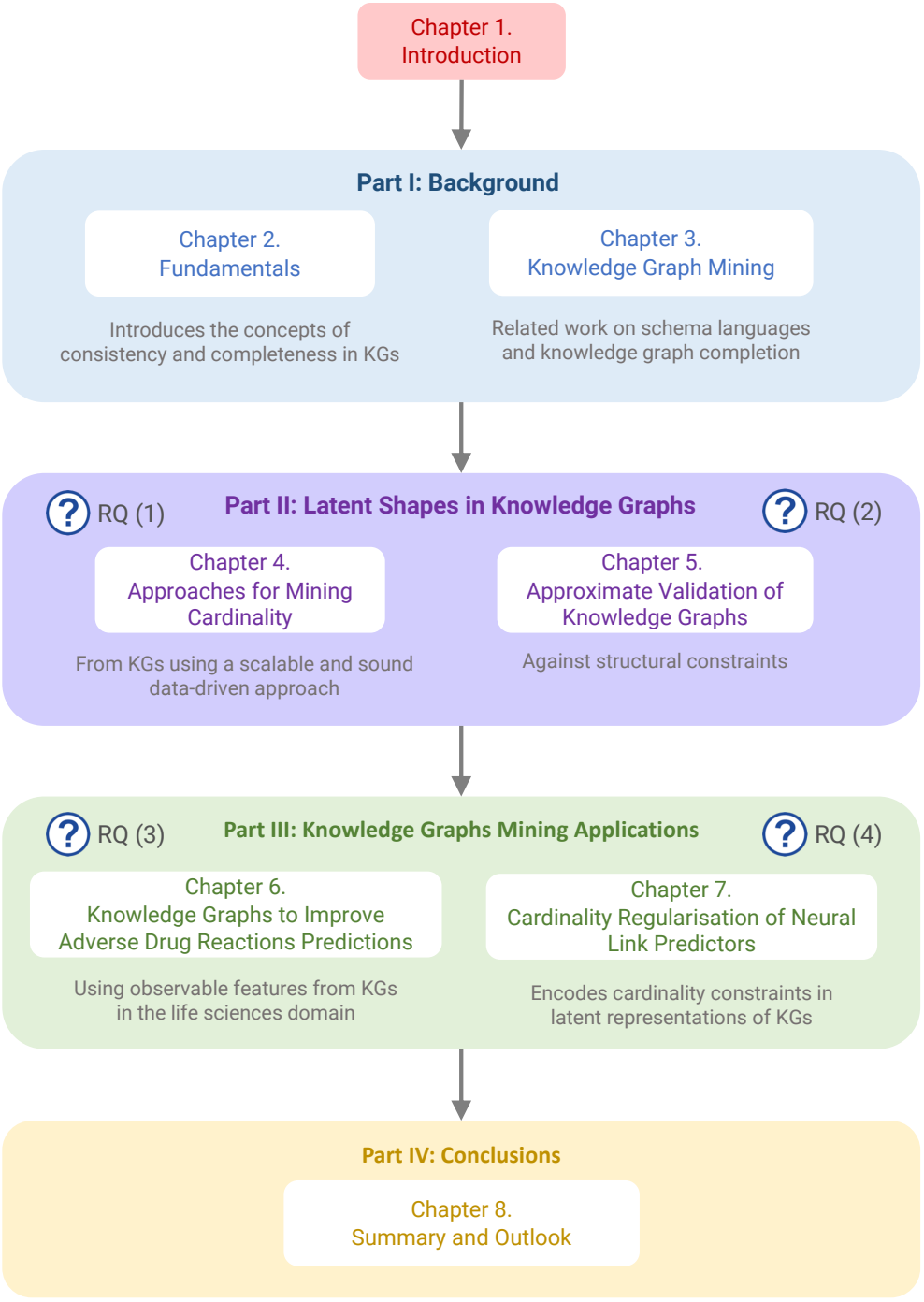


Figure 1.2: Thesis structure by parts and chapters.

and schema inference in knowledge graphs. We then review existing approaches for completion of knowledge graphs using statistical relational learning models.

Part II: Latent Shapes in Knowledge Graph. In this part, we address RQ (1) and the problem of missing schema for knowledge graphs, and propose data-driven approaches to mine patterns that could be used to build a schema for a given knowledge graph. To this end, in Chapter 4 we push forward the use of cardinality constraints to exhibit the natural structure of knowledge graphs, and propose a mining algorithm using the SPARQL query language. However, our SPARQL approach has some limitations that we overcome by proposing an efficient mining algorithm based on the well-known divide-and-conquer method, which delivers equivalent results in a more efficient way—using less time and memory. Once the important parts of the structure of a knowledge graph are made explicit, they can be used to define latent shapes (shapes graphs), and to check consistency of new content.

In Chapter 5, we address our RQ (2) and propose a novel approximate solution for validating the content of a knowledge graph using machine learning algorithms, when accuracy-efficiency trade-offs are allowed—errors are accepted in order to speed up validation. We follow the assumption that entities following similar structural patterns should share the validity state w.r.t. a given shapes graph. We test our assumption and approach using knowledge graphs from the Web Data Commons project, which contain noisy triples.

Part III: Knowledge Graph Mining Applications. This part is dedicated to the applications of structural patterns learnt from knowledge graphs in two areas. First, in Chapter 6 we address RQ (3) and present an application in the Bioinformatics domain, encoding entity patterns as observable features used to boost the performance of classical machine learning algorithms. More specifically, we validate a cardinality based propositionalisation for entities that can be used to feed information from knowledge graphs to different learning algorithms. The specific problem we target is the prediction of adverse drug reactions using a knowledge graph that we build by merging information about drugs, proteins, and side effects from different sources. We evaluate our approach using existing benchmarks and compare against several machine learning models obtaining new state-of-the-art results.

Second, in Chapter 7, we study the use of cardinality information in recently introduced neural representation learning models for knowledge graphs. Neural link prediction models aim to jointly learn a scoring function to score facts and the distributional representation (a.k.a. embeddings) of entities and relations in the knowledge graph by minimising a loss function. Since the learnt embeddings do not respect schema information—cardinality more specifically—we propose a regularisation term to enforce these patterns during training; hence, addressing RQ (4). For the evaluation of RQ (4.1), we use standard benchmark datasets derived from Freebase, YAGO, and WordNet databases to predict missing link.

Part IV: Conclusion. We dedicate the last part, consisting of Chapter 8, to conclude this thesis by summarising our findings from the previous three parts. We then list the limitations of the proposed approaches and discuss possible research avenues for the future of knowledge graph mining.

Part I

Background

CHAPTER 2

Fundamentals

Contents

In this chapter, we will lay out the main concepts used throughout this thesis, namely, knowledge bases and knowledge graphs. We pay special attention to the quality dimensions of completeness and consistency that have attracted considerable attention given the applicability of knowledge graphs in artificial intelligence problems. We review the definition of Semantic Web knowledge bases and how they have been interpreted by practitioners and researchers. We then review the recent concept of relational knowledge graphs, which correspond to a recently popular view over knowledge bases under the lenses of graph theory. The graph view is maybe the most important characteristic of knowledge graphs that provides great flexibility and has been welcomed by different communities that found hard to adopt the previous Semantic Web solutions. Despite the fact that knowledge graphs are

interpreted as graphs, they have certain particularities that make their interpretation even more complex than simple graphs. We identify some of these particularities and discuss their relation with the completeness and consistency dimensions of data quality. These two dimensions of quality have been tackled using machine learning techniques that provide high-performance solutions and opportunities for dealing with the uncertainty found in most knowledge graphs. Finally, we present a brief overview of machine learning and deep learning frameworks relevant in the context of this thesis.

2.1 Knowledge bases

A *knowledge base* (KB) is a set of statements that describe the knowledge about the truths of the actual world plus a set of constraints that describe statements that must be true in all possible worlds and statements that ought to be true (in all possible worlds) (Reiter, 1986; Dignum and Riet, 1991). This definition assumes that the knowledge of the world may be incomplete and that knowledge bases may contain temporal information. A knowledge base differentiates between facts and constraints. Sometimes facts are also seen as constraints, e.g., the following statement "all Irish citizens live in Ireland" may seem to be a general fact, but that does not make it a constraint. Some day an Irish citizen might decide to move outside of Ireland, which would make the statement false. On the other hand, there are constraints that are always true and constraints that ought to be true (Dignum and Riet, 1991). Dignum and van de Riet provide a clear example for this:

(a) All people have an age which is non-negative, and
(b) All people that borrow a book must return it (within 3 weeks).

The first constraint is clearly always true, whilst the second may be violated.

Together knowledge bases and *inference engines* were originally considered as the building blocks of *expert systems*, which are known as the first knowledge-based systems in Artificial Intelligence. A knowledge base stores structured and unstructured data using a flexible schema—and thus differs from the more common term *database*—to represent facts about the world. The facts in a knowledge base are fed to an inference engine that can reason about those facts applying logical rules for deducing new facts (information) or identifying inconsistencies (Hayes-Roth, 1983). Unlike databases, the structure of a knowledge base is richer and equivalent to a theory in First-Order logic or an *ontology*, and defines the concepts and relationships used to describe and represent a domain. A Description Logics knowledge base is a pair $(\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ (or T-Box) is the set alignments of classes and properties (i.e., schema), similar to object-oriented classes, and $\mathcal{A}$ (or A-Box) is the set of instances (materialised types) of those classes (i.e., data). Knowledge bases focus mainly on the A-Box than on the T-Box, henceforth, we regard them as A-Box and ontologies as T-Box. Knowledge bases are one of the most (if not the most) important component of the *Semantic Web* (Berners-Lee, J. Hendler, and Lassila, 2001) providing the

knowledge exchange part.[1] "The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries" (https://www.w3.org/2001/sw/). The vision of the Semantic Web is to convert the traditional Web from a Web of Document into a Web of Data, where we do not only have documents and links between them, but typed entities (e.g., *Alan_Turing* of type Person, *Galway* of type Location) and relations between entities (e.g., *bornIn*, *studiedAt*) available on the Web. One of the main ideas behind the Semantic Web was to bring the structure and provide a well-defined meaning to Web pages. Such "meaning" could then be used by software applications to enable computers and people to cooperate and unleash endless possibilities—the data in Web pages would not only be visible but could also be understood.

This framework builds upon several technologies interconnected in what is called the *Semantic Web Stack*. At the bottom of this stack we find one of the fundamental standards for data modelling in the Semantic Web, the *Resource Description Framework* (RDF) (Cyganiak, Wood, and Lanthaler, 2014) used for describing Web resources and their relationships. RDF allows to make statements about resources using expressions of the form *subject–predicate–object*, known as *triples*. The *subject* denotes the resource, and the *predicate* expresses the relationship between the *subject* and the *object*. Objects can be resources or string literals.

An ontology defines the vocabulary of concepts and relations in a knowledge base. Ontologies consist of: (i) a set of *properties*, which are binary relations between subject resources and object resources; (ii) a set of *classes*, where elements of a class are known as *instances* of that class; and (iii) the possible constraints on the properties and classes. For instance, rdfs:subClassOf, rdfs:domain, rdfs:range, and rdf:type are properties in Item i; while schema:Country, foaf:Person, and dbo:SportsClub are classes in Item ii that can be used to state the class of an instance using the rdf:type property, e.g., (*Chile*, rdf:type, schema:Country) (Arenas, Gutiérrez, and Pérez, 2009).[2] Additionally, one could state constraints like "an instance of the class Parent is who has at least one relation *hasChild* with an instance of class Person."

Next, we present a more formal definition of Semantic Web knowledge bases, and the common assumptions adopted for their interpretation either locally (e.g., by agents) or globally (e.g., in the World Wide Web). For a thorough description of the Semantic Web Stack, we refer the readers to the W3C standards documentation (https://www.w3.org/2001/sw/).

2.1.1 Semantic Web knowledge bases

Knowledge bases in the Semantic Web are represented using the Resource Description Framework (RDF) data model, therefore they are also called *RDF graphs*. We define a *Semantic Web Knowledge Base* following Arenas, Gutiérrez, and Pérez (2009).

[1] http://bit.ly/TheSemWeb Archived version of the Scientific American article, May 2001 by Tim Berners-Lee, James Hendler and Ora Lassila.

[2] Henceforth, we will use abbreviations such as rdfs, rdf, schema, foaf, or dbo to refer to vocabulary namespaces. A full list of these mappings is available at http://prefix.cc/.

Definition 2.1

Let $\mathcal{R}$ be the set of *entities* (resources), $\mathcal{B}$ the set of *blank nodes*, $\mathcal{P}$ the set of *predicates*, and $\mathcal{L}$ the set of *literals*. A finite *knowledge base* $\mathcal{K}$ is a set of *triples*, say $\mathcal{K} = \{(s,r,o)_i\}_{i=1}^{m}$, where each $(s,r,o) \in (\mathcal{R} \cup \mathcal{B}) \times \mathcal{P} \times (\mathcal{R} \cup \mathcal{B} \cup \mathcal{L})$. In each (s, r, o), s denotes the *subject*, r denotes the relation *predicate*, and o denotes the *object*. The sets $\mathcal{R}$, $\mathcal{B}$, and $\mathcal{L}$ are infinite and pair-wise disjoint.

Figure 2.1 presents an small example of a Semantic Web knowledge base with the separation between ontology and instance data. Graphically, each triple (s, r, o) is represented by a labelled edge $s \xrightarrow{r} o$. In the definition of RDF, the set of edge labels can be a non-empty intersection with the set of node labels. This differentiates a Semantic Web knowledge base from a graph in the classical sense (Arenas, Gutiérrez, and Pérez, 2009).

We also define some auxiliary functions as follows:

- $pred(\mathcal{K}, \tau) = \{r \mid \exists s, o\ (s,r,o) \in \mathcal{K} \wedge (s, \texttt{rdf:type}, \tau) \in \mathcal{K}\}$ returns the set of predicates appearing with instances of entity type τ;
- $proj(\mathcal{K}, s, r) = \{(s,r,o) \mid \exists o\ (s,r,o) \in \mathcal{K}\}$ returns the triples in $\mathcal{K}$ whose projected subject is s and predicate is r; and
- $sameAsPairs(\mathcal{K}) = \{(s,o) \mid \exists\ s, o\ (s, \texttt{owl:sameAs}, o) \in \mathcal{K}\}$ as the function that returns all pairs of equivalent entities but defined using a different naming, e.g., *U.S.* and *United_States_of_America*.

Note that the predicate `owl:sameAs` is used in most cases to represent equality, but other predicates could also be used for the same purpose.

Among the most popular Semantic Web knowledge bases are: DBpedia (Auer, Bizer, Kobilarov, et al., 2007) project that extracts structured content from articles in Wikipedia; Freebase (Bollacker, R. P. Cook, and Tufts, 2007) a collaborative and community generated knowledge base comprising structured data from different sources; Wikidata (Vrandecic, 2012) a collaborative knowledge base supporting Wikipedia with factual information; and YAGO (Suchanek, Kasneci, and Weikum, 2007), an open source knowledge base automatically generated from Wikipedia and other sources.

Semantic Web knowledge bases require of certain characteristics that make them suitable for open environments such the Web. They inherit two main characteristics from ontology languages and description logics, namely, the non-unique name assumption and the open-world assumption, which influence how we can interpret and imply knowledge.

2.1.2 Unique name assumption

The *unique name assumption* (UNA) is a simplifying assumption made in some ontology languages and description logics. It entails that two different names always refer to different entities in the world (Russell and Norvig, 2010). On the one hand, the *Web Ontology*

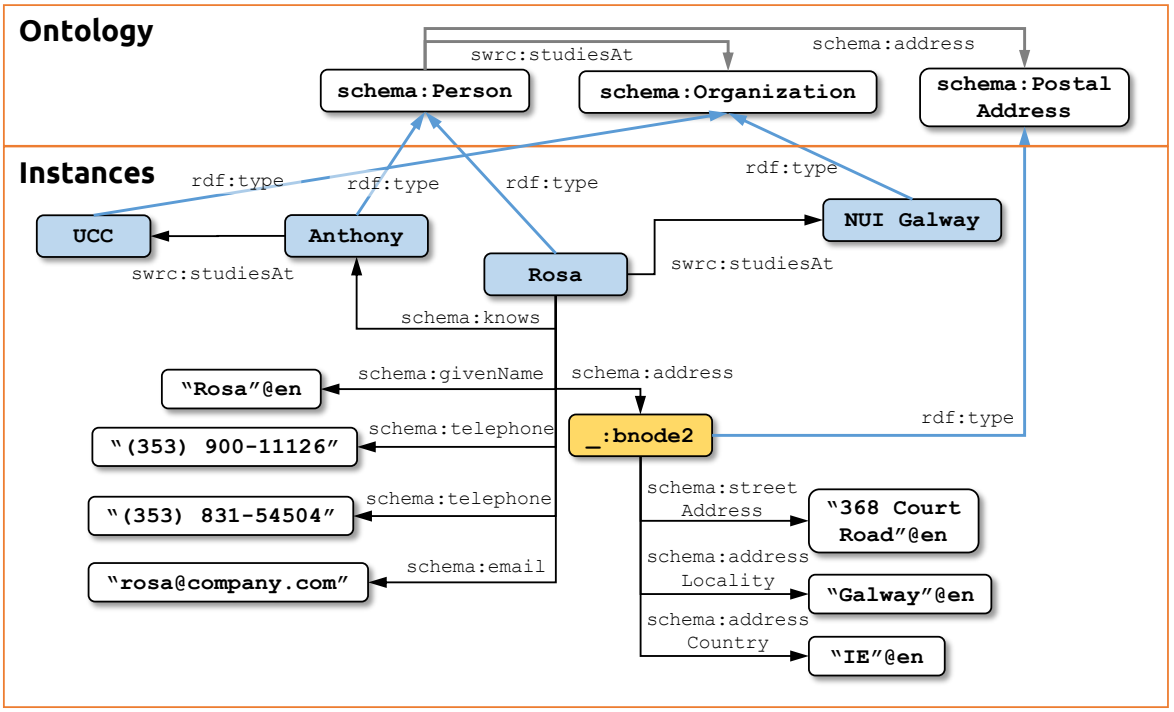


Figure 2.1: Example Semantic Web knowledge base with basic information about people.

Language (OWL) (Motik, Peter F. Patel-Schneider, and Parsia, 2012) default semantics does not adopt the UNA, thus two different constants can refer to the same individual (entity resource). This is a desirable behaviour in an environment such as the Web, where reaching an agreement for labelling entities is infeasible (Horrocks and Tessaris, 2002). On the other hand, validation checking approaches in RDF usually adopt a *closed-world assumption* (CWA) with UNA, i.e., inferring a statement to be false on the basis of failure to prove it, and if two entities are named differently they are assumed to be different entities (Bosch, Acar, Nolle, et al., 2015).

To deal with this, the *Shape Constraint Language* (SHACL) defines the so-called *UNA 2.0*, which is a simple workaround where all entities are treated as different, unless explicitly stated otherwise by `owl:sameAs` (or equivalent) property. From a practical point of view, the CWA is also a desirable feature for data mining algorithms when considering the semantics of Semantic Web knowledge bases, avoiding misinterpretations of the data.

Some properties and patterns in knowledge graphs are highly sensible to the adoption of UNA, e.g., cardinality. Figure 2.2 (left) shows an example where the adoption of classical UNA will lead to a count of five different entities (i.e., `ex:A`, `ex:B`, `ex:C`, `ex:D`, `ex:E`), and `ex:A` would have cardinality one for the property `ex:p1`. While when adopting UNA 2.0 (Figure 2.2, right) these counts change: now we have four different entities (i.e., `ex:A`, `ex:B`, `ex:C`, `ex:E`), and the cardinality of `ex:p1` in `ex:A` increases to 2. Here, we call *rewriting* the process of applying UNA 2.0 to a previously *unnormalised* knowledge base.

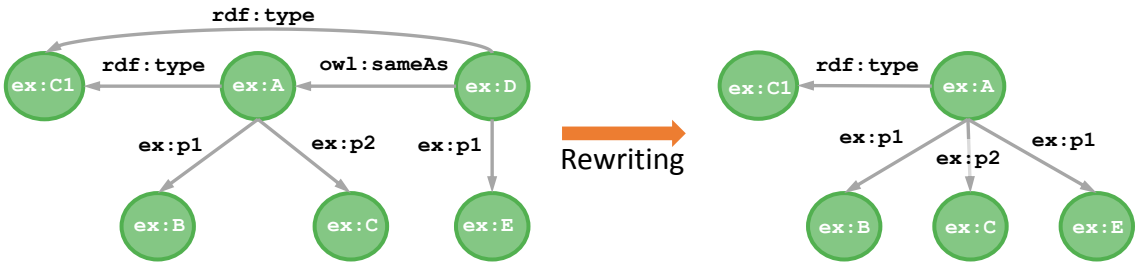


Figure 2.2: Example of UNA 2.0 as defined in SHACL.

2.1.3 Interpretation of knowledge bases

In knowledge representation, there are a few ways to interpret what it is known to be true or what not. In general, no single person or agent can have complete knowledge about the world. Only in cases where the person or agent has complete control over the information (e.g., the domain is very constrained) we may say that everything can be known. This shows a direct relationship between the interpretation of a knowledge base and its completeness. A knowledge base can be interpreted in different ways according to their completeness (Reiter, 1977; Minker, 1982; Levy, 1996; L. A. Galárraga, Teflioudi, Hose, et al., 2013; X. Dong, Gabrilovich, Heitz, et al., 2014; Razniewski and Nutt, 2014; Razniewski, Suchanek, and Nutt, 2016). In the following, we describe the most common assumptions to interpret a knowledge base.

Open-World Assumption (OWA). This assumption says that the truth value of a statement may be true irrespective of whether or not it is in the knowledge base. Commonly, knowledge bases are interpreted under the OWA because they miss facts that hold in the real world, thus are incomplete (Reiter, 1977). It is also a desirable behaviour for collaborative environments to support the creation and addition of new knowledge as statements, e.g., when adding a new entity or integrating two or more knowledge bases. Because of OWA, we cannot tell whether a knowledge base is complete or not given a query. For instance, if we only know the facts (*Aidan*, *bornIn*, *Ireland*) and (*Aidan*, *fatherOf*, *Aoife*), it is unknown whether *Aoife* was also born in Ireland even though her father was. A completeness statement for knowledge bases under OWA could only be possible if we have access to a hypothetical-ideal knowledge base which allows to infer *all true statements*.

Closed-World Assumption (CWA). This assumption is the opposite of the OWA and says that statements that are true are also known to be true, and statements not present in the knowledge base (and that cannot be inferred from it) are false (Reiter, 1977). Under this assumption, a knowledge base is complete and contains all the facts that it attempts to model, usually within a fixed world such a specific topic or domain. By interpreting a knowledge base under CWA we can ensure that the knowledge base is always complete for all queries. For instance, if we only know the facts (*Aidan*, *bornIn*, *Ireland*) and (*Aidan*,

fatherOf, *Aoife*), we also know that *Aoife* was not born in Ireland because that fact is not in the knowledge base. Compared with OWA, CWA is too restrictive and used only when the knowledge base is known to be complete (e.g., small domain), or when the knowledge base is known to be incomplete but that is enough to answer queries. Some authors have proposed extending the framework of the Semantic Web to get the benefits of Semantic Web but still allow forms of CWA and negation as failure (NAF) (see Grimm and Motik (2005), Horrocks, Parsia, Patel-Schneider, et al. (2005), Sengupta, Krisnadhi, and Hitzler (2011), and Patel-Schneider (2015), among others). However, adding CWA and NFA poses new issues and requirements, such as more powerful query languages and modifications to the original framework, that do not nicely fit the goal of the Semantic Web. Yet the CWA assumption is partially favourable and desirable for use cases such as data validation and knowledge discovery. Since in such cases, approaches require to have access to all the knowledge available.

However, in practice, a knowledge base can be complete for some entities/relations, while it is known to be incomplete for others.

Partial Closed-World Assumption (PCWA). This is an intermediate ground between OWA and CWA. PCWA was first introduced in Motro (1989), where queries were used to describe the complete parts of a database introducing what is known as partially complete databases.[3] This assumption says that if a knowledge base contains one value for a subject–relation pair, then we fall into CWA and the knowledge base contains all values for that given subject and relation (Motro, 1989; Levy, 1996; L. A. Galárraga, Teflioudi, Hose, et al., 2013; Razniewski and Nutt, 2014; Razniewski, Savkovic, and Nutt, 2016). However, if it is not known any value for the subject–relation pair, we cannot say anything—we fall into OWA. Clearly, CWA and PCWA look similar; however, PCWA is more cautious making judgements only if an entity has a value for a given relation. Intuitively, PCWA is especially well suited for *functional relations*, where an entity can have at most one object (e.g., identifiers).

In L. A. Galárraga, Teflioudi, Hose, et al. (2013), PCWA is used to generate negative examples for a rule mining system, but they refer to it as *Partial Completeness Assumption* (PCA). Likewise, in X. Dong, Gabrilovich, Heitz, et al. (2014), PCWA is used to generate negative examples during the development of Google's Knowledge Vault project—a Web-scale system for probabilistic knowledge fusion, but still the authors refer to it as *Local Closed-World Assumption* (LCWA).

Example 2.1 shows examples of interpreting a knowledge base using the different assumptions just introduced.

[3] Motro (1989) also introduced the notion of incorrect databases which contain facts that do not hold in the real world.

Example 2.1

Clearly, each one of the assumptions have direct implications on the interpretation of a knowledge base. Considering the knowledge base in Figure 2.1 and the statement (*Anthony*, `swrc:studiesAt`, *NUI Galway*), we have:

- OWA: the fact is unknown, and it may or may not be true.
- CWA: the fact is unknown, and false because it is not in the KB.
- PCWA: the fact is unknown, and false because we know at least one `swrc:studiesAt` property for *Anthony*, then we know all of them.

Another example would be considering the fact (*Anthony*, `schema:address`, *_:bnode2*):

- OWA: the fact is unknown, but it may or may not be true.
- CWA: the fact is unknown, and false because it is not in the KB.
- PCWA: the fact is unknown, and we cannot say whether is true or false because we do not know any address for *Anthony* yet.

2.2 Knowledge graphs

Semantic Web knowledge bases usually represent the knowledge using a graph (RDF graphs), where *entities* are nodes of the graph and *relations* are the edges connecting those nodes. In such a graph, entities can also be linked to different types, e.g., *Chile* is an entity of type `Country`. Sometimes one can also find an accompanying *schema* or *ontology* defined using schema or ontology languages, containing entity and relation types, their connections and restrictions according to a given domain. However, not all graph-based knowledge repositories use RDF (graphs) or Semantic Web technologies to manage data.

The YAGO (Suchanek, Kasneci, and Weikum, 2007) project uses a simple triple-based representation using tab-separated values (TSV), where entities and relations are not represented by IRIs or blank nodes but using (locally) unique labels. Similarly, the Never-Ending Language Learning (a.k.a. NELL) (Carlson, Betteridge, Kisiel, et al., 2010) project forms part of the 'Read the web' research project[4], which looks to visit Web pages iteratively and extracts knowledge out of unstructured data in the source pages. The goal of these projects is to extract facts from different sources and build ever-growing knowledge bases. Yet the output of these projects share some characteristics with Semantic Web knowledge bases that we will review below.

The term *Knowledge Graph* was coined by Google in May 2012 for naming their Google Knowledge Graph Search API[5] and Knowledge Vault (X. Dong, Gabrilovich, Heitz, et al., 2014) projects that aim to augment search results and to enhance smart assistance services

[4] http://rtw.ml.cmu.edu/rtw/
[5] https://developers.google.com/knowledge-graph/

like Google Assistant and Google Home. Rapidly, the attention over Google's works made people from both academia and research to adopt and reuse the term 'knowledge graph' when referring to Semantic Web knowledge bases (or RDF graphs) such as DBpedia (Auer, Bizer, Kobilarov, et al., 2007) and Freebase (Bollacker, R. P. Cook, and Tufts, 2007), but also for referring to other knowledge networks such as WordNet (Miller, 1995), NELL (Carlson, Betteridge, Kisiel, et al., 2010), and GeoNames[6]. This term is now a popular wildcard used to refer to *any* graph-based knowledge repository.

Knowledge graphs are closely related to knowledge bases, but usually are considered to have a more limited scope to serve particular a particular purpose. The restricted scope gives a more practical foundation to knowledge graphs that drifts from the universal notion of Semantic Web knowledge bases. Paulheim (2017) argues that actually there is no formal definition for knowledge graphs, and he provides four characteristics that a knowledge graphs should have:

(i) it mainly describes real world entities and their interrelations, organised as a graph;

(ii) it defines possible classes and relations of entities in a schema;

(iii) it allows for potentially interrelating arbitrary entities with each other; and

(iv) it covers various topical domains.

Criterion (i) defines the actual *instances* or triples of a knowledge graph Criterion (ii) defines the schema information Criterion (iii) talks about the possibility to define arbitrary relations between instances without restrictions such as domain/range, and finally criterion (iv) states that knowledge graphs are supposed to cover at least a major portion of the domains in the real world, and are not supposed to be restricted to a single one.

We agree with most of the criteria introduced by Paulheim (2017) but (ii) and (iv). For the case of criterion (ii), we differentiate between instance data and schema and argue that a schema (or ontology) is not always provided along the instances. Moreover, it is not guaranteed that the instance data will be compliant to the schema, specially, considering the flexible nature of knowledge graphs that makes them have an ever changing schema. The absence of schema is relatively common in research areas outside the Semantic Web like Machine Learning with relational data, where the graph structure is the most relevant part exploited for discoveries. Furthermore, we do not consider criterion (iv) to be a required characteristic but an optional or nice to have. We believe that knowledge in sources like GeoNames—with purely geographic entities—are as valuable and rich as in DBpedia or Freebase. Everything depends on the context of the problem one is addressing with the knowledge graph. However, we acknowledge that a cross-domain knowledge graph would be more interesting and potentially more valuable for discoveries than one focused just in one domain. We will validate this in Chapter 6, where we join different data sources from Bioinformatics.

[6] http://www.geonames.org/

2.2.1 Definitions

In this thesis, we adopt the term knowledge graph to refer to Semantic Web knowledge bases and other graph-based knowledge that do not necessarily follow the semantics of RDF. We follow a commonly used mathematical notation for knowledge graphs, which is based on graphs definition. They are also known as *heterogeneous information networks* (Y. Sun and J. Han, 2012; Shi, Y. Li, Jiawei Zhang, et al., 2017) or simply labelled graphs.[7] There is no widely agreed definition on what a knowledge graph is (Ehrlinger and Wöß, 2016). Henceforth, in this thesis we define a knowledge graph as follows:

Definition 2.2

A *knowledge graph* is a quintuple $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \Sigma_{\mathcal{E}}, \Sigma_{\mathcal{R}}, \ell)$ representing an edge-labelled directed multi-graph, where $\mathcal{E}$ is the set of entities, $\mathcal{R} \subseteq \mathcal{E} \times \Sigma_{\mathcal{R}} \times \mathcal{E}$ is the set of directed labelled edges (relations) between two entities, $\Sigma_{\mathcal{E}} \subseteq \Sigma^*$ is a finite set denoted as the *node vocabulary*, $\Sigma_{\mathcal{R}} \subseteq \Sigma^*$ is the finite set denoted as the *edge vocabulary*, and ℓ is the labelling function that assigns a label in $\Sigma_{\mathcal{E}}$ to a node in $\mathcal{E}$, and a label in $\Sigma_{\mathcal{R}}$ to a relation in $\mathcal{R}$. Each node $v \in \mathcal{E}$ represents an entity, and each edge $e = (v_1, r, v_2) \in \mathcal{R}$ represents a r-labelled relationship between entities v_1 and v_2 (*endpoints*), where v_1 is the domain of e, and v_2 is the range of e, denoted $Dom(e)$ and $Ran(e)$, respectively. We also denote by $N_e = |\mathcal{E}|$ and $N_r = |\mathcal{R}|$ the sizes of the sets of entities and relations, respectively. We denote by $\mathscr{G}$ the infinite set of all possible knowledge graphs.

Alternative notations also say that a knowledge graph $\mathcal{G}$ comprises a set of facts or triples (h, r, t) (similar to the (s, r, o) notation in knowledge bases), where $h, t \in \mathcal{E}$ are the head and tail of the relation $r \in \mathcal{R}$. In such cases, it is considered that the edges carry the labels information: subject, relation and object labels, thus, the vocabularies $\Sigma_{\mathcal{E}}$ and $\Sigma_{\mathcal{R}}$ are usually omitted. Thus, we may also sometimes refer to the set of edges $\mathcal{R}$ as knowledge graph, assuming that the set $\mathcal{E}$ can be inferred from $\mathcal{R}$. For exploring a knowledge graph, we would like to navigate the nodes and edges sequentially, using what is called a *path*.

Definition 2.3

A *path* P in a knowledge graph $\mathcal{G}$ is a sequence of edges $\langle e_1, \ldots, e_k \rangle$, where for $1 \leq i \leq k$, $e_i = (v_{i-1}, a_i, v_i)$ is an edge in $\mathcal{G}$ with label a_i and endpoints v_{i-1} and v_i. The domain of a path $Dom(P)$ is the domain of the first edge in the path $Dom(e_1)$, while the range of a path $Ran(P)$ is the range of the last edge in the path $Ran(e_k)$. The integer k is called the *length* of P, denoted by $|P|$. A *path label*, $\ell(P)$, is defined as $\ell(v_0)\ell(e_1)\ldots\ell(v_{k-1})\ell(e_k)\ell(v_k)$, i.e., the concatenation of all node and edge labels on the path P. When only the relations in the path are relevant, a path is called *relation path*. A relation path $R = \langle r_1, \ldots, r_k \rangle$ represents k different relations or hops in a graph.

[7] The term label and type in this case are used interchangeably.

We use $P(u \rightsquigarrow v)$ to denote a variable-length path whose first node is u and last node is v, where u and v are known as endpoints.

Example 2.2

For instance, in our running example of Figure 2.1, we can have the following two paths: $P_1 = \langle Anthony, \texttt{swrc:studiesAt}, UCC \rangle$ and $P_2 = \langle Anthony, \texttt{schema:knows}^{-1}, Rosa \rangle$.

To fully navigate a knowledge graph, we would like to walk through the graph in both directions, meaning that an edge can be traversed in its inverse direction as well. Such navigation is also known as two-way regular path queries (Calvanese, De Giacomo, Lenzerini, et al., 2000; Baeza, 2013), where the set of edges is extended with the inverted edges. This is to avoid the problem of getting stuck in sink nodes with no outgoing edges. For this, we assume that for every edge $e \in \mathcal{R}$ with label $\ell(e) = l$ there is an auxiliary *inverse* edge e^{-1} with label $\ell(e^{-1}) = l^{-1}$ to preserve the semantics of the original directed relations and model cases such as *parentOf* and *hasParent* or *childOf* $^{-1}$.

Given a knowledge graph $\mathcal{G}$, we would also like to extract small portions of the graph centred on one of the nodes. For that we define the notion of *subgraph*. Intuitively, a subgraph represents the local patterns or neighbourhood of a node v, which contains all reachable nodes from v following paths of variable length.

Definition 2.4

Let $\mathsf{Graph}^d(\mathcal{G}, v)$ be the *subgraph* function that extracts $\mathcal{G}' = (\mathcal{E}', \mathcal{R}', \ell')$ around a node $v \in \mathcal{E}$ from $\mathcal{G}$ up to depth d. The set of nodes $\mathcal{E}' \subseteq \mathcal{E}$ in $\mathcal{G}'$ contains all nodes that are reachable in $\mathcal{G}$ from v following paths of length $\leq d$. Similarly, the set of edges $\mathcal{R}' \subseteq \mathcal{R}$, where if $(v_1, a, v_2) \in \mathcal{R}'$ then $v_1, v_2 \in \mathcal{E}'$. Note that when $d = 0$, $\mathsf{Graph}^d(\mathcal{G}, v)$ extracts only the node v with no edges.

2.3 Completeness and consistency in knowledge graphs

Despite all the effort made to build (large-scale) knowledge graphs, they are still far from perfect and usually present problems such as missing information and schema inconsistencies, among others. In this dissertation, we mainly focus on two relevant Data Quality dimensions (R. Y. Wang and Strong, 1996): (1) *completeness*: data do not leave any open questions; and (2) *consistency*: data must be valid according to a set of defined rules or constraints.

Generally, in the context of knowledge graphs—or any knowledge repository—we should assume two things: (a) they are pretty incomplete; and (b) they are pretty complete.

Although these assumptions seem contradictory, in practice, they are actually not. Everything depends on what we are requesting from a specific knowledge graph. For instance, given a knowledge graph about geography such as Geonames, we have some certainty that it will contain at least all seven continents[8]; however, it might not contain all cities in the world. So, we may say that Geonames is complete when answering questions about continents, but incomplete when answering questions about cities.

In terms of data quality, knowledge graphs suffer similar problems to traditional databases (Abiteboul, Hull, and Vianu, 1995). Some information may be unknown. For instance, in a patients medical database, someone's blood type may be missing. This is called *incomplete* information (usually denoted by `NULL` entries). On the other hand, information may also be *inconsistent*. For example, in the same database someone might have two different blood types. A database can contain few cases of patients with two different blood types. In that case, we say that that database has little inconsistency compared with other databases where this is more frequent. In fact, it is considered that there are different levels of inconsistency for data (Grant and Hunter, 2006). Moreover, relational databases do not contain negative information, thus, there is no information about what blood types a patient does not have. All these cases are also valid for knowledge graphs: they comprise only true statements and do not explicitly say what statements do not hold in reality.

In this dissertation, we aim to study the dimensions of consistency and completeness separately, but also how they interact and benefit from each other in the context of knowledge graphs. Next, we describe the related work on both dimensions in more detail.

2.3.1 Consistency

Consistency is a relevant dimension of data quality, and many researchers have investigated the checking and handling of inconsistencies in databases. In the context of knowledge graphs, we define *consistency* as the requirement that a knowledge graph must be valid according to a set of defined rules or constraints (T. A. Nguyen, Perkins, Laffey, et al., 1985). Herein we define a *constraint* as a rule (prescriptive pattern) created at design time that data must satisfy all the time to maintain the consistency of it. Such a set of rules and/or constraints is usually presented as an *ontology*. An ontology mainly defines entity types and their relationships, but can also explicitly state the domain and range of relations, e.g., an instance of type Person can have a relation *name* with a literal value. Ontologies are mostly handcrafted and their construction requires of huge effort. For instance, Cyc is a curated knowledge graph developed back in the 1980s (Lenat, 1995) that contains hundreds of thousands of terms in the domain of human knowledge for supporting Artificial Intelligence tasks. It is estimated that more than 900 person-years of effort have been invested in its creation (Sarjant, Legg, Robinson, et al., 2009). Yet Cyc is far from complete.

[8]Here, we consider the main land divisions as continents: Asia, Africa, North America, South America, Antarctica, Europe, and Australia.

Ontologies are good because of their interoperability and reusability in various applications, and there are also numerous languages, standards and software for processing them (see Noy, Sintek, Decker, et al. (2001)). However, one of the main criticism against ontologies is that they are usually too complex and not intuitive for general public, limiting their use only to specialized users. The large computational resources and time required for checking facts against an ontology are also concerns for real applications and practitioners. Still for practitioners, ontologies were, for a long time, the only way to express constraints and validate the consistency of knowledge graphs (knowledge bases). In principle, such practice is wrong because ontology languages, such as OWL (based on description logics), adopt the OWA and they should not be interpreted under a CWA just for checking the consistency of data. To overcome this limitation, Tao, Sirin, Bao, et al. (2010) proposed to use OWL expressions with CWA and a weak variant of UNA to express integrity constraints. In Section 5.2, we will discuss other approaches that have been proposed for validating knowledge graphs.

On 9th January 2018, the World Wide Web Consortium (W3C) published the SHACL Shapes Constraint Language, a standard generated by the RDF Data Shapes Working Group[9] for validating RDF graphs against a set of conditions. SHACL proposes to express these conditions as shapes (also called "shape graphs") that can be used to validate whether or not data graphs satisfy a set of conditions. Previous to SHACL, in 2013, Shape Expressions (ShEx) was proposed to provide a human-readable syntax for declaring shapes (E. Prud'hommeaux, Labra Gayo, and Solbrig, 2014). ShEx language is based on regular expressions and RelaxNG (a schema language for XML) and it was submitted as a W3C member submission in 2014.[10] (SHACL and ShEX are the two most adopted constraint languages for knowledge graphs, and we will review them later on in Section 3.1.2.)

The goal of shape graphs (expressed using SHACL, ShEx or another language) is to accept and represent data which is valid with respect to a schema. Such schema information is valuable, among other things for communicating data structures to interfaces, generating or validating data, or driving user interface generation and navigation (Labra Gayo, E. Prud'hommeaux, Boneva, et al., 2018). Shape-aware applications can benefit from knowing that the instance data conforms with a given schema.

2.3.2 Completeness

In their seminal paper, R. Y. Wang and Strong (1996) describe completeness as "the extend to which data are of sufficient breadth, depth, and scope for a task at hand." Although this definition has been widely adopted in industry, it is still not very well-suited for knowledge bases. Here, we use an adaptation of the definition of completeness for a query according to Razniewski, Suchanek, and Nutt (2016) and Lajus and Suchanek (2018), which is based

[9] https://www.w3.org/2014/data-shapes/ (Accessed on June 5th, 2018)
[10] http://www.w3.org/Submission/shex-defn/ (Accessed on June 5th, 2018)

on the work in databases (Motro, 1989; Levy, 1996; Razniewski, Korn, Nutt, et al., 2015). *Completeness* is the extend to which a given knowledge base contains all true statements.

Completeness is defined by help of a hypothetical *ideal* knowledge base $\mathcal{K}^*$, which is the instance of all *true* statements, i.e., "the real world". Next, we build upon Motro (1989) to give a more concrete definition of completeness for knowledge bases. Let $\mathcal{K}$ be the instance of all *stored* facts, i.e., "the knowledge base" as a database. Figure 2.3 illustrates these sets using a Venn diagram. Using the set difference operator, we have that: (1) $\mathcal{K} \setminus \mathcal{K}^*$ is the set of facts that are false; and (2) $\mathcal{K}^* \setminus \mathcal{K}$ is the set of true facts not included in the knowledge base. Formally, we say that a knowledge base $\mathcal{K}$ is *complete* (with respect to $\mathcal{K}^*$) iff $\mathcal{K}^* \setminus \mathcal{K} = \emptyset$. Because it is practically impossible to construct such a knowledge base $\mathcal{K} = \mathcal{K}^*$, even for a very small domain of knowledge, completeness is usually bound to a particular query (Razniewski, Suchanek, and Nutt, 2016). We can thus think that a particular query is the scope mentioned by Wang's definition of completeness in R. Y. Wang and Strong (1996).

Empirical studies on knowledge bases have found that most of them are actually incomplete and with a large number of missing triples. For instance, in Suchanek, Gross-Amblard, and Abiteboul (2011) and Min, Grishman, Wan, et al. (2013) authors found that between 69–99% of instances in popular knowledge bases lack at least one relation that other entities of the same type have. They found that DBpedia v9 contains only 6 out of 35 Dijkstra Price winners, and in YAGO the average number of children per person is 0.02. In Wikidata (Vrandecic, 2012) (as of 2016) it is known the father for only 2% of all people. Similarly, Google reports that 71% of people in Freebase have unknown place of birth and 75% have unknown nationality (X. Dong, Gabrilovich, Heitz, et al., 2014).

A search for completeness of knowledge bases is hard labour in practice due to the OWA that says that facts that are not in the knowledge base are unknown and may or not be true (see Section 2.1.3). Several approaches adopt the more flexible PCWA and aim for the completion of knowledge bases (graphs) by targeting missing links between the already observed entities and relations. This problem is known as the *Link Prediction problem* and we will review some methods in Chapter 3 that learn link patterns to predict relationships between entities. Link prediction is one of the research problems that has attracted increasing attention largely due to its application in Natural Language Processing (NLP) tasks (Nickel, K. Murphy, Tresp, et al., 2016).

Following Lajus and Suchanek (2018), we can define the three interpretations of knowledge bases that we have reviewed in Section 2.1.3. Similarly, the same interpretations are applicable and extensible to knowledge graphs. OWA states that nothing follows from the absence of a fact in the knowledge base, i.e., the absence of evidence is not evidence of absence. Although most knowledge bases are interpreted under OWA, this assumption does not help when trying to check consistency, mine or discover patterns from knowledge bases. Therefore, often the CWA is adopted assuming that the knowledge base is complete. The CWA can be formally expressed as follows:

$$\forall s, r, o : (s, r, o) \notin \mathcal{K} \implies (s, r, o) \notin \mathcal{K}^*. \tag{2.1}$$

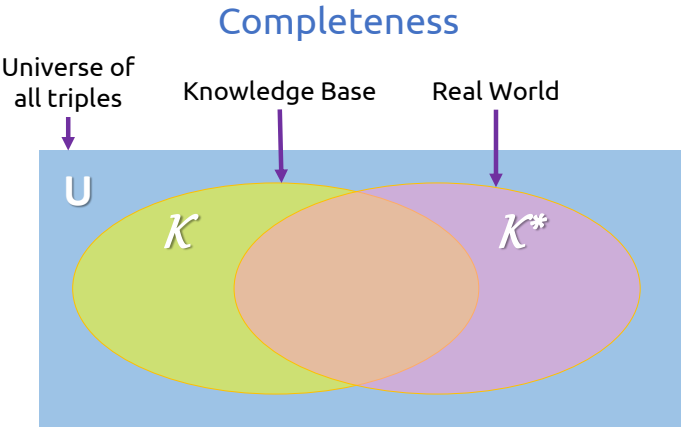


Figure 2.3: Venn diagram explaining the completeness of a knowledge base.

Because the CWA is a strong assumption in practice, the PCWA was proposed to deal with this. The PCWA can be formally expressed as:

$$\forall s, r, o, o' : (s, r, o) \in \mathcal{K} \wedge (s, r, o') \notin \mathcal{K} \implies (s, r, o') \notin \mathcal{K}^*. \tag{2.2}$$

While OWA semantics is appropriate when using OWL, it is not suitable for completion. On the other hand, the more restrictive CWA does not allow the addition of new facts that can yield inconsistency of the knowledge. We argue that only under more flexible assumptions such as PCWA it is possible to aim for the completion of knowledge bases. This is supported by good experimental results in several cases (L. A. Galárraga, Teflioudi, Hose, et al., 2013; X. Dong, Gabrilovich, Heitz, et al., 2014). Later in Chapter 3, we will discuss the use of PCWA for the generation of negative examples to train knowledge graph embeddings models that learn scoring functions to differentiate true vs false triples in a knowledge graph. Knowledge graph embedding models are popular solutions to tackle the link prediction and knowledge graph completion tasks.

Next, we give an overview and present the fundamentals of machine learning and deep learning frameworks used to discover missing links and complete knowledge graphs.

2.4 Machine learning: A brief overview

As mentioned earlier, machine learning has been used by the most popular approaches tackling the knowledge graph completion task. In this section, we provide a very brief overview on general machine learning and the types of machine learning algorithms that relate to our contributions.

Machine Learning is the science (and art) of programming computers so they can *learn from data*. A general machine learning process consist of three core variables: (i) a task T; (ii) a performance measure P; and (iii) an experience E. A formal definition is as follows:

Definition 2.5

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E. (Mitchell, 1997)

To illustrate this definition we follow our Iris flowers example, where determining the specie of Iris flowers (i.e., setosa, versicolor, virginica) is the task T, a neural network with normalised squared error loss can be the performance measure P, and a set of feature vectors for a sample of flowers the experience E. Figure 2.4 shows a general framework for machine learning, where examples from an input space (experience) are mapped into points of an output space (task) of labels. A prediction is then evaluated w.r.t. the ground truth and the loss function (performance measure) is computed using both values. Then the learning process consist of finding the best hypothesis (also known as model) that optimises the performance measure and reduces the prediction error a.k.a. loss or cost.

Formally, we define a *domain* $\mathcal{D}$ that consists of two components: (1) a feature space $\mathcal{X}$; and (2) a marginal probability distribution $\Pr(X)$, where $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \in \mathcal{X}$. Given a specific domain $\mathcal{D} = \{\mathcal{X}, \Pr(X)\}$, a *task*, denoted as $\mathcal{T} = \{\mathcal{Y}, f(\cdot, \cdot)\}$, consists of two components: (1) a label space $\mathcal{Y}$; and (2) an objective function $f(\cdot, \cdot)$, which is not observed but can be learnt from the training data. In supervised machine learning, training data consists of pairs $\{\boldsymbol{x}_i, y_i\}$, where $\boldsymbol{x}_i \in X$ and $y_i \in \mathcal{Y}$. From a probabilistic point of view, $f(\boldsymbol{x}, y)$ can be written as the conditional probability $\Pr(y \mid \boldsymbol{x})$.

Machine learning systems can be classified based on different categories such as level of human supervision, incremental or on-the-fly learning, and instance-based vs. model-based learning. In the following, we present a common classification based on the amount of supervision used during training. For an overview of other classifications, we refer the reader to Mitchell (1997), Chapelle, Schölkopf, and Zien (2006), K. P. Murphy (2012), and Goodfellow, Bengio, and Courville (2016).

Supervised machine learning. In *supervised learning*, the training data consist of a set of pairs $(\boldsymbol{x}_i, y_i)$, where $\boldsymbol{x}_i$ is an input object (typically a vector called *feature vector*) and y_i is the associated *target* or *label* (Mitchell, 1997). A typical supervised learning task is *classification* that consists of a learning algorithm that seeks a function $h \colon X \to \mathcal{Y}$, where X is the input space and $\mathcal{Y}$ is the output (or label) space. Such function h in an element of some space of possible functions $\mathcal{H}$, which is usually called *hypothesis space*. Another task is *regression* which consists in predicting a target numeric value from the input example (e.g., predicting the price of a house in a given area). In classification, the label y_i belongs to one of a finite number of classes, whilst in regression, y_i is a continuous number. Formally, h can be represented using a scoring function $f \colon X \times \mathcal{Y} \to \mathbb{R}$ such that h returns a $\hat{y}$ value that gives the highest score: $h(\boldsymbol{x}) = \arg\max_y f(\boldsymbol{x}, y)$. We denote by $\mathcal{F}$ the space of scoring functions. A

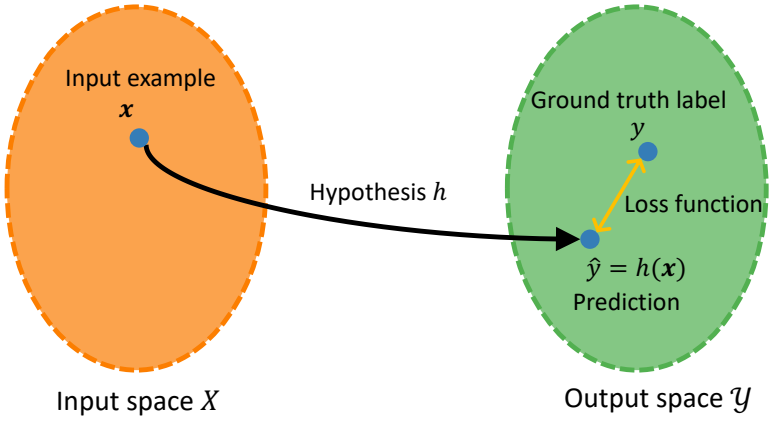


Figure 2.4: Machine learning general framework (adapted from T. Liu (2011)).

classical example of this type of learning is the spam identification in emails.

Unsupervised machine learning. In *unsupervised learning*, the task is to infer a function that describes the structure of "unlabelled" data, i.e., data that have not been classified or categorised (Mitchell, 1997). The lack of labels makes this type of task more challenging to evaluate than the supervised one. Algorithms for clustering, dimensionality reduction, anomaly detection, and association rule learning belong to this machine learning category. An example of this type of learning is the market basket analysis, where rules can be extracted from analysing what supermarket shoppers buy each time and identifying patterns of products bough together at a given time. While supervised learning intends to infer a conditional probability $\Pr(y \mid \boldsymbol{x})$, unsupervised learning intends to infer an a priori probability distribution $\Pr(X = \boldsymbol{x})$.[11]

Semi-supervised machine learning. *Semi-supervised learning* is a class of supervised learning, where the training data contains labelled and unlabelled data (Chapelle, Schölkopf, and Zien, 2006). Usually, the amount of labelled data is smaller than the amount of unlabelled data, mainly because the former is usually more expensive to obtain. A semi-supervised algorithm is given a set of m i.i.d. examples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in X$ with their corresponding labels $y_1, \ldots, y_m \in Y$ that compose the labelled part of data, plus additional u unlabelled examples $\boldsymbol{x}_{m+1}, \ldots, \boldsymbol{x}_{m+u} \in X$. The goal of semi-supervised learning is to make use of the combined information to obtain a better performance measure than what could be obtained either by discarding the unlabelled data (falling into supervised learning) or by discarding the labels (falling into unsupervised learning). An example of this learning is Google Photos[12], where faces of people are recognised and clustered in an unsupervised manner, but the user could tag these people so the system can then be able to name everyone in every photo.

Reinforcement learning. *Reinforcement learning* is a different type of learning inspired by behaviourist psychology (Goodfellow, Bengio, and Courville, 2016). The learning system,

[11] $\Pr(X = \boldsymbol{x})$ indicates the probability of X taking on the value $\boldsymbol{x}$.
[12] A photo sharing and storage announced in May 2015 by Google (https://photos.google.com/).

also called an *agent*, selects and performs actions in an observed environment to get *rewards* in return (or *penalties* as a form of negative rewards). The goal for the agent is to learn by itself the best strategy, called a *policy*, that maximises the reward over time. A policy defines the actions that the agent should follow in a given situation. An example of this learning is Google DeepMind's AlphaGo program that became famous world wide after beating the world champion Lee Sedol at the game of *Go* in 2016.[13] For that a software learnt a policy by analysing millions of games and then playing many times against itself.

What machine learning offers is to find rules that are *probably* correct about *most* members of the set they concern (Goodfellow, Bengio, and Courville, 2016). These rules are not entirely certain (as in logics) and do not hold for every member of a set. The **no free lunch theorem** (Wolpert, 1996; Wolpert and Macready, 1997) states that every classification algorithm has the same error rate when classifying previously unknown points. Therefore, there is no machine learning algorithm that is universally better than any other. When applying machine learning, our goal is to understand what kind of distributions or relations are most relevant in the data and *bias* the algorithm towards that distributions to design algorithms that perform well on a specific task.

2.4.1 Regularisation

Based on the no free lunch theorem for supervised learning (Wolpert, 1996), when designing our machine learning algorithms we must use bias or encode a set of preferences into the learning algorithm to obtain a better performance on the task at hands. There are many ways to express such preferences for different solutions in machine learning. One way of setting preferences in an algorithm is to restrict its hypothesis space and the functions allowed in there. Another solution can involve adding preferences to one solution over another in the hypothesis space. Approaches to express such preferences are known as *regularisation* and can be explicit or implicit modifications we make to a learning algorithm. More formally, a regularisation term is a function $R(\Theta)$ applied to the model parameters Θ, which can be added to a loss function $J(\Theta)$ (that encodes a performance metric P) in a learning algorithm:

$$J(\Theta) = P + \lambda R(\Theta),$$

where P is a performance metric such as mean squared error (for regression) or binary loss (for classification), λ is the factor that controls the strength of the regularisation. When $\lambda = 0$ means that no preference is added, and the larger λ the stronger the preference we impose into the learning algorithm.

The no free lunch theorem also applies to regularisation, i.e., there is no best form of regularisation as there is no best machine learning algorithm (Goodfellow, Bengio, and Courville, 2016). In Chapter 7, we will design a regularisation term for incorporating cardinality restrictions to representation learning algorithms for knowledge graphs.

[13] AlphaGo versus Lee Sedol, Wikipedia (https://en.wikipedia.org/wiki/AlphaGo_versus_Lee_Sedol).

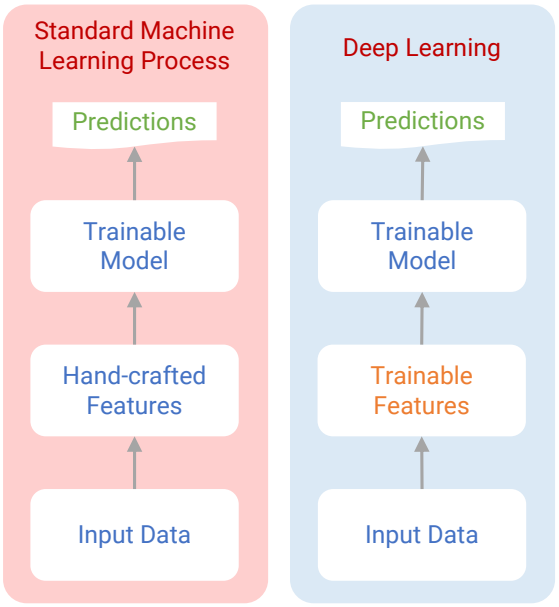


Figure 2.5: Deep learning general framework versus traditional machine learning.

2.5 Deep learning: A brief overview

Deep learning (a.k.a. hierarchical learning) is a particular category of machine learning based on artificial neural networks (e.g., Ivakhnenko (1971), Bengio, Cardin, Mori, et al. (1988), LeCun, Boser, Denker, et al. (1989), LeCun, Bengio, and G. E. Hinton (2015), Schmidhuber (2015), and Goodfellow, Bengio, and Courville (2016), among others). Artificial neural networks, or connectionist systems, are computing systems inspired by the biological neural networks in the human brain. Deep neural networks are multi-layered models that can learn representations of data with multiple levels of abstraction, in other words, a hierarchy of features. By learning representations deep neural networks dramatically reduce the need for feature engineering and human intervention. They have shown to deliver state-of-the-art results in many challenging tasks for humans such as speech recognition, natural language processing, visual object recognition, and in other domains such as drug discovery. Figure 2.5 shows a parallel comparison between deep learning and traditional machine learning.

Artificial neural networks (more precisely, feedforward neural networks) are the building block of deep learning, where neural networks are composed of several layers. A neural network is used to approximate some target function f^*. In supervised learning setting, a classifier $y = f^*(\boldsymbol{x})$ maps an input $\boldsymbol{x}$ to a category y, in deep learning a neural network defines a mapping $\boldsymbol{y} = f(\boldsymbol{x}; \Theta)$ that can also learn the values of the parameters Θ that yield the best function approximation or model. (Note that other methods for feature (representation) learning exist in literature. See Bengio, Courville, and Vincent (2012) for a review.) Thus a learning algorithm will aim at adapting the initial parameters Θ to make f as similar as possible to f^*. The parameters Θ is what we refer to as trainable features in Figure 2.5.

The network characteristic comes from the fact that many different functions are composed in the design of a neural network, usually represented by a directed acyclic graph (see Figure 2.6). For example, a composition of functions f and g in a chain forms $g(f(\boldsymbol{x}))$, where f is called the *first layer* of the network, and g is called the *second layer*. The chain structure is the most commonly used design structure, but several works have recently proposed techniques to automatically discover the design of neural networks (Elsken, Metzen, and Hutter, 2019). Note that many more functions could be stacked following the same logic, giving a "depth" component to the network and the name to *deep learning* (Krizhevsky, Sutskever, and G. E. Hinton, 2012).

A classical neural network is known as the multilayer percentron (MLP) that is composed of at least three layers of nodes, namely, an input layer, a hidden layer, and an output layer. Figure 2.6 shows a one-layer MLP model with fully connected layers, where the nodes of a layer are connected to *all* the nodes of the next layer. Let $\boldsymbol{x} = \{x_i\}_{i=1}^{d}$ be an input example of dimension d with label $y \approx f^*(\boldsymbol{x})$, $f(\cdot)$ an activation function in a hidden layer, and $g(\cdot)$ an activation function in the output layer. We can specify a one-layer MLP model as:

$$
\begin{aligned}
h_j &= f(\sum_{i=1}^{N} w_{ij} x_i + b_j) \\
y_k &= g(\sum_{j=1}^{M} w_{jk} h_j + b_k),
\end{aligned}
\tag{2.3}
$$

where w_{ij} are the weight of the hidden layer, w_{jk} the weights of the output layer, and b_j and b_k the corresponding bias. The weights w_{ij} are given to the connections between the i-th input node and the j-th hidden layer node (h_j). Similarly, the weights w_{jk} are for the connections between the j-th hidden layer node and the k-th output layer node. We can also use a matrix notation to specify the model:

$$
\begin{aligned}
\boldsymbol{h} &= f(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1) \\
\boldsymbol{y} &= g(\boldsymbol{W}_2 \boldsymbol{h} + \boldsymbol{b}_2),
\end{aligned}
\tag{2.4}
$$

where each layer has its own weight matrix $\boldsymbol{W}$ and bias vector $\boldsymbol{b}$. Activation functions are typically chosen to be element-wise functions such as sofmax, sigmoid, and the rectified linear unit (ReLU) (Nair and G. E. Hinton, 2010). The flow of information and computations through the network, e.g., from obtaining $\boldsymbol{h}$ and passing it to the subsequent layers, is called *forward propagation*.

Although convexity is a desirable property for loss functions, most of the time in artificial neural networks they are non-convex due to the use of non-linearity activation functions in these models. To address this issue, gradient-based optimisers are used to train neural networks in iterations taking batches of data in each iteration. Gradient descent is a first-order iterative optimisation algorithm for finding the minimum of a (loss) function. To find a local minimum, the algorithm takes steps proportional to the negative of the gra-

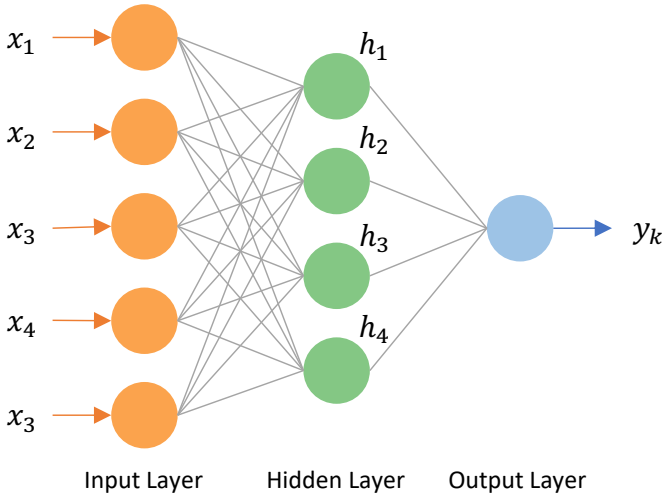


Figure 2.6: Neural network architecture with one hidden layer.

dient (or approximate gradient) of the function at the current point (Goodfellow, Bengio, and Courville, 2016). Thus, it travels down the slope of the function in steps until it reaches the lowest point. However, the number of iterations in gradient descent depends on the number of samples and features, thus, it may become very slow to compute. A solution is to replaces the actual gradient (calculated over the entire data) by an estimate (calculated from a randomly selected subset of the data). This is known as *stochastic gradient descent* (SGD) (Robbins and Monro, 1951). SGD has been successfully applied to train artificial neural networks and together with the back-propagation algorithm (Rumelhart, G. E. Hinton, and Williams, 1986) form one of the most popular gradient-based optimiser in literature (Goodfellow, Bengio, and Courville, 2016). Back-propagation provides an efficient method for computing gradients to SGD. Unlike convex optimisation, the use of stochastic gradient descent to non-convex loss functions does not have guaranties of convergence to a global minimum, and the output is sensitive to the values of the initial parameters (Glorot and Bengio, 2010).

In this dissertation, we mostly focus on solutions using supervised machine learning to learn functions from training data generated from knowledge graphs. Next, we describe two common learning frameworks for supervised learning based on the output space, namely, learning to rank and multi-label learning, and the performance metrics we will use for each one.

2.6 Learning problems

Even when a problem is considered as a supervised learning problem one could be stating it in different ways depending on the output space. For instance, one could learn different

objects appearing in an image or assigning a relevant title to an image. The former we say is a multi-label problem (multiple objects can be found in a given image), whilst the latter is known as a ranking problem (only the title with highest probability is selected). Here, we refer to these different ways as *learning problems*. Specifically, we focus on two of these frameworks, namely, learning to rank and multi-label learning, which are commonly used in different domains and problems.

2.6.1 Learning to rank

Given the large amount of data available in some environments such as the Web, several approaches have been proposed in Information Retrieval to create ranking models for learning what content is relevant to users (T. Liu, 2011). Recently, given the availability of potential training data, it has become possible to leverage machine learning methods to build effective ranking models. Methods that learn how to combine features for ranking by means of discriminative learning are called *learning-to-rank* methods. Learning to rank methods are *feature based*, i.e., they require a function that builds vector representation of the examples; and have a *discriminative training*, meaning that the learning process is based on training data and they have their own input, output, hypothesis space, and loss function (T. Liu, 2011).

In classification tasks such as the knowledge graph completion, it is common to choose a threshold θ to separate true (positive) values from false (negative) ones.[14] However, sparseness on the data might cause a strong bias pushing most values towards zero, which difficult the selection of a good value for θ. In cases like these, and whenever it is not necessary to determine the truth value of an element, it is recommended to use the likelihood returned by the model and consider a better alternative to rank the outputs. That is how learning to rank becomes a relevant part of models tackling knowledge graph completion tasks.

Formally, let $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ be a set of n elements in $\mathbb{R}^d$, and let $l\colon \mathcal{X} \to \mathbb{N}$ be a labelling function, where $l(\boldsymbol{x})$ is the label of object $\boldsymbol{x}$. For instance, $l\colon \mathcal{X} \to \{0, 1\}$ would be a binary labelling where 1 (true) and 0 (false) are the labels, or an arbitrary integer in the case of multi-label problems. Let $f\colon X \to \mathbb{R}$ be a scoring function, where f aims to score a set of examples $\mathcal{X}$ such that positive examples are scored higher than negative ones. Formally, $\forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X} : \; l(\boldsymbol{x}_i) > l(\boldsymbol{x}_j) \Rightarrow f(\boldsymbol{x}_i) > f(\boldsymbol{x}_j)$. We also use $rank_f(\boldsymbol{x}_i)$ to denote the rank position of object $\boldsymbol{x}_i$ according to the scoring function f, i.e., the position of score $f(\boldsymbol{x}_i)$ in a descending order of all scores $f(\boldsymbol{x})\ \forall \boldsymbol{x} \in \mathcal{X}$.

Usually, for the link prediction task a pairwise ranking algorithm is used as the loss function. More specifically, the so-called *hinge loss* (Rosasco, Vito, Caponnetto, et al., 2004) is used to subtract positive from negative score values in order to penalise cases where positive triples are given scores lower than negative triples. The hinge loss is a convex function used for training classifiers. For a binary classifier with output $t \in \{-1, +1\}$, and

[14] Usually a $\theta = 0.5$ is selected to separate positives from negatives in classification problems.

classifier prediction y, the hinge loss of y is defined as:

$$\ell(y) = \max(0, 1 - t \cdot y).$$

Note that y is the probability returned by the classifier and not the predicted class label.

2.6.2 Multi-label learning

When the output space for an object $\boldsymbol{x}$ is a set of labels, we say that the problem is a *multi-label learning problem*. In the following, we formalise the learning framework with q-labels as in M.-L. Zhang and Z.-H. Zhou (2006) and Zha, Mei, J. Wang, et al. (2009). Let $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ be the input space of m different data points in $\mathbb{R}^d$, and let $\mathcal{Y} = \{y_1, y_2, \ldots, y_q\}$ be the finite set of labels. Given a training set $\mathcal{D} = \{(\boldsymbol{x}_i, Y_i)\}_{i=1}^m$, where $\boldsymbol{x}_i \in \mathcal{X}$ is a d-dimensional feature vector $[x_{i1}, x_{i2}, \ldots, x_{id}]^\top$ and $Y_i \in 2^{\mathcal{Y}}$ is a vector of labels associated with $\boldsymbol{x}_i$, the goal of the learning system is to output a multi-label classifier $h : \mathcal{X} \to 2^{\mathcal{Y}}$ which optimises some specific evaluation metric. In most cases, however, the learning system will not output a multi-label classifier, but instead will produce a real-valued function (regressor) of the form $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, where $f(\boldsymbol{x}, y)$ can be regarded as the confidence of $y \in \mathcal{Y}$ being the proper label of $\boldsymbol{x}$. It is expected that for a given instance $\boldsymbol{x}$ and its associated label set Y, a successful learning system will tend to output larger values for *relevant* labels $y_i \in Y$, and smaller values for *irrelevant* labels $y_k \notin Y$, i.e., $f(\boldsymbol{x}, y_i) > f(\boldsymbol{x}, y_k)$ for any $y_i \in Y$ and $y_k \notin Y$. In other words, the model should consistently be more "confident" about true positives (actual labels) than about false positives.

Intuitively, the regressor $f(\cdot, \cdot)$ can be transformed into a ranking function $rank_f(\cdot, \cdot)$, which maps the outputs of $f(\boldsymbol{x}, y)$ for any $y \in \mathcal{Y}$ to $\{y_1, y_2, \ldots, y_q\}$ such that if $f(\boldsymbol{x}, y_i) > f(\boldsymbol{x}, y_k)$ then $rank_f(\boldsymbol{x}, y_i) < rank_f(\boldsymbol{x}, y_k)$. The ranking function can naturally be used for instance for selecting top-k predictions for any given example, which can be very useful in cases where only limited numbers of prediction candidates can be further analysed by the experts.

2.6.3 Evaluation metrics

Evaluation metrics or performance measures help us to quantitatively measure the behaviour of a model. They evaluate how well a model performs in different dimensions over a previously unseen test set. In this section, we present the most common evaluation metrics for the two learning frameworks we reviewed.

2.6.3.1 Learning to rank

To evaluate the performance of learning to rank models, we can use most of the metrics available in Information Retrieval. The following is a list of evaluation metrics from (Man-

ning, Raghavan, and Schütze, 2008; M. Zhang and Z. Zhou, 2014), which are frequently used to evaluate learning-to-rank models over a test set $\mathcal{S}$ with p examples.

Mean rank, MR: Evaluates the average of predicted ranks. MR is sensitive to outliers.

$$MR = \frac{1}{p}\sum_{i=1}^{p} rank_f(\boldsymbol{x}_i).$$

Mean reciprocal rank, MRR: Evaluates the average of reciprocal ranks, which is the inverse position of the first relevant answer, and is therefore well-suited for applications where only the first result matters. MRR is less sensitive to outliers than MR.

$$MRR = \frac{1}{p}\sum_{i=1}^{p} \frac{1}{rank_f(\boldsymbol{x}_i)},$$

where $\boldsymbol{x}_i$ is the highest ranked relevant item for a query q_i.

Hits@k: Measures the number of elements retrieved among the k elements with the highest score. Since this metric is per example, we report its average dividing by the number of examples. We usually extract this score for $k \in \{1, 3, 5, 10\}$.

$$Hits@k = \sum_{i=1}^{p} l(\boldsymbol{x}_i),$$

where l is the labelling function that returns 1 when $\boldsymbol{x}_i$ is positive and 0 otherwise.

P@k: P@k stands for the precision at k, i.e. precision computed only among top-k ranking labels per example. We usually extract this score for $k \in \{3, 5, 10\}$.

$$P@k = \frac{|\{\text{relevant results in the top } k \text{ positions}\}|}{k}.$$

2.6.3.2 Multi-label learning

To evaluate the performance of multi-label learning models, we use specific evaluation metrics that are different from the ones used in traditional supervised learning (M.-L. Zhang and Z.-H. Zhou, 2014). Let $\mathcal{S}$ be multi-label test set comprising p multi-label examples $\{(\boldsymbol{x}_i, Y_i)\}_{i=1}^{p}$, where $\boldsymbol{x}_i \in \mathcal{X}$, $Y_i \in \mathcal{Y} = \{0, 1\}^q$, $\mathcal{L}$ is a label set, and $|\mathcal{L}| = q$, i.e., the total number of labels. Here, $f(\cdot, \cdot)$ represents the multi-label regressor and $f(\boldsymbol{x}, Y) = \{0, 1\}^q$ is the set of label memberships predicted by f for the example $\boldsymbol{x}$. We compute the following example-based ranking metrics for evaluating the results of the predictions over $\mathcal{S}$ (Tsoumakas, Katakis, and Vlahavas, 2010; M.-L. Zhang and Z.-H. Zhou, 2014):

One error: Evaluates the fraction of examples whose top-ranked label is not in the set of relevant labels of the instance.

$$One - Error(f) = \frac{1}{p}\sum_{i=1}^{p} I(\arg\max_{y\in\mathcal{Y}} f(\boldsymbol{x}_i, y) \notin Y_i),$$

where, I is an indicator function and $f(\boldsymbol{x}_i, y)$ is the score of label y for an instance $\boldsymbol{x}_i$.

Coverage error: Evaluates how far we need, on average, to move down the ranked list of labels in order to cover all the relevant labels of the instance.

$$Cov - Error(f) = \frac{1}{p}\sum_{i=1}^{p} \max_{y\in Y_i} rank_f(\boldsymbol{x}_i, y) - 1.$$

Ranking loss: Evaluates the fraction of reversely ordered label pairs, i.e. an irrelevant label is ranked higher than a relevant label.

$$R - Loss(f) = \frac{1}{p}\sum_{i=1}^{p} \frac{1}{|Y_i||\overline{Y_i}|}\left|\left\{(y', y'') : rank_f(\boldsymbol{x}_i, y') > rank_f(\boldsymbol{x}_i,\ y''), (y', y'') \in Y_i \times \overline{Y_i}\right\}\right|,$$

where $\overline{Y_i}$ is the complementary set of Y_i in $\mathcal{Y}$.

Average precision: Evaluates the average fraction of relevant labels ranked higher than a particular label $y \in Y_i$ which actually are in Y_i.

$$AP(f) = \frac{1}{p}\sum_{i=1}^{p} \frac{1}{|Y_i|}\sum_{y\in Yi} \frac{\left|\left\{y' : rank_f(\boldsymbol{x}_i, y') \leq rank_f(\boldsymbol{x}_i, y),\ y' \in Y_i\right\}\right|}{rank_f(\boldsymbol{x}_i, y)}$$

Area under the precision-recall curve (AUC-PR): PR curves summarise the trade-off between the true positive rate (TPR) and the positive predictive value (PPV) for a predictive model using different probability thresholds. The values of AUC-PR range from 0 to 1, and the higher the better.

$$Prec(t) = PPV(t) = \frac{TP}{TP + FP}$$
$$Rec(t) = TPR(t) = \frac{TP}{TP + FN}$$
$$AUC - PR = \int_{x=0}^{1} Prec(Rec^{-1}(x))\, dx,$$

where x is the variable that defines a decision threshold used as cut-off for positives and negatives in the functions $Prec(\cdot)$ and $Rec(\cdot)$.

Area under the receiver operating characteristic curve (AUC-ROC):

ROC curves summarise the trade-off between the true positive rate (TPR) and false

positive rate (FPR) for a predictive model using different probability thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives. The values of AUC-ROC range from 0 to 1, and the higher the better.

AUC-ROC stands for "area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the ROC curve from (0,0) to (1,1).

$$TPR(t) = \frac{TP}{TP + FN}$$
$$FPR(t) = \frac{FP}{FP + TN}$$
$$AUC - ROC = \int_{x=0}^{1} TPR(FPR^{-1}(x))\, dx,$$

where x is the variable that defines a decision threshold used as cut-off for positives and negatives in the functions $TPR(\cdot)$ and $FPR(\cdot)$.

For *One-Error*, *Coverage* and *Ranking Loss*, the smaller the metric value the better the system's performance, with optimal value of $\frac{1}{m}\sum_{i=1}^{m}|Y_i| - 1$ for *Coverage* and 0 for *One-Error* and *Ranking Loss*. For *Average Precision* metric, the larger the metric value the better the system's performance, with optimal value of 1.

When dealing with skewed data—as in most of this thesis—AUC-PR provides a more informative picture of the model's performance than AUC-ROC (Davis and Goadrich, 2006), because it ignores true negatives (TN). Both AUC-PR and AUC-ROC are usually computed using the trapezoidal rule.[15]

2.7 Summary

In this chapter, we have introduced the fundamentals required to understand the contributions of this thesis. We have recapitulated the background and definitions associated with knowledge bases, knowledge graphs, and overviewed relevant machine learning approaches that focus on the completion task of these resources. In Section 2.1, we reviewed the concept of knowledge bases and focus on their interpretation in the Semantic Web community. Two of the assumptions made for Semantic knowledge bases are the non-UNA and OWA. These properties seem highly desirable for open environments like the Web, but they pose a number challenges for many tasks that require a closed view of the data.

While knowledge bases have long been used in formal logic and the Semantic Web, the concept of knowledge graphs (Section 2.2) has been gaining momentum and is beginning to replace the former as a wildcard for any graph-structured knowledge repository. However,

[15] Trapezoidal rule is a technique to approximate an integral (https://en.wikipedia.org/wiki/Trapezoidal_rule).

despite the recent interchangeable use of the terms, there are many gaps on the approaches trying to complete their content. Completeness (Section 2.3) is presented as one of the tasks with growing interest from both academia and industry due to its practical applications. We have discussed how the consistency and completeness dimensions of a knowledge graph are tightly correlated. These two quality dimensions have proven to be very challenging given the size and flexibility allowed by graph-structured data—and they are not easy to handle in their most general case. Given the centrality of these dimensions for the area of knowledge graph mining, we will discuss them separately in more details in Chapter 3.

We have also introduced relevant concepts related to machine learning (Section 2.4) and deep learning (Section 2.5) that will help us later to present the contributions of this thesis. Finally, the learning tasks and a set of evaluation metrics is presented in Section 2.6), which will help us to assess the performance of a model in a given task.

CHAPTER 3

Knowledge Graph Mining

Contents

Knowledge graphs are a rich source of information that can be exploited in many different scenarios. To exploit them, traditionally, graph structural features have been used for tasks such as label propagation, link prediction, and recommendation systems, among others. Recently, representation learning techniques have become popular given their ability to learn functions and generic representations for knowledge graphs. In this chapter, we provide an overview of knowledge graph mining focusing on two tasks: (a) mining of structural patterns, and (b) link prediction as a form of knowledge graph completion.

Producers and consumers of knowledge graphs have different conceptualisations that are hard to share with each other; however, both require means for knowing what is the structure of data to perform mining. Knowledge graphs differ significantly from traditional

relational databases, since they are modelled as schema-free graphs. Given the need for structure, in task (a), we study the languages that can be used to define the structure of knowledge graphs and several structural patterns. We review the most popular proposals for constraint languages that allow to define schemas for knowledge graphs. And motivated by the idea that structure is closely tied to the completion of knowledge graphs, in task (b), we study the so-called *knowledge graph embedding models*. Knowledge graph embedding models capture some of the structural patterns of links that can be used in the link prediction task, which has become one of the most relevant research areas in statistical relational learning (Getoor and Taskar, 2007).

3.1 Schema in knowledge graphs

In database management, a *database schema* is the logical organisation of all or part of a relational database (Codd, 1970; Abiteboul, Hull, and Vianu, 1995). A database schema indicates the structure of data, i.e., how the entities (tables or relations) relate to one another and what are the fields (properties) included on each entity. All this is described in a data definition language such as the structured query language (SQL). A database schema can be represented in a visual way or as a set of formulas called integrity constraints imposed on a database. Schemas are created in a process known as *data modelling* and have as design goal helping programmers and data engineers to interact with the database. The data modelling process comes before any population of the database—since the schema is required to add any instance data. Schema diagrams, also known as entity-relationship (ER) diagrams (P. P. Chen, 1976), can help to visualise the entity types and relationships that can exist between instances of those entity types. Schema information has a number of benefits for data management, such as helping indexing schemes, query optimisation techniques, access control schemes, validation, among others (Abiteboul, Hull, and Vianu, 1995).

In Semantic Web knowledge bases (and knowledge graphs), the term schema is usually understood as the set of classes and relations. The RDF Schema (RDFS) (Brickley, R. Guha, and McBride, 2014) standard is an extension of RDF that provides a data-modelling vocabulary for RDF data. It allows to describe groups of related resources and the relationships between them. For instance, we could define the relation `schema:author`, whose subjects must be a `schema:CreativeWork` (the domain), and objects must be a `schema:Person` (the range). RDFS allows to describe a reasoning like: "If *Victor Vianu* is an author of the book *Foundations of Databases*, then Vianu is a Person "—considering that a book is a type of creative work. In fact, RDFS is not much of a schema as the one in databases, since it does not allow the definition of data constraints. RDFS is property oriented, which differs from the classical object oriented modelling where the class `schema:CreativeWork` would be defined first with all its attributes. Conversely, using RDFS classes can be extended and new properties can be added without the need to re-define the original class.

A richer representation of knowledge can be achieved with the definition of a so-called

ontology in Computer Science.[1] This should not be confused with the term "Ontology" (with uppercase initial) that refers to a branch of philosophy which deals with the *nature* and *structure* of "reality."[2] (We redirect the reader to Guarino, Oberle, and Staab (2009) for a deeper introduction to the topic of ontologies.) In a nutshell, an ontology formally models the structure of a system stating the relevant concepts (entities) and relations that can formally exist for an AI system—what exists is what can be represented (Gruber, 1995). The goal of ontologies is to describe a specific domain shared by a community of users and do not necessarily represent the structure of the instance data. The most prevalent and common definition for ontology is given by Studer, Benjamins, and Fensel (1998): "An ontology is a formal, explicit specification of a shared conceptualisation."

The Web Ontology Language (OWL) (Motik, Peter F. Patel-Schneider, and Parsia, 2012) is the standard language for defining ontologies in the Semantic Web, typically defined as a taxonomy and set of inference rules. OWL extends the expressibility of RDFS and allows to describe the required information for instances and classes. For instance, it could detect typing errors such as `"3.14159265359"^^xsd:integer`, where the datatype `xsd:integer` is wrongly used instead of `xsd:double` to define the number π. OWL allows to express schemas that should be interpreted under the standard first-order semantics and not as checks (Motik, Horrocks, and Sattler, 2009). As a World Wide Web language, OWL's formal semantics—based on Description Logic—uses the open world assumption (OWA) (cf. Section 2.1.3) and does not use the unique name assumption (nUNA). Meaning that the conclusions that one can draw from such ontologies differ from the ones that the users intuitively expect. Assuming the OWA, it will not give any conclusion based on the absence of data: *The absence of evidence is not evidence of absence*. For instance, a hospital ontology could specify that each person has a blood type, but a person could be added without a blood type to the knowledge graph and this will not raise any error. Under OWL's semantics, the data can be incomplete, but incomplete is different from inconsistent. This is one of the central ideas in our work that will be studied and evaluated in the chapters to come.

Recently, an alternative to describe and validate the structure of a knowledge graph has been proposed by the W3C standard Shapes Constraint Language (SHACL) (Knublauch and Kontokostas, 2017). SHACL allows to define the constraints that an RDF knowledge graph (*data graph*) should follow in a so-called *shapes graph*. A SHACL shapes graph describes the structure and conditions that the instance data (or *data graph* in SHACL terminology) satisfy. These shapes can be used for validating instance data, building user interfaces, and data integration, among others (Labra Gayo, E. Prud'hommeaux, Boneva, et al., 2018). Likewise, Shape Expressions (ShEx) (E. Prud'hommeaux, Boneva, Labra Gayo, et al., 2018) is a grammar-based language coming from a W3C community group that similarly to SHACL allows the definition of shapes that RDF knowledge graphs should satisfy. We

[1] Herein, we consider the term ontology for defining schema data and different from instance data. And we disagree with the use of ontology as synonym of knowledge graph.

[2] In philosophy, the first studies of Ontology are attributed to Aristotles for his book Metaphysics (ca. 985) and Parmenides.

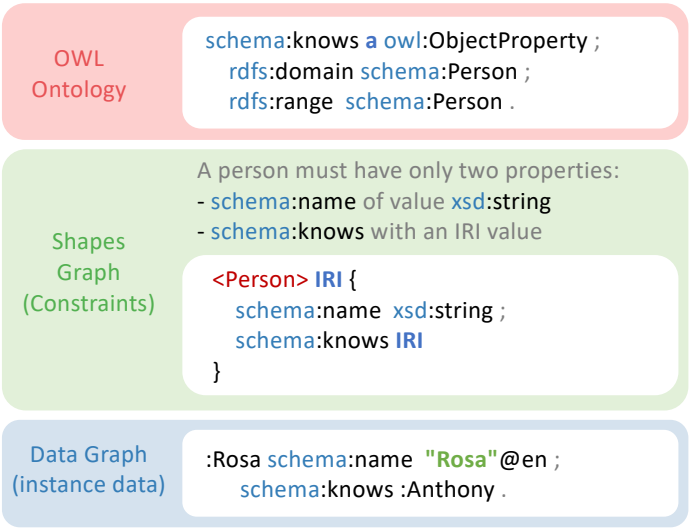


Figure 3.1: Comparison between ontology and shapes graph definition (adapted from Labra Gayo, E. Prud'hommeaux, Boneva, et al. (2018)).

describe SHACL and ShEx with more detail in Section 3.1.2. Figure 3.1 shows a comparison between the definitions of an ontology and a shapes graph based on the example data graph presented previously in Figure 2.1.

The main goal for a schema is to facilitate the understanding of the semantics of data, and provide the structure of the data. This can, for example, make more effective and efficient the job of programmers and data engineers when producing or consuming data.

3.1.1 Dynamic schema problem

Although there are several options for describing the schema of a knowledge graph, all of them (including the ones discussed above) present an issue, which is that *they do not describe the current structure of data*. For instance, a schema could state that a customer entity must have between one and three last names, but in reality all instances have only one. In that scenario, all customers are valid and properly satisfy the constraint, but they exhibit a *slightly different structure* from what the knowledge engineers envisioned. The standards presented above allow users to describe upfront the structure of the data, indicating what is allowed and valid on the eyes of knowledge engineers; however, one of the core features of the RDF language is the schema-less notion, one can say anything about anything. Knowledge graphs allow to add new information, relationships and entities, a posteriori. This feature allows the definition of new entities and facts (relations between the entities) in a free way, without having to comply with rigorous schemas (see Example 3.1).

In other words, we can say that knowledge graphs expressed in RDF have a dynamic schema (Muñoz, 2016; González and Hogan, 2018; Hogan, 2018). Ontologies as in OWL and RDFS are focused on 'real-world things' in a given domain, and have the goal of reasoning

where inference can be applied on top of rules they allow users to define. Since the value of ontologies is on the level of details in which they describe the real-world, they usually tend to be very complex and hard to understand by most. Although ontologies are used by knowledge practitioners to define constraints, in terms of data validation, ontologies should never be used for validating instance data. Because OWL, RDFS and RDF adopt the OWA, it cannot be ensured whether the instance data really complies with all the conditions stated during the data modelling process. SHACL and ShEx are good alternatives to deal with this, since they are aimed for rigorous data validation and use a CWA (cf. Section 2.1.3).

Example 3.1

In an RDF knowledge graph, we can insert new facts without having an entity type definition. Let us say we start with few triples containing information about a hospital patient:

```
@prefix ex: <http://www.example.org/> .
@prefix schema: <http://www.schema.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

ex:Bianca rdf:type schema:Patient ;
    schema:name "Bianca" ;
    schema:gender schema:Female ;
    schema:birthDate "1987-09-09"^^xsd:date .
```

Then we can add more information about this patient:

```
ex:Bianca schema:spouse ex:Fernando ;
    schema:healthCondition ex:Condition_A .
ex:Condition_A rdf:type schema:MedicalCondition ;
    schema:name "PhD Stress" ;
    schema:signOrSymptom [
        rdf:type schema:MedicalSignOrSymptom ;
        schema:name "feeling of procrastination"
    ] .
```

And later define the classes:

```
schema:Patient schema:diagnosis schema:MedicalCondition ;
    schema:drug schema:Drug ;
    schema:healthCondition schema:MedicalCondition .
```

As a result, the schema of an RDF knowledge graph is not priorly fixed but dynamic and ever-changing.

A proposed solution to deal with the dynamic schema of graphs is to infer a data-driven schema. Hogan (2018) gives some example desiderata for such approach that we reproduce in the following:

Scalability: Given that some knowledge graphs are in the order of millions of nodes and edges, a suitable notion of schema should be computable from graphs of that size.

Stability: A minor change in the underlying graph should not be able to affect a major change in the corresponding schema.

Conciseness: The schema should be significantly smaller than the graph that it describes.

Connectivity: The schema should not simply describe the nodes in the graph, but should capture information on how the graph is connected.

Readability: The schema should be human-interpretable, meaning that its structure can be directly understood rather than representing abstract objects without direct significance.

The desiderata proposed in Hogan (2018) is presented as a guide towards a general notion of schema for graphs. In this dissertation, we consider the desiderata proposed by Hogan in our contribution in Chapter 4 when extracting cardinality information to uncover the structure of knowledge graphs. And in Chapter 5 when defining an approximate algorithm for knowledge graphs validation.

Next, we review the languages proposed in literature to describe constraints and validate knowledge graphs.

3.1.2 Validation approaches for knowledge graph

To check whether instance data satisfy a given set of constraints (i.e., ensure data consistency) is a very important requirement for any structured data model. This process is also known as *data validation* and is commonly achieved using constraint languages. A few examples of constraint languages are: SQL for relational databases; XML Schema, DTD, Schematron, and RelaxNG for XML; JSON Schema for JSON; and CSV Schema for CSV. These languages provide means for describing schema structure and constraints to data.

Due to the latent need for constraint languages in applications dealing with RDF knowledge graphs, several initiatives have been proposed in recent years. In this section, we briefly describe the most representative approaches that have been used for validating RDF knowledge graphs. Figure 3.2 shows a categorisation of RDF knowledge graph validation approaches using four categories (Meester, Heyvaert, Dörthe Arndt, et al., 2019). Below we present the four categories of validation approaches and their description in the following sub-sections:

- Hard-coded approaches (Section 3.1.2.1),
- Integrity constraint approaches (Section 3.1.2.2),
- Query-based approaches (Section 3.1.2.3), and
- High-level languages (Section 3.1.2.4).

For more details on the practical side and examples of validation languages for RDF knowledge graphs, we encourage the reader to consult the book “Validating RDF Data”

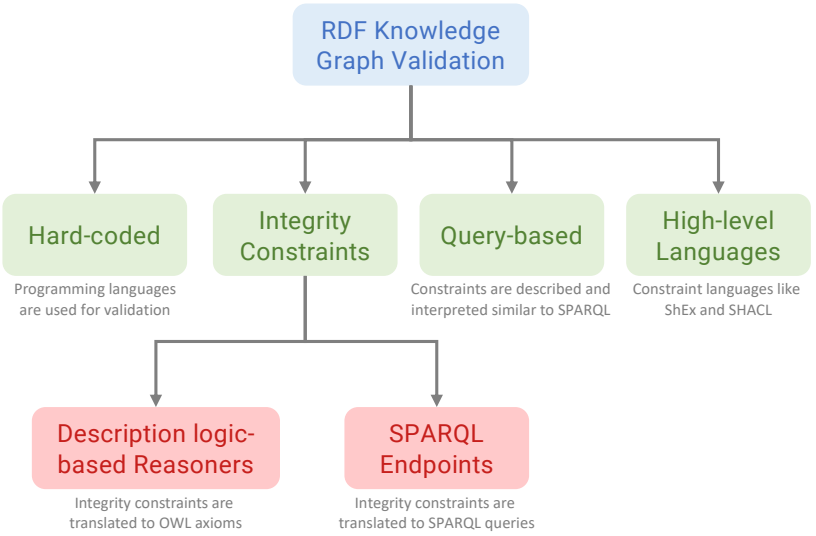


Figure 3.2: Classification of RDF knowledge graph approaches (Meester, Heyvaert, Dörthe Arndt, et al., 2019).

by Labra Gayo, E. Prud'hommeaux, Boneva, et al. (2018). This books introduces data validation in a practical way and focuses mainly on high-level languages, namely, ShEx and SHACL, presenting their design and a comparison.

3.1.2.1 Hard-coded approaches

This is one of the first approaches taken for data validation, where both description and validation of constraints is embedded in the source code of software systems. These hard-coded systems contain the business rules and the routines to evaluate them. They are usually seen as black box approaches.[3] Early in the adoption of RDF, Hogan, Harth, Passant, et al. (2010) analysed the quality of the Semantic Web and provided a list of frequently observed problems in RDF publishing. Later in Hogan, Umbrich, Harth, et al. (2012), the authors analysed again RDF data crawled from the Web (ca. 1.106 billion of unique N-Quads from May 2010), and provide some of the first guidelines in the form of best practices for RDF data providers. Among the conclusions of Hogan et al., we highlight the statement: "universal notions of quality are inherently difficult to pinpoint and measure", which we relate to the definition of data quality given by R. Y. Wang and Strong (1996) in which quality is associated with the concept of "fitness for use". Thus, we can say that quality is dependant of use cases and context where the data is intended to be used.

[3] A black box approach is that where only the input and output are known, but how the internals work is unknown.

3.1.2.2 Integrity constraint approaches

There are two main limitations against the use of RDFS and OWL for validating RDF knowledge graphs, namely, the OWA and nUNA. This makes hard to evaluate consistency, and even inappropriate, when some of the relations are not explicitly stated. Some validation approaches interpret the axioms found in ontologies as *integrity constraints*, which are used to validate RDF knowledge graphs (Motik, Horrocks, and Sattler, 2009; Tao, Sirin, Bao, et al., 2010; Patel-Schneider, 2015). In literature, there are two of these approaches:

Description Logic-based reasoners. OWL axioms can be interpreted using Description Logics. Motik, Horrocks, and Sattler (2009) compared the approaches in OWL and relational databases for schema modelling, schema and data reasoning problems, and constraint checking. They presented algorithms for checking integrity constraints satisfaction for different kinds of ontologies. In relational data, integrity constraints satisfaction checking correspond to model checking, whereas in Description Logics this does not hold. In Description Logics, the only form of constraints checking is checking the satisfiability of an A-Box (data) w.r.t. a T-Box (schema), which is a different problem not concerned with the shape of data. Their approach extended OWL using general Description Logics and propose to differentiate between regular axioms and other T-Box axioms to be designated as integrity constraints. Moreover, they propose algorithms for checking (validating) data against these newly defined integrity constraints.

SPARQL endpoint. Tao, Sirin, Bao, et al. (2010) showed that the expressiveness allowed by OWL is limited to express integrity constraints and proposed a different semantics for the interpretation of ontology axioms: to adopt a CWA with a weak variant of the UNA for that. Such alternative semantics of OWL provides knowledge practitioners with the capability to combine open-world reasoning and closed-world constraint validation. They showed that integrity constraints validation can be reduced to answering SPARQL queries under certain conditions. More specifically, Tao et al. described RDF, RDFS, and OWL axioms using SPARQL queries designed with property paths that simulate the `rdfs:subClassOf` entailment. Their work was implemented in Stardog ICV[4], a commercial knowledge graph management platform (see Example 3.2). SPARQL 1.1 query language is used to translate this kind of constraints and identify violations whenever a non-empty result is returned.

Recently, Patel-Schneider (2015) draws our attention to how most use cases using OWL to express integrity constraints tend to adopt the OWA without the UNA for the parts of the knowledge graph that are known to be incomplete, whereas the CWA with the UNA can be adopted otherwise (when the data is complete). Thus, inference will be applied only over OWL axioms following the OWA, while the integrity constraints will be used for validation with the CWA. In his setting, validation is separated into integrity constraints and closed world recognition, and similarly to Tao, Sirin, Bao, et al. (2010), a translation of RDF and RDFS axioms to SPARQL queries is proposed.

[4] `https://www.stardog.com` (Accessed on June 9th, 2018)

```
ASK {
    { SELECT ?Person {
            ?Person rdf:type schema:Person .
            ?Person schema:name ?o .
        } GROUP BY ?Person HAVING (COUNT(*) = 1)
    }
    { SELECT ?Person {
            ?Person rdf:type schema:Person .
            ?Person schema:name ?o .
            FILTER (isLiteral(?o) && datatype(?o) = xsd:string)
        } GROUP BY ?Person HAVING (COUNT(*) = 1)
    }
}
```

Figure 3.3: SPARQL query to check whether a person has one name of type literal.

Example 3.2

In the snippet below, we show some integrity constraints using Stardog ICV notation to declare that instances of `schema:Person` type must have exactly one value for the (functional) property `schema:name`, which must be a `xsd:string` literal.

```
schema:Person a owl:class ;
rdfs:subClassOf [ owl:onProperty schema:name ;
owl:minCardinality 1 ] .
schema:name a owl:DatatypeProperty , owl:FunctionalProperty ;
rdfs:domain schema:Person ;
rdfs:range xsd:string .
```

3.1.2.3 Query-based approaches

This group of approaches performs validation of RDF knowledge graph using queries described and executed similarly to the SPARQL query language (S. Harris, Seaborne, and E. Prud'hommeaux, 2013). Using SPARQL, constraints are defined as queries that can be executed against the data graph in an SPARQL endpoint: only the pieces of the RDF knowledge graph that are compatible with the structure defined by the queries are returned to the user. SPARQL provides with a good level of expressiveness that allows to describe test patterns as queries (Kontokostas, Westphal, Auer, et al., 2014). Such patterns can be knowledge graph and ontology agnostic or tailored to specific use cases. For instance, to validate the data of a person one could use a SPARQL query like in Figure 3.3 to check that a person has only one name and this is a literal. The query in Figure 3.3 will return a simple `true` value if all `?Person` entities of type `schema:Person` have only one property `schema:name` and this property is a `xsd:string` literal. This category covers more generic approaches when compared with the previously mentioned implementation of integrity constraints using SPARQL (e.g., Tao, Sirin, Bao, et al. (2010)).

Similarly, other kinds of logical constraints can also be defined following the semantics of SPARQL. In Labra Gayo, E. Prud'hommeaux, Boneva, et al. (2018), the pros and cons of using SPARQL for data validation are analysed. On the one hand, we have the benefits that

SPARQL is:

(a) very expressive and can handle most RDF validation needs; and
(b) ubiquitous since most RDF products support it.

On the other hand, there are also some problems because SPARQL is:

(c) very expensive to compute and also very verbose (making difficult to write and debug by non-experts);
(d) non-deterministic in the sense that the same constraint can be expressed in different ways; and
(e) complex to exhaustively write queries which accept all valid permutations and reject all incorrect structures.

SPARQL has been adopted in other validation initiatives such as SPARQL Inferencing Notation (SPIN) (Knublauch, J. A. Hendler, and Idehen, 2011). SPIN is a low-level language that allows users to define templates and user-defined functions using SPARQL to validate RDF knowledge graphs. In Fürber and Hepp (2010), the authors propose to use SPIN for identifying data quality problems in the Semantic Web data (a.k.a. Web of Data). In the same vein, Kontokostas, Westphal, Auer, et al. (2014) define so-called Data Quality Test Patterns (DQTP), which are tuples (V, S), where V a set of typed pattern variables and S is a SPARQL query template with placeholders for the variables from V.

Example 3.3

An example DQTP is as follows, where we test for the cardinality of a relation `P1` comparing it with a value `V1` using operator `OP` (e.g., $<, <=, >, >=, =, !=$). Note that here $V = \{\texttt{P1}, \texttt{V1}, \texttt{OP}\}$.

```
SELECT DISTINCT ?s WHERE { ?s P1 ?c }
GROUP BY ?s HAVING count(?c) OP V1
```

An instantiation of this DQTP could be defined using `P1=schema:birthDate`, `OP=>`, and `V1=1` to assess entities that have more than one date of birth.

Kontokostas et al. developed a validation framework for their DQTP called RDFUnit[5], which recently in 2017 also added support for constraints defined in SHACL.

3.1.2.4 High-level languages

Although core languages of the Semantic Web, namely, SPARQL and OWL with CWA have been used for RDF validation, they are considered to be too complex and inappropriate for the validation task. Technically, their goal is to support inferences in a reasoner software, which significantly differs from what a validation software does. Thus, several proposals

[5]Project web site at http://aksw.org/Projects/RDFUnit.html.

for new high-level and declarative languages were proposed in recent years for defining constraints for RDF knowledge graphs.

So far we have mentioned validation approaches that are tied to some technologies and implementations. High-level languages are specifically defined to describe constraints in a technology and implementation agnostic manner (Coyle and Tom Baker, 2013; Fokoue and A. Ryman, 2013; T. Hartmann, 2016; Labra Gayo, E. Prud'hommeaux, Boneva, et al., 2018). Several proposal of such languages have been presented in the recent years, but only few of them have been widely adopted by the community. Next, we present some of these proposals.

Resource description languages. The first of these languages we review is the Description Set Profiles (DSP) (Nilsson, 2008), which describes the structure of a *description set*[6] by using templates and constraints. In DSP, templates are used to express structures, while constraints are used to limit those structures. DSP was proposed as a formal representation of constraints of a Dublin Core Application Profile (Coyle and Tom Baker, 2013).

Example 3.4

The following DSP example matches descriptions with a single resource, which must be an instance of the class `foaf:Person`.

```
<?xml version="1.0" ?>
<DescriptionSetTemplate xmlns="http://dublincore.org/xml/dc-dsp/2008/03/31" >
    <DescriptionTemplate ID="person" minOccurs="1" maxOccurs="1" standalone="yes">
        <ResourceClass>http://xmlns.com/foaf/0.1/Person</ResourceClass>
    </DescriptionTemplate>
</DescriptionSetTemplate>
```

Later, the RDF Data Descriptions (RDD) (Schmidt and Lausen, 2013) was proposed in COLD workshop 2013. An RDD allows to specify instance-level data constraints over RDF—akin to DTDs for XML—using First-order Logics, which facilitated their translation to SPARQL. The language also allows to specify which semantics to use at different scopes, i.e., one can validate some classes using the CWA and others using the OWA. The RDD language provides a user-readable and machine-processable way to describe constraints, which differs from the previous SPARQL and OWL approaches.

[6] A set of one or more descriptions (or statements) that describe a single resource.

Example 3.5

An example RDF Data Description is as follows, specifying that the name of a person is the identifier of the person, and the range of schema:knows is a IRI.

```
CWA CLASSES {
    OWA CLASS schema:Person {
        KEY schema:name : LITERAL ;
        RANGE(schema:Person) schema:knows : IRI ;
    }
}
```

Also proposed in 2013, are the languages Open Services for Lifecycle Collaboration (OSLC) Resource Shapes (Fokoue and A. Ryman, 2013; A. G. Ryman, Hors, and Speicher, 2013) by IBM and the Dublin Core Application Profiles (Coyle and Tom Baker, 2013). These languages were presented and discussed in the W3C/MIT hosted workshop of RDF validation in September 2013.[7] A Resource Shape is an RDF vocabulary for specifying and validating constraints on RDF knowledge graphs.

Example 3.6

Example on how to represent the constraints on range and cardinality of the schema:name property in OSLC.

```
:customer a oslc:ResourceShape ;
oslc:property [
    dcterms:title "name" ;
    oslc:propertyDefinition schema:name ;
    oslc:valueType xsd:string ;
    oslc:occurs oslc:Exactly-one
] .
```

Languages for describing the structure of knowledge graphs have been well received in the community, but most importantly, they showed to be a powerful alternative to SPARQL and OWL with the potential to become an standard. In the next two sub-sections, we review Shape Expressions and SHACL, two of the latest languages coming out of W3C groups that have been inspired somehow by OSCL, Dublin Core Application Profiles, RDF Data Descriptions, and SPIN.

Shape expressions. Shape Expressions (ShEx) is intended to be an RDF constraint language, which allows to describe the structure or "shape" of RDF knowledge graphs in a human-readable syntax (E. Prud'hommeaux, Labra Gayo, and Solbrig, 2014). It is based on RELAX NG[8] Compact Syntax with conventions similar to Turtle or SPARQL. We follow Boneva, Labra Gayo, and E. G. Prud'hommeaux (2017) for describing a subset of ShEx

[7] RDF validation workshop (https://www.w3.org/2012/12/rdf-val/).

[8] REgular LAnguage for XML Next Generation was defined by a committee specification of the OASIS RELAX NG technical committee in 2001 and 2002.

shape schemas. A shapes schema $\mathcal{S}$ defines a set of named shapes, where a shape is a description of the graph structure that can be visited starting from a particular node. Shapes are expressed using shape expressions, where (Boolean combinations of) other shapes and recursion is allowed.

Formally, a *shapes schema* $\mathcal{S}$ is a pair $(\mathcal{L}, def)$, where $\mathcal{L}$ denotes a set of *shape labels* used as names of shapes and def is a function that maps a shape label with a shape expression. $L \rightarrow S$ is used as a short for $def(L) = S$, where a shape label L is associated with a shape expression S. A *shape expression* is a Boolean combination of two atomic components: value description and neighbourhood description. A *value description* is a set that declares the admissible values for a node: IRIs, literals and blank nodes. While a *neighbourhood description* defines the expected neighbourhood of a node and is given by a triple expression. A *triple expression* is a group of expressions that describes the expected neighbourhood of a node likewise DTDs and XML Schema in XML. A triple expression is composed of each-of (separated by the semicolor operator '`;`'), some-of (separated by the pipe '`|`') and repetition operator is satisfied if some distribution of the triples in the neighbourhood of a node exactly satisfies the expression.

Example 3.7

To illustrate these definitions, we consider the shape schema $\mathcal{S}_0$ below comprising two shape definitions (enclosed by curly braces `{}`) with shape labels `ex:PersonShape` and `ex:PatientShape`, respectively. Everything what is between the curly braces is known as a shape, and contain triple expressions making use of all operators: each-of (`;`), some-of (`|`), and repetition. In the *def* (`ex:PersonShape`) shape, there is one triple expression, which in turn contains five triple constraints separated by '`;`'.

```
ex:PersonShape {
    schema:name xsd:string {1} ;
    schema:address schema:PostalAddress {1, 3} ;
    schema:age xsd:integer MinInclusive 18 MaxInclusive 99 ;
    schema:email IRI * ;
    ^schema:knows @ex:PersonShape ?
}
ex:PatientShape {
    ex:patientNumber xsd:integer | schema:identifier IRI
}
```

The shape expression `ex:PersonShape` → `schema:name xsd:string {1} ; schema:address schema:PostalAddress {1, 3} ...` is a shape, where `schema:name xsd:string {1}` is one of the five constraints in the triple expression.

A ShEx schema is a collection of expressions that describe the constraints that a given RDF graph should meet to be considered valid. It contains the allowed values for the predicates of an entity, directionality of the relations, and cardinality constraints for the number of allowed occurrences of a predicate. In our example 3.7, the ShEx schema constrains in-

stances of the ex:PersonShape entity type[9] as follows: a single name of type literal; one to three addresses of type schema:PostalAddress; an age between 18 and 99 years; zero or more (* in regular expressions) e-mail addresses for contact; and at least one (? in regular expressions) contact (defined recursively by the same shape indicated by the '@' symbol).

Note that a ShEx schema can be serialised in three different ways, namely, **ShExC** for Shape Expressions Compact Syntax, **ShExJ** for JSON-LD (JavasSript) syntax, and **ShExR** for RDF Turtle syntax. In this work, we use ShExC because of its human-friendly syntax. For more details, we refer readers to the ShEx Primer (Bake and E. Prud'hommeaux, 2017).

Example 3.8

In the SHACL shape below, we specify that the target node ex:Alice should have one schema:name value of type literal, one to three addresses, the gender should be either schema:Male or schema:Female, and the values of schema:knows should be of type schema:Person.

```
ex:Person a sh:NodeShape ;
sh:targetNode ex:Alice ;
sh:property [ sh:path schema:name ;
    sh:minCount 1 ;
    sh:maxCount 1 ;
    sh:datatype xsd:string ;
] ;
sh:property [ sh:path schema:address ;
    sh:minCount 1 ;
    sh:maxCount 3 ;
    sh:nodeKind sh:IRI ;
] ;
sh:property [ sh:path schema:gender ;
    sh:in (schema:Male schema:Female) ;
] ;
sh:property [ sh:path schema:knows ;
    sh:class schema:Person ;
] .
```

Shapes constraint language. Shapes constraint language (SHACL) is a W3C recommendation released in 2017. SHACL has a lot in common with ShEx, specially the goal to become the *de facto* constraint language for RDF knowledge graphs. Likewise, SHACL also allows to define shapes that nodes in the graph must satisfy (see Example 3.8). It was influenced mainly by SPIN, but also from OSLC Resource Shapes and ShEx.

The grammars of SHACL and ShEx are different; however, their expressivity is very similar (Labra Gayo, García-González, Fernández-Alvarez, et al., 2019). In SHACL, there are two main types of shapes: node shapes and property shapes. *Node shapes* allow to declare constraints over a node, whilst *property shapes* allow to declare constraints on the values

[9] However, there is no functional relation between entity types and shapes. A node can have zero or more rdf:type relations, and different applications can apply different shapes to the nodes, i.e., a node can have different meaning and different structure depending on the context.

associated with a node through a path. A node shape usually contains several property shapes declared using the `sh:property` relation. While property shapes use `sh:path` to declare the path—usually a single relation IRI—that goes from the focus node to the value they describe.

Unlike ShEx, there are very few scientific publications studying the SHACL language formalisation and semantics. Corman, Reutter, and Savkovic (2018b) propose a formal semantics for the SHACL core, which can handle the arbitrary recursion proposed in the specification of SHACL but explicitly left undefined. Due to the status of recommendation and the growing interest in SHACL, it is to expect that more works like this will appear in the coming years.

Another important part of SHACL are the validation reports. A *validation report*, represented by `sh:ValidationReport`, is the value returned by a SHACL processor after validation. If the RDF knowledge graph conforms to the shapes graph, then the report returns a `sh:conforms` declaration with value `true`; and `false` and a set of validation errors otherwise. A user can define the result message returned in each validation component. For more details, we refer readers to the SHACL recommendation (Knublauch and Kontokostas, 2017).

3.2 Schema inference approaches

A *data constraint* is a pattern that must be present in data. If such patterns are known, they are grouped in a so-called *schema*—probably declared using a language like the ones we have reviewed in Section 3.1.2. But clearly, there will be times when such patterns will not be explicit during the production/consumption of data. There will be cases in which the patterns will not be fully enumerated, and cases where the patterns will not be intuitive even for the data producers. We have mentioned, however, that even when the patterns are fully enumerated the instance data might not follow them. Thus, it becomes relevant to have methods for enumerating patterns fully and for inferring the schema.

In relational databases, data producers must follow a *schema-first* approach (Pham and Boncz, 2016), where the schema is defined upfront. And all instances of a database are ensured to follow the pre-defined schema. Similarly, the schema-first approach has been traditionally adopted by the Semantic Web community; however, this contradicts the OWA present in RDF and OWL. Following a semantic approach (under the OWA), one should not assume any upfront schema, which is also called the *schema-last* approach. Such contradiction is perhaps one of the major issues for data consumers who require a clear schema of the data to write queries and manage knowledge graphs (Lausen, Meier, and Schmidt, 2008).

To tackle the lack of a rigid schema, the research community has proposed several *schema inference* approaches that aim to discover the schema and constraints that data naturally satisfy. Several techniques have been proposed to infer a schema from RDF knowledge graphs using so-called bottom-up or data-driven approaches that take the instance data as input

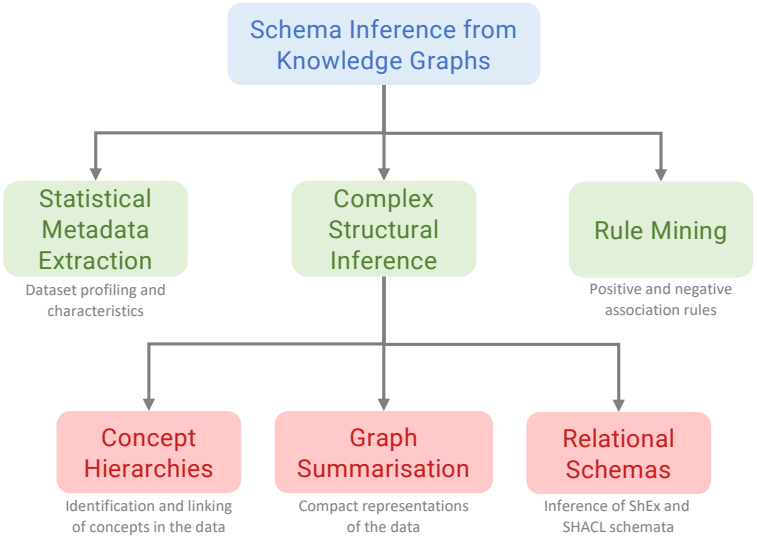


Figure 3.4: Classification of schema inference approaches from knowledge graphs.

and return a schema. In this section, we briefly describe the most representative approaches that have been used for inferring schemas from knowledge graphs. Figure 3.4 shows a categorisation of approaches for schema inference from knowledge graph using three main categories. Below we present the three categories and describe them in the following subsections:

- Statistical metadata extraction approaches (Section 3.2.1),
- Rule mining approaches (Section 3.2.2), and
- Complex structural inference approaches (Section 3.2.3).

3.2.1 Statistical metadata extraction

Statistical approaches focus on the statistical distribution of classes and properties across a knowledge graph, which is also known as dataset profiling. They usually compile dataset metadata and aggregated statistics such as triple count, number of (incoming/outgoing) links, among others (Fernández, Martínez-Prieto, Fuente Redondo, et al., 2018). Several tools have been built and published to profile and mine RDF knowledge graphs and expose their characteristics. ExpLOD (Khatchadourian and Consens, 2010) analyses the Linked Open Data (LOD) cloud, which is a set of RDF graphs available on the Web with links among them. ProLOD++ (Abedjan, Grütze, Jentzsch, et al., 2014) does a similar profile of LOD datasets and provides association rule discovery to uncover synonymous predicates, and uniqueness discovery along ontology hierarchies. Loupe (Mihindukulasooriya, Poveda-Villalón, García-Castro, et al., 2015) is another tool that exposes the main characteristics of RDF knowledge graphs and allows to explore the classes and properties, similar to

Laundromat (Beek, Rietveld, Bazoobandi, et al., 2014), LODstats (Ermilov, Lehmann, Martin, et al., 2016), ABSTAT (Spahiu, Porrini, Palmonari, et al., 2016) and LODAtlas (Pietriga, Gözükan, Appert, et al., 2018). Recently, in Rietveld, Beek, Hoekstra, et al. (2017), the authors developed LOD Laundromat Meta Dataset, which exposes metadata of most LOD datasets on the Web in a uniform way. RDF knowledge graphs can also be available as online services. SPORTAL (Hasnain, Mehmood, Zainab, et al., 2016) is a tool that submits queries to SPARQL endpoints to obtain detailed profiles of the content in the endpoint. The output description of the datasets is then published in an online catalogue. Because submitting aggregate queries to SPARQL endpoints may be too expensive and therefore not feasible (most of the time), in Soulet and Suchanek (2019) the authors study re-writing techniques for profiling queries. For a recent survey, we refer the reader to Ellefi, Bellahsene, Breslin, et al. (2018) that proposes a taxonomy of dataset profiling approaches.

3.2.2 Rule mining

Rule mining approaches extract frequent rule-form patterns hidden in the data. The usual requirements for methods that extract rules from knowledge graphs are:

(a) scalability to learn from large-scale knowledge graphs comprising billions of triples;
(b) robustness to tolerate a certain number of incorrect facts present in the knowledge graph; and
(c) the ability to cope with uncertainty to generate certainty values for the inferred rules or axioms.

Völker and Niepert (2011) introduced a statistical approach to the mining of association rules of the form $X \Rightarrow Y$, where the *confidence* ($conf(\cdot)$) of a rule is defined based on the *support* ($supp(\cdot)$) of its constituents as follows:

$$supp(X) = \frac{|\{t_i \in D : X \subseteq t_i\}|}{|D|}, \text{ and}$$

$$conf(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)},$$

where the item set $X \subseteq I = \{i_1, i_2, \ldots, i_n\}$, $D = \{t_1, t_2, \ldots, t_m\}$ is a set of m transactions, and each transaction is a subset of items (binary attributes), i.e., $t_i \subseteq I$. These association rules are used to generate OWL 2 EL ontology axioms from an RDF knowledge graph without requiring negative examples. For instance, the association rule $\{C_i\} \rightarrow \{C_j\}$ is translated into the axiom $C \sqsubseteq D$, e.g., $\exists$ birthPlace.PopulatedPlace $\sqsubseteq$ Person (Völker and Niepert, 2011).

ProLOD++ (Abedjan, Grütze, Jentzsch, et al., 2014) computes statistical metadata of datasets, but also applies the FP-Growth algorithm to mine both positive and negative association rules, and performs data cleansing. Such rules could help ontology re-engineering

by, for example, identifying underspecified and overspecified classes. In an underspecified class certain properties frequently present in real-world data are not specified in the vocabulary, while in an overspecified class one or more properties are declared for the class in the ontology, but are rarely (if ever) used for real-world data. ProLOD++ can also be used to discover synonymous predicates or relations ($r_1 = r_2$ or $r_1 = {r_2}^{-1}$) such as `dbp:author` and `dbp:writer`. In Mohamed, Muñoz, Nováček, et al. (2017), we have presented a method to identify equivalence rules as $P_1 \equiv P_2$, where P_1 and P_2 are paths in a knowledge graph. Such path equivalence rules can also help ontology re-engineering and querying.

Inductive Logic Programming (ILP) can also be used to mine Horn clauses from knowledge graphs; however, learning them requires negative examples. AMIE+ was developed to extract Horn rules from large knowledge graphs without the need for negative or counterexamples and assuming a partial closed-world assumption (PCWA) (L. A. Galárraga, Teflioudi, Hose, et al., 2013; L. Galárraga, Teflioudi, Hose, et al., 2015). Examples of rules extracted by AMIE are: $\mathit{relative}(y, z) \wedge \mathit{sister}(z, x) \Rightarrow \mathit{relative}(x, y)$ and $\mathit{livesIn}(h, p) \wedge \mathit{marriedTo}(h, w) \Rightarrow \mathit{livesIn}(w, p)$. In AMIE, the mined rules are also used to predict new facts and complete a knowledge graph. We will review more recent knowledge graph completion approaches in Section 3.3.

3.2.3 Complex structural inferences

Unlike previously described approaches, complex structural inference aims at the provision of schemas or topological information of a knowledge graph. Complex structures go beyond simple statistics and association rules and aim to expose the schema that (most) data follow.

Concept hierarchies. In Cimiano, Hotho, and Staab (2004), the authors apply clustering techniques based on context vectors and Formal Concept Analysis (FCA) to construct taxonomies from textual data. Other approaches also propose the use of clustering (Maedche and Zacharias, 2002) using graph-based and ontological distances, and ILP-based approaches (d'Amato, Fanizzi, and Esposito, 2010) to construct taxonomies from Linked Data. Similarly, for the friend-of-a-friend (FOAF) network on the Semantic Web, Grimnes, Edwards, and Preece (2004) apply clustering to identify classes of people and ILP to learn descriptions of these groups. DL-Learner (Lehmann, 2009) is another ILP-based approach successfully applied to DBpedia (Auer, Bizer, Kobilarov, et al., 2007) and YAGO (Suchanek, Kasneci, and Weikum, 2007) is able to generate OWL class expressions (Hellmann, Lehmann, and Auer, 2009).

In Kellou-Menouer and Kedad (2015), the authors extract type definitions described by profiles, i.e., property vector where each property is associated with a probability. A further analysis of semantic and hierarchical links between types (or classes) is done to extract a global schema. They only consider direct relations and property paths but ignore `rdf:type` triples. A hierarchical organisation of entity types (classes) is extracted in Christodoulou,

Paton, and Fernandes (2015) using clustering analysis at instance-level grouping together entities with similar sets of properties. Their hierarchical organisation does not take into account any notion of cardinality of relations. They serialise the inferred schema—modelled using UML—back to RDF triples using a vocabulary defined by themselves.

Recent applications of FCA have also been scaled to large heterogeneous knowledge graphs to identify hierarchical structures called *formal concept lattice* (González and Hogan, 2018). This new representation allows to define an algebra over lattices that unlocks the study of dynamic knowledge graphs. For example, formal concept lattices could be used to identifying past or future changes by looking at several snapshots in time of a knowledge graph (Käfer, Umbrich, Hogan, et al., 2012). The authors evaluate their approach by identifying changes in Wikidata (Vrandecic, 2012) over 11 weeks of data updates.

Graph summarisation. Graph summarisation aims to describe data using a small amount of information. A summary is composed of graph patterns that can be seen as views, which can help users to understand complex knowledge graphs easily (Song, Yinghui Wu, P. Lin, et al., 2018). Čebirić, Goasdoué, and Manolescu (2015) define some desiderata that RDF knowledge graph summaries should satisfy which we reproduce in the following:

Completeness: The saturation of the summary of $\mathcal{G}$ must be the same as the summary of its saturation $\mathcal{G}^\infty$[10], due to the semantics of an RDF graph being its saturation.

Schema independence: It must be possible to summarize $\mathcal{G}$ whether or not it has associated RDFS triples.

Compactness: The summary should be typically smaller than the RDF knowledge graph, ideally by orders of magnitude.

Representativeness: The summary should not lose too much information from $\mathcal{G}$.

Accuracy: The summary should avoid, to the extent possible, reflecting data that does not exist in $\mathcal{G}$.

Different summarisation approaches can choose whether to satisfy all or part of the desiderata proposed by Čebirić et al. However, some trade-off exists between the compactness and representativeness of summaries, so that no information is lost while aiming for small summaries. Figure 3.5 shows one of the graph summaries that could be extracted from the knowledge graph in Figure 2.1. For recent surveys of knowledge graph summarisation methods we refer the reader to (Song, Yinghui Wu, P. Lin, et al., 2018; Čebirić, Goasdoué, Kondylakis, et al., 2019).

Relational schemas. More recent approaches have been trying to extract relational schemas that can be used for validation or checking of a knowledge graph. These approaches form

[10]The saturation $\mathcal{G}^\infty$ is defined as the closure, i.e., the fixed-point graph obtained after recursively applying entailment rules on $\mathcal{G}$.

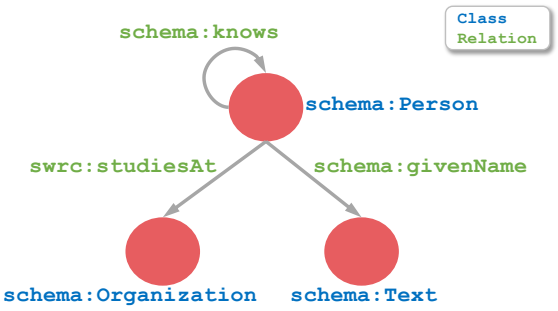


Figure 3.5: Example graph summary from the knowledge graph in Figure 2.1.

part of a bigger research area that aims to infer schemas from different databases, e.g., XML schema inference (Bex, Gelade, Neven, et al., 2010) and JSON schema inference (Baazizi, Colazzo, Ghelli, et al., 2019).

Rivero, Hernández, Ruiz, et al. (2012) uses straightforward SPARQL 1.1 queries to automatically discover ontological models that contain types and properties, sub-types, domain, range, and minimum cardinalities of these properties. More recent works use statistical patterns to build shape graphs from instance data Fernández-Álvarez, García-González, Frey, et al. (2018). The profiles extracted by Fernández-Álvarez et al. are then serialised as ShEx, which is a widely used RDF constraint language with several validation tools (see Example 3.9).

Example 3.9

The following is a ShEx shape inferred in Fernández-Álvarez, García-González, Frey, et al. (2018) when analysing the dbo:Country class from DBpedia.

```
:Country {
    rdf:type [dbo:Country] ;                    # 100.0%
    dbo:wikiPageID xsd:integer ;                # 97.1%
    owl:sameAs IRI + ;                          # 96.9%
    foaf:name xsd:string + ;                    # 96.6%
    dcterms:subject IRI + ;                     # 96.0%
    dbo:dissolutionYear xsd:gYear + ;           # 83.1%
                # 82.6% have cardinality { 1 }
    dbo:foundingYear xsd:gYear + ;               # 82.0%
                # 81.5% have cardinality { 1 }
    dbp:continent rdf:langString +               # 80.6%
                # 80.3% have cardinality { 1 }
}
```

Each constraint pattern generated by the algorithm is associated with a *trustworthiness score*, which reflects the confidence in a rule obtained from rule mining approaches.

A workflow for RDF Shape induction is proposed in Mihindukulasooriya, Rashid, Rizzo, et al. (2018) that mixes profiling techniques and machine learning to automatically generate SHACL shapes for RDF knowledge graphs. They evaluated their work over a sub-

set of DBpedia achieving 97% precision on the derived shapes and cardinality constraints. Similarly, in Spahiu, Maurino, and Palmonari (2018), the authors use ABSTAT (Spahiu, Porrini, Palmonari, et al., 2016) to learn semantic profiles of a knowledge graph and convert them into SHACL constraints that can be used to assess the quality of the data. Although there is a growing body of literature characterizing knowledge graphs and trying to extract structural patterns, these works focus on an obtaining a global schema for data, and are not robust enough since the semantic of data is mostly ignored. For example, equivalence axioms like the ones encoded by `owl:sameAs` statements are not accounted when computing cardinality. Finally, their end goal is the provision of a serialised schema shape for knowledge graphs and not how the found patterns could be used by other applications.

3.2.4 Elements of structure: cardinality constraints

Although many of the elements that compose a knowledge graph schema (e.g., classes, relationships) have been widely studied in literature, there are still some gaps. In this thesis, we hypothesise that elements like cardinality can be used to expose the structure that data naturally exhibit as it does for relational databases (Thalheim, 1992; Liddle, Embley, and Woodfield, 1993) and XML (Ferrarotti, S. Hartmann, and Link, 2013; Ferrarotti, S. Hartmann, Link, et al., 2013). Cardinality is present in every relation between entities. For example, given the soccer player (entity) *Alexis Sánchez* and the relation *playsFor*, we can expect that this relation will have at least one occurrence with a football team (many in reality), whereas if we take the country (entity) *Republic of Ireland* and the relation *hasPresident*, we can only map a single person because of the functional nature of the relation. In knowledge graphs, especially when encoded using RDF, there may be semantic elements that affect cardinality like the equality axioms (`owl:sameAs` and the like) or noise introduced by erroneous/false facts in the data.

The notion of cardinality is crucial for the reminder of this thesis, thus, here we provide an overview of the concept and the implications it has for the structure of data. Mathematically, the cardinality of a finite set is the number of elements it contains. In the context of knowledge graphs, data we only consider finite sets of entities and relations. (Note that still you can say that there are infinite combinations of characters to name or label an entity, but not all of them are actually used in a knowledge graph instance.) Example 3.10 shows different ways to express cardinality in a mathematical form.

Henceforth, our focus will be on relation cardinalities. *Relation cardinality* (also referred to as *multiplicity*) covers a critical aspect of relational data modelling, referring to the number of times an instance of one entity can be related with instances of another entity though this relation. Information about cardinalities can be useful for data users and knowledge engineers when writing queries, reusing or engineering knowledge graphs (Schmidt, Meier, and Lausen, 2010). We will in Chapter 4 show how the aforementioned problem of missing structure, can be (partially) overcome by mining cardinality from instance data. We use re-

lation cardinality to obtain the latent structure of a knowledge graph: The cardinality of a relation limits the number of values that may have for a given entity. We argue that extracting cardinality from knowledge graphs can help in the following ways: (i) to communicate the structure of a given entity type (class); (ii) to measure the completeness and correctness of a KG; and (iii) to guide knowledge graph completion methods to achieve results that better match the knowledge graph structure. We believe that tools that capture the structure of entities and entity types are key elements for completing a KG.

Example 3.10

Let us consider the following statements that give information about the number of elements allowed in the domain of a structure (Grant and Hunter, 2006).

- $E_1 = \exists x_1 x_2 x_3 \forall y (x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_2 \neq x_3 \wedge (y = x_1 \vee y = x_2 \vee y = x_3))$ restricts the domain to have 3 elements.
- $E_2 = \exists x_1 x_2 x_3 \forall y (y = x_1 \vee y = x_2 \vee y = x_3)$ restricts the domain to have at most 3 elements.
- $E_3 = \forall y (y = a_1 \vee y = a_2 \vee y = a_3)$ restricts the domain to the 3 elements a_1, a_2, and a_3.

These statements provide different ways to express cardinality and boundaries for a domain.

No much attention has been paid to cardinality by the approaches introduced in Section 3.2. In their approach, Völker and Niepert (2011) derive several constraints from the knowledge graph but considered cardinality restrictions (upper bounds) only as future work. Cardinality constraints in RDF have been defined for data validation in languages such as OWL (Motik, Peter F Patel-Schneider, and Parsia, 2012), Shape Expressions (ShEx) (E. Prud'hommeaux, Labra Gayo, and Solbrig, 2014), OSLC Resource Shapes (A. G. Ryman, Hors, and Speicher, 2013), and Dublin Core Description Set Profiles (DSP)[11]. OSLC integrity constraints include cardinality of relations which are more similar to UML cardinality for associations (i.e., exactly-one, one-or-many, zero-or-many, and zero-or-one). Mihindukulasooriya, Rashid, Rizzo, et al. (2018) proposed a machine learning approach to determine OSLC-like cardinality achieving a good precision. However, the expressivity of OSLC is limited compared with the definitions proposed in OWL[12], DSP, ShEx, Shapes Constraint Language (SHACL), Stardog ICV[13], and SPIN Modelling Vocabulary (Knublauch, J. A. Hendler, and Idehen, 2011). These languages define flexible boundaries for cardinality constraints: a lower bound in $\mathbb{N}$, and an upper bound in $\mathbb{N} \cup \{\infty\}$. By definition, it is assumed that cardinality constraints are known beforehand and expressed in one of these languages for validating knowledge graphs.

[11] http://dublincore.org/documents/dc-dsp/
[12] OWL allows to express cardinality through `minCardinality`, `maxCardinality`, and `cardinality` restrictions.
[13] http://docs.stardog.com/icv/icv-specification.html

In Chapter 4, we propose an approach for mining flexible cardinality patterns in an accurate and robust manner. Due to the bottom-up approach, we do not refer to such cardinalities as constraints but as *bounds* (Muñoz and Nickles, 2017), which do not constrain the data but indicate soft bounds. Where these bounds are used to impose structural restrictions, they are also referred to as *cardinality constraints* (i.e., hard bounds). They can (and should) be considered constraints only after a user assessment and use in validation.

3.3 Completion of knowledge graph

In Section 2.3.2 we have defined the completeness of a knowledge graph[14] with the help of a hypothetical ideal knowledge graph $\mathcal{K}^*$ containing all the true facts in the world. Having such a knowledge graph would allow us to "know everything", and to know what we do not know by subtracting $\mathcal{K}$ from $\mathcal{K}^*$. Given a knowledge graph about personal information of people, we may ask the following questions: *does Anthony study in other universities?*, *does Aidan have other nationalities?*, or *are all children of Anthony in the knowledge graph?* Those are questions that we cannot answer if we limit ourselves to the content stored in that knowledge graph. And because it is practically impossible to have such an ideal (complete) knowledge graph for every context and domain of knowledge, many works propose alternatives to deal with the incompleteness of a knowledge graph by, for example, discovering missing links between known entities in the graph. We refer to the problem of finding new links in a knowledge graph as *link prediction*.

Link prediction is one of the tasks that have as goal the *completion* of knowledge graphs. In this section, we describe the tasks involved in knowledge graph completion, and survey relevant approaches that address them. Particularly, we will focus on those models known as neural link predictors (Minervini, Demeester, Rocktäschel, et al., 2017; Rocktäschel, 2018; Muñoz, Minervini, and Nickles, 2019) that apply shallow neural networks to predict the likelihood of triples. Most approaches are based on machine learning techniques which we will also introduce here.

3.3.1 What is knowledge graph completion?

We begin by emphasizing the inequality between completeness and completion. Completeness is an end goal for knowledge graphs which is affected by the OWA, which is the principled reason why one cannot deem a (Web) knowledge graph as complete. There will always be entities and relations that we do not know exist. While the completeness of a knowledge graph dependents on the context and domain of knowledge, the completion's goal (assuming an incomplete knowledge graph) is based on either 1) what we already

[14] Without loss of generality, we use the term knowledge graphs to refer to graph-structured knowledge bases (see Section 2.2 for details).

know, i.e., the knowledge graph itself, or 2) external sources of knowledge (e.g., text corpora or other knowledge graphs). In other words, completeness will try to answer more global questions like *does Aidan have other nationalities?*, while completion deals with more local questions like *which nationalities does Aidan have?*, assuming that we know *Aidan* and all possible nationalities.

Knowledge graph completion (KGC)—also known as *knowledge base completion* (KBC) if the KB has the form of a KG—is the task that deals with automatically understanding the structure of (large-scale[15]) knowledge graphs and predicting missing relationships or links, i.e., inferring new facts (Bordes, Usunier, García-Durán, et al., 2013; Nickel, K. Murphy, Tresp, et al., 2016). Knowledge graph completion methods can be focused on predicting missing entities, missing types of entities, and/or missing relations (links) between entities (Paulheim, 2017). The link prediction task has gained attention based on the high quality predictions required by some applications (e.g., when predicting a side effect for a drug), which makes it one of the main problems in statistical relational learning (Getoor and Taskar, 2007).

3.3.2 Completion tasks

The knowledge graph completion task has been approached from different viewpoints and application contexts. In two recent reviews (Nickel, K. Murphy, Tresp, et al., 2016; Q. Wang, Mao, B. Wang, et al., 2017), four downstream tasks have been identified as part of the completion of knowledge graphs considering solely the triples in the knowledge graph. Link prediction, triple classification, entity resolution and entity classification are the tasks previous works have addressed. Each one presents a different problem, which usually can be cast as a link prediction task. We review these four tasks in the following sections.

Link prediction. The *link prediction* task is concerned with predicting the existence (or probability of correctness) of (typed) edges in the graph, i.e., predicting triples of the form (h, r, t) in the knowledge graph (Nickel, K. Murphy, Tresp, et al., 2016). More formally, it is the task of predicting h given (r, t) or t given (h, r), where the former is denoted as a query (*?*, r, t) and the latter as (h, r, *?*). For example, an answer to the query (*?*, *studiedAt*, *NUI Galway*) is equivalent to predicting the alumni of *NUI Galway*, while an answer to (*Liam Neeson*, *studiedAt*, *?*) is equivalent to predicting the college(s) attended by *Liam Neeson*. A similar idea can also be applied to predict the relation type given the head and tail entities, i.e., predict r given the pair (h, t), or answering (h, *?*, t), which is usually referred to as *relation prediction* (Y. Lin, Z. Liu, Luan, et al., 2015; D. Q. Nguyen, Sirts, L. Qu, et al., 2016). During evaluation, each correct test triple (h, r, t) is corrupted by replacing its subject or object entity (resp. its relation r) by each possible entity (resp. relation type), and then rank these candidates in ascending order of their score. Because link prediction adds missing links to the knowledge graph it is also referred to as *knowledge graph completion* (see Section 3.3).

Usually, the goal of link prediction is to learn a scoring function ϕ that given a triple

[15] This is not a requirement of the KGC task but of the machine learning or deep learning techniques used.

$(h, r, t) \in \mathcal{G}$ returns its corresponding *score*, $\phi\colon (h, r, t) \to \mathbb{R}$. Such a score can then be used for ranking missing triples according to the likelihood that the corresponding facts hold true, and several models have been proposed based on different assumptions (Bordes, Usunier, García-Durán, et al., 2013; Y. Lin, Z. Liu, M. Sun, et al., 2015; Nickel, Rosasco, and Poggio, 2016; Trouillon, Welbl, Riedel, et al., 2016).

Triple classification. The *triple classification* task consists in verifying whether an unseen triple (h, r, t) holds true (is correct) or not. It was first introduced by Socher, D. Chen, Manning, et al. (2013). This task can also be seen as a kind of completion of a knowledge graph, as such it has been studied in Socher, D. Chen, Manning, et al. (2013), Z. Wang, Jianwen Zhang, J. Feng, et al. (2014b), and Y. Lin, Z. Liu, M. Sun, et al. (2015).

Since most models output a score for each triple, which is expected to be higher for true triples, it is a common practice to introduce thresholds δ_r for each relation r to set the truth boundaries. Thus, any unseen triple, say (h, r, t), will be predicted as true if its score is higher than δ_r, and inexistent (false, but there is no negation in RDF due to OWA) otherwise. More technically, the relation-specific thresholds δ_r can be determined separately according to (maximising) the classification accuracy over a small sample of observed triples for that relation, extracted from the knowledge graph or given in a separate validation set. Usually, a different classifier for triples is generated for each relation.

Entity resolution. The *entity resolution* task consists on determining whether two entities refer to the same object. In some knowledge graphs, it is assumed that there are many nodes which actually refer to identical objects but have different labels (e.g., an author's name can be written in different ways). This task is also known as record linkage, object identification, instance matching, and de-duplication (Nickel, K. Murphy, Tresp, et al., 2016).

Bordes, Glorot, Weston, et al. (2014) consider a setting for this problem, where the knowledge graph contains equivalence relations between entities. For instance, it is stated that *William J. Smith* is the same person as *Smith, W. J.* using a triple (*William J. Smith*, *equalsTo*, *Smith, W. J.*). This is encoded by the `owl:sameAs` relation in the Semantic Web. Therefore, the entity resolution problem can be cast as a triple classification and how likely a triple (h, *equalsTo*, t) is. Triple scores can be then used to perform this prediction. However, knowledge graphs do not always encode the *equalsTo* relation as facts. Thus, a solution using solely the entity representations (vectors) was proposed by Nickel, Tresp, and Kriegel (2011). Given two entities x, y and their vector representations $\boldsymbol{x}$, $\boldsymbol{y}$, the similarity between x and y is computed as $k(x, y) = e^{-\|\boldsymbol{x}-\boldsymbol{y}\|_2^2/\sigma}$, and used to measure the likelihood that x and y refer to the same entity.

Entity classification. The third task is entity classification. The *entity classification* task aims to categorise entities into different semantic categories. For instance, *NUI Galway* is an Organisation, but also a University, and *Emma* is a Person. Generally, this relationship is encoded by the *isA* relation, and by the `rdf:type` relation in Semantic Web knowledge bases. Unlike the *equalsTo* relationship, the *isA* relationship can be found more frequently in knowledge bases. Thus, the classification problem can be treated as a specific link prediction task,

answering the questions (x, *isA*, *?*). Entity classification has been studied in Nickel, Tresp, and Kriegel (2011) and Nickel, Tresp, and Kriegel (2012), where AUC-PR was used as evaluation metric.

3.3.3 Statistical properties for completion

Most completion methods over knowledge graphs make some assumptions based on statistical properties to simplify their task. Knowledge graphs typically adhere to some deterministic rules, e.g., type constraints when saying that *Róisín* is a Researcher, or transitivity, e.g., if we know that *Aoife* was born in *Galway* and *Galway* is located in *Ireland*, then we know that *Aoife* was born in *Ireland*. Such assumptions also define how the (prediction) results should be interpreted.

Homophily. *Homophily* or "love of the same" is the tendency of individuals to associate with "similar" ones. McPherson, Smith-Lovin, and J. M. Cook (2001) tracked back the homophily principle and found that it has been first mentioned in Aristotle's *Rhetoric* and *Nichomachean Ethics* manuscripts, where he noted that people "love those who are like themselves", and in *Phaedrus*, Plato observed that "similarity begets friendship." For instance, if we refer to people, this principle suggests that people are most likely to make friends with others of similar demographic characteristics. The term homophily has also been referred to as autocorrelation (D. D. Jensen and Neville, 2002) when dealing with multi-relational data, i.e., graphs with more than one type of link.

In their seminal paper, McPherson, Smith-Lovin, and J. M. Cook (2001) study homophily in social networks, an environment where it has been strongly observed and where friends will usually share common interests such as movies, work, games, hobbies. This principle also says that related individuals influence each other, which has been studies in social networks (McPherson, Smith-Lovin, and J. M. Cook, 2001). For instance, if a cinephile person likes the 2006 movie *Dreamgirls* it is very probable that she will suggest it to friends, and since her friends (very likely) have a similar taste, they will also like the movie. Bischoff (2012) was able to predict real-life friendships by examining online interactions in the social music platform *Last.fm*, analysing similarities of taste as well as demographic attributes and geographical locations.

Global and long-range statistical dependencies. These are dependencies that can span paths in a knowledge graph, involving several triples and types of relationships (Nickel, K. Murphy, Tresp, et al., 2016). For instance, the Irish citizenship of *Aoife* depends statistically on the city where she was born (Galway), which involves the triples: (i) (*Aoife*, *bornIn*, *Galway*), (ii) (*Galway*, *locatedIn*, *Ireland*), and (iii) (*Aoife*, *citizenOf*, *Ireland*). Relational learning has the capability to learn better representations by exploiting such patterns, yielding more accurate models. In Mohamed, Muñoz, Nováček, et al. (2017), we investigate the equivalence between relation paths. Two paths are called equivalent if they have the same path extension, given by the nodes that can be reached following the relation path. For example,

the following relation paths are equivalent: $\langle livesIn \rangle \equiv \langle worksIn, locatedIn \rangle$. Path equivalence resembles the topological notion of *homotopy*, where the entities and relations in the middle of a path can be different but the start and end nodes must be the same. For example, we can say that the following paths are homotopic or equivalent: $\langle Aoife, hasPartner, Eoin \rangle \equiv \langle Aoife, hasChild, Peter, hasChild^{-1}, Eoin \rangle$.

3.4 Statistical relational learning

Statistical relational learning (SRL) is concerned with the creation of statistical models for relational data (Getoor and Taskar, 2007). Recently, SRL has been applied to knowledge graphs with success, delivering new state-of-the-art results in several tasks (Nickel, K. Murphy, Tresp, et al., 2016; Q. Wang, Mao, B. Wang, et al., 2017). For the completion task, it is required to answer the question whether a triple (h, r, t) is true or false. This is a canonical relational machine learning task, where the problem is to find a function ϕ_r such that for every *possible* triple (h, r, t):

$$p((h, r, t) = 1) = \phi_r(\boldsymbol{z}),$$

where a classifier (function ϕ) is built for each relation type r. It is unknown at this point, however, what is the input $\boldsymbol{z}$. First, we can assume $\boldsymbol{z}$ to be a vector of known features for both subject and object entities, where the features can be derived from a neighbourhood of the entities in the knowledge graph. For instance, we can have the k-dimensional vectors $[a_{i1}, a_{i2}, \ldots, a_{ik}]^\top$ and $[a_{j1}, a_{j2}, \ldots, a_{jk}]^\top$ assigned to entities h and t, respectively, and build the input $\boldsymbol{z}$ using concatenation: $\boldsymbol{z} = [a_{i1}, a_{i2}, \ldots, a_{ik}, a_{j1}, a_{j2}, \ldots, a_{jk}]^\top$. Classifiers such as fixed basis functions, kernels or neural networks could be applied over the input z to answer the initial question. Second, one could treat the features as latent (unknown) variables, where k-dimensional vectors are build for each entity. Latent features in graphs usually are a kind of collective learning, where information can globally be propagated in a network of random variables. For instance, in a propagation one can learn that *book A* is a likely best selling because the author's publisher is known to have many best sellers. In this section, we review statistical relational learning approaches that have become very popular in the recent years due to their ability to handle noisy, inconsistent, incomplete, and uncertain data.

Given the wide spectrum of tasks and knowledge graphs of different domain, size, level of incompleteness, there is no such one-fits-all solution. Initial approaches considered simple structural patterns such as paths and sub-graphs, which are known as *graph-based feature approaches*. Rettinger, Lösch, Tresp, et al. (2012) studied how graph-based features can be extracted from Semantic Web knowledge bases for several machine learning tasks. In the past years, more tasks (Section 3.3.2) and techniques have appeared in the domain of knowledge graphs and deep learning. The latter are known as *latent feature approaches* and can

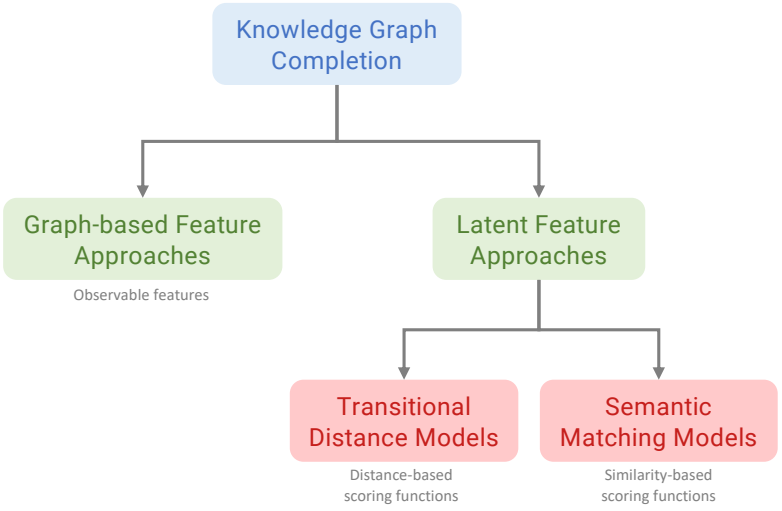


Figure 3.6: Classification of knowledge graph completion approaches.

be roughly classified into two groups: transitional distance models and semantic matching models (Q. Wang, Mao, B. Wang, et al., 2017). Figure 3.6 shows a classification of the approaches that we describe in the following.

3.4.1 Graph-based feature approaches

Graph-based feature approaches assume that links between entities could be predicted by features extracted from the observed links in the knowledge graph. For example, we can assume that if you have the triples (*Mark*, *livesIn*, *Vienna*) and (*Greta*, *marriedTo*, *Mark*), then since usually couples live in the same city we could predict that (*Greta*, *livesIn*, *Vienna*) as well. The predictions made by approaches in this category are easy to explain since the features could be directly mapped to observed triples (links or edges in the graph). This is a huge difference when compared with latent feature models that we will review in Section 3.4.2.

One set of approaches in this classification are rule mining approaches that we already reviewed in Section 3.2.2. Rules extracted from graph data can be used to infer new links. Works such as AMIE (L. A. Galárraga, Teflioudi, Hose, et al., 2013; L. Galárraga, Teflioudi, Hose, et al., 2015) can be used to extract rules like

$$livesIn(h, p) \wedge marriedTo(h, w) \Rightarrow livesIn(w, p),$$

which can be used to infer missing *livesIn* relationships between married people in the knowledge graph. Similarly, logic oriented approaches proposed in the Semantic Web (Lehmann, 2009; d'Amato, Fanizzi, and Esposito, 2010; Lisi, 2010) can be applied to learn logic rules from instance data. Additionally, in the Semantic Web, several approaches

use graph-based features with machine learning for tasks such as class prediction or link prediction (Rettinger, Lösch, Tresp, et al., 2012; Ristoski and Paulheim, 2016a). Lösch, Bloehdorn, and Rettinger (2012) studied two families of graph kernels obtaining competitive results compared with Support Vector Machines (SVMs) in the tasks of entity classification and link prediction. A deeper study of graph kernels was done by Vries and Rooij (2015), where kernels that count subtrees outperform all the studies kernels in classification tasks. An interesting contribution by de Vries and de Rooji is the adaptation of Weisfeiler-Lehman graph kernel (Shervashidze, Schweitzer, Leeuwen, et al., 2011) for subtree counting kernels in RDF knowledge graphs.

More recent works have tried to mix graph-based features with representation learning. In Ristoski and Paulheim (2016a), the Weisfeiler-Lehman graph kernel algorithm proposed in Vries and Rooij (2015) and graph walks are fed to Word2Vec—a method to learning word embeddings. Weisfeiler-Lehman graph kernel and graph walks are used to build sequences that are then passed to Word2Vec for learning embeddings of entities and relations appearing in the sequences. Because the embedding approach was originally created for words, RDF2Vec (Ristoski and Paulheim, 2016a) does not consider the directionality of relations or the complex structure of the graph (limiting to local information only). Other word embedding techniques like Global Vectors (GloVe) (Pennington, Socher, and Manning, 2014) were also tested with the sequences generated in RDF2Vec (Cochez, Ristoski, Ponzetto, et al., 2017) to consider global patterns for creating the vector space.

The Path Ranking Algorithm (PRA) (Lao and Cohen, 2010; Lao, Mitchell, and Cohen, 2011) uses directed paths generated by random walks of bounded length in the graph to predict links in multi-relational knowledge graphs. The paths connecting two entities are used to compute path probabilities that support a given relation. For example, the relation *teamHomeStadium* is supported by the path $c \xrightarrow{teamPlaysInCity} c \xrightarrow{cityStadiums} c$ (where c represents a concept), which can be read as "the stadiums located in the same city with the query team." PRA shows significant improvements over NELL compared with a previous rule-based (Horn-clause) inference approach (Lao, Mitchell, and Cohen, 2011). Extensions to PRA consider sub-graphs and connecting paths. Gardner and Mitchell (2015) show that the path probabilities computed in PRA are not really needed and propose a simpler sub-graph feature approach that allows to obtain a richer set of features. PRA and SFE are considered single-task approaches and a multi-task learning PRA, referred to as coupled PRA or CPRA, is proposed in Q. Wang, J. Liu, Y. Luo, et al. (2016). The main idea of CPRA is to identify highly correlated relationships using agglomerative clustering and couple the prediction of such relations using multi-task learning. Recently, Mohamed, Nováček, and Vandenbussche (2018) builds on top of SFE and PRA and proposes an approach that deals with cases where there are non-connected sub-graph paths by extracting so-called distinct sub-graph paths as features.

Although graph-based feature approaches have shown to outperform previous ILP methods (e.g., FOIL (Quinlan, 1990)) for link prediction, recent research has been focused mainly on latent feature approaches that learn distributed representations of entities and

relations in the knowledge graph. As observed in (Toutanova and D. Chen, 2015; Nickel, K. Murphy, Tresp, et al., 2016), graph-based features such as PRA paths and latent features are often complementary for the knowledge graph completion problem.

3.4.2 Latent feature approaches

Traditional statistical learning such as Markov-logic networks (Richardson and Domingos, 2006) suffer from scalability issues when dealing with large networks. Similar problem is attributed to graph-based feature when the length of the sub-graphs or random walks is large yielding a large number of features. This has been addressed by methods that embed multi-relational data into low-dimensional representations of entities and relations. Latent feature approaches are also known as knowledge graph embedding in literature, and their key idea is to embed knowledge graph elements (i.e., entities and relations) into a continuous vector space that allows algebraic operations while preserving the structure of the knowledge graph. These approaches have been successfully applied in natural language processing (NLP) tasks like semantic parsing (Berant, Chou, Frostig, et al., 2013) and named entity disambiguation (Hakimov, Oto, and Dogdu, 2012), question answering (Bordes, Chopra, and Weston, 2014; Bordes, Weston, and Usunier, 2014), recommender systems (Yu, Ren, Y. Sun, et al., 2014; F. Zhang, Yuan, Lian, et al., 2016), among others. We refer the reader to recent survey papers Nickel, K. Murphy, Tresp, et al. (2016) and Q. Wang, Mao, B. Wang, et al. (2017) for a detailed comparison of the approaches. Table 3.1 summarises the scoring functions reviewed in this section and the different constraints imposed by each model (Trouillon, Dance, Gaussier, et al., 2017; Q. Wang, Mao, B. Wang, et al., 2017).

Latent feature approaches are also known as *neural link predictors* because they can be interpreted as simple multi-layer neural networks consisting of an *encoding layer* and a *scoring layer*. Given a triple (h, r, t), the encoding layer maps entities $h, t \in \mathcal{E}$ to their k-dimensional distributed representations $\boldsymbol{h}$ and $\boldsymbol{t}$. Then, the scoring layer computes the likelihood of the triple based on a relation-dependent function ϕ_r. Henceforth, the scoring function ϕ is defined as $\phi(h, r, t) = \phi_r(h, t)$, where $\phi_r \colon \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$ that obtains the head and tail embeddings $\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^k$ based on the relation $r \in \mathcal{R}$.

A neural link predictor with parameters Θ defines a conditional probability distribution over the truth value of a triple (h, r, t) (Nickel, K. Murphy, Tresp, et al., 2016):

$$\Pr(y_{hrt} = 1 \mid \Theta) = \sigma(\phi_r(h, t)), \tag{3.1}$$

where $y_{hrt} \in \{0, 1\}$ is the truth label of the triple, $\Theta = \{\boldsymbol{e}_i\}_{i=1}^{N_e} \cup \{\boldsymbol{r}_j\}_{j=1}^{N_r}$ denotes the set of all entity and relation embeddings, $\sigma(x) = 1/(1 + \exp(-x))$ is the standard logistic function, and ϕ_r denotes the model's scoring function (cf. Table 3.1). Most models consider the k-dimensional embeddings as real-valued $\boldsymbol{h}, \boldsymbol{t}, \boldsymbol{r} \in \mathbb{R}^k$ with the exception of ComplEx (Trouil-

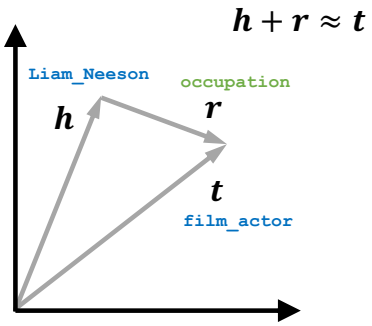


Figure 3.7: Illustration of a fact according to the interpretation of TransE.

lon, Welbl, Riedel, et al., 2016), where $\boldsymbol{h}, \boldsymbol{t}, \boldsymbol{r} \in \mathbb{C}^k$.

Translational distance models. These models exploit distance-based scoring functions. The score of a triple is usually given by the distance between the two entities (subject and object) after a translation promoted by the relation. The most representative model of this class is TransE (Bordes, Usunier, García-Durán, et al., 2013). TransE represents both entities and relations in the same vector space, say $\mathbb{R}^d$. Given a triple (h, r, t), a relation specific latent feature vector indicates a translation between the embedded entities $\boldsymbol{h}$ and $\boldsymbol{t}$, such that $\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$, when (h, r, t) is true. Figure 3.7 illustrates the translation proposed in TransE to model true facts in the knowledge graph. TransE model is inspired by the results in Mikolov, K. Chen, Corrado, et al. (2013), where the authors propose continuous vector representations for words in what is now known as Word2Vec. Mikolov et al. showed that some linguistic regularities between words could be computed by their vector difference in the embedding space, $\boldsymbol{w}_{King} - \boldsymbol{w}_{Man} + \boldsymbol{w}_{Woman} \approx \boldsymbol{w}_{Queen}$ is an example. In particular, a triple (h, r, t) is scored using the function:

$$\phi_r(h, t) = -\|\boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t}\|_\rho,$$

where $\rho \in \{1, 2\}$ indicates the norm, and the score $\phi_r(h, t)$ is expected to be large if (h, r, t) holds.

A simplified version of TranE is the Unstructured model (UM) (Bordes, Glorot, Weston, et al., 2012) that do not takes the relation into account, i.e., $\boldsymbol{r} = \boldsymbol{0}$. In UM, the scoring function is defined as:

$$\phi_r(h, t) = -\|\boldsymbol{h} - \boldsymbol{t}\|_2^2.$$

This model clearly cannot distinguish between relations. Another work by the same authors, proposed Structured embedding (SE) (Bordes, Weston, Collobert, et al., 2011), where two matrices are used to represent the relation r when projected against the head or tail:

$$\phi_r(h, t) = -\|\boldsymbol{M}_r^1 \boldsymbol{h} - \boldsymbol{M}_r^2 \boldsymbol{t}\|_1.$$

Finally, several other extensions to TransE have been proposed to overcome the problem dealing with 1-to-N, N-to-N, and N-to-N relations. These new models include TransH (Z.

Wang, Jianwen Zhang, J. Feng, et al., 2014b) that adds relation-specific hyperplanes for the embeddings, TransR (Y. Lin, Z. Liu, M. Sun, et al., 2015) that introduces relation-specific spaces, among others.

Semantic matching models. On the other hand, semantic matching models exploit similarity-based scoring functions to score triples. The first model we review is RESCAL (Nickel, Tresp, and Kriegel, 2011, 2012), a.k.a. bilinear model (Jenatton, Roux, Bordes, et al., 2012). RESCAL explains triples via pairwise interactions of latent features, and its score function is defined as:

$$\phi_r(h,t) = \boldsymbol{h}^\top \boldsymbol{M}_r \boldsymbol{t} = \sum_{i=0}^{d-1}\sum_{j=0}^{d-1}[\boldsymbol{M}_r]_{ij}\cdot[\boldsymbol{h}]_i\cdot[\boldsymbol{t}]_j,$$

where $[\boldsymbol{x}]_i$ denotes the i-th entry of the vector, $[\boldsymbol{M}]_{ij}$ the ij-th entry of a matrix, $\boldsymbol{h},\boldsymbol{t}\in\mathbb{R}^d$ are vector representations of the subject and object entities, and $\boldsymbol{M}_r\in\mathbb{R}^{d\times d}$ is a weight matrix associated with the relation defined as a set of rank-1 matrices, $\boldsymbol{M}_r=\sum_i \pi_r^i \boldsymbol{u}_i\boldsymbol{v}_i^\top$. Because RESCAL's scoring function computes the pairwise interactions between all elements of $\boldsymbol{h}$ and $\boldsymbol{t}$, it requires $\mathcal{O}(d^2)$ parameters per relation, making it very expensive for large knowledge graphs. In García-Durán, Bordes, and Usunier (2014), the authors propose a model that merges two- and three-way interactions. In addition to RESCAL other tensor factorisation models have been proposed for knowledge graphs: CANDECOMP/-PARAFAC decomposition (Carroll and Chang, 1970), Tucker decomposition (Tucker, 1966), and A. P. Singh and Gordon (2008), Drumond, Rendle, and Schmidt-Thieme (2012), and Rendle (2013), among others.

Some of the most popular and scalable neural link predictors are DistMult (B. Yang, Yih, X. He, et al., 2015), ComplEx (Trouillon, Welbl, Riedel, et al., 2016), and HolE (Nickel, Rosasco, and Poggio, 2016). DistMult uses a simplification of RESCAL, where $\boldsymbol{M}_r$ is restricted to diagonal matrices, thus reducing the number of parameters to $\mathcal{O}(d)$. The scoring function of DistMult is defined as:

$$\phi_r(h,t) = \boldsymbol{h}^\top \mathrm{diag}(\boldsymbol{r})\boldsymbol{t} = \sum_{i=0}^{d-1}[\boldsymbol{r}]_i\cdot[\boldsymbol{h}]_i\cdot[\boldsymbol{t}]_i,$$

where $\boldsymbol{r}\in\mathbb{R}^d$, and $\boldsymbol{M}_r=\mathrm{diag}(\boldsymbol{r})$. One of the shortcomings of DistMult is that it can only deal with symmetric relations due to the equality $\boldsymbol{h}^\top\mathrm{diag}(\boldsymbol{r})\boldsymbol{t}=\boldsymbol{t}^\top\mathrm{diag}(\boldsymbol{r})\boldsymbol{h}$. ComplEx proposes to solve DistMult's shortcoming by introducing complex-valued embeddings to better capture the asymmetry of relations. The scoring function of ComplEx is defined as:

$$\phi_r(h,t) = \mathrm{Re}(\boldsymbol{h}^\top\mathrm{diag}(r)\bar{\boldsymbol{t}}) = \mathrm{Re}\left(\sum_{i=0}^{d-1}[\boldsymbol{r}]_i\cdot[\boldsymbol{h}]_i\cdot[\bar{\boldsymbol{t}}]_i\right),$$

where $\bar{\boldsymbol{t}}$ is the conjugate of $\boldsymbol{t}$ and $\mathrm{Re}(\cdot)$ represents the real part of a complex number.

Another model extending RESCAL is Holographic Embeddings (HolE) that uses circu-

lar correlation (computed using fast Fourier transforms):

$$[\boldsymbol{h} \star \boldsymbol{t}]_i = \sum_{k=0}^{d-1} [\boldsymbol{h}]_k \cdot [\boldsymbol{t}]_{(k+i) \bmod d},$$

where $\boldsymbol{h} \star \boldsymbol{t} \in \mathbb{R}^d$. HolE defines a scoring function for a triple as follows:

$$\phi_r(h,t) = \boldsymbol{r}^\top (\boldsymbol{h} \star \boldsymbol{t}) = \sum_{i=0}^{d-1} [\boldsymbol{r}]_i \sum_{k=0}^{d-1} [\boldsymbol{h}]_k \cdot [\boldsymbol{t}]_{(k+i) \bmod d}.$$

As per the not commutativity of the circular correlation, HolE also can model asymmetric relations like RESCAL and ComplEX but using $\mathcal{O}(d)$ parameters. Despite the different formulations of ComplEx and HolE, Hayashi and Shimbo (2017) actually showed the that the scoring functions of these two models are equivalent. This was later corroborated by the original authors of the corresponding models (Trouillon and Nickel, 2017).

Many other models have been proposed to embed entities and relations in knowledge graphs based neural networks. Multi-layer perceptron (MLP) (X. Dong, Gabrilovich, Heitz, et al., 2014) is an approach that uses a simple neural network with one hidden layer. This approach was used in Google's knowledge vault project, and its score function is:

$$\phi_r(h,t) = \boldsymbol{w}^\top \tanh(\boldsymbol{M}^1 \boldsymbol{h} + \boldsymbol{M}^2 \boldsymbol{r} + \boldsymbol{M}^3 \boldsymbol{t}),$$

where $\boldsymbol{M}^1, \boldsymbol{M}^2, \boldsymbol{M}^3 \in \mathbb{R}^{d \times d}$ are the first layer weights, and $\boldsymbol{w} \in \mathbb{R}^d$ the second layer weights, shared across different relations. Neural association model (NAM) (Q. Liu, Jiang, Ling, et al., 2016) is the last model we review, which uses a deep architecture. For a triple (h, r, t), the head and relation embeddings are concatenated in the input layer, $\boldsymbol{z}^{(0)} = [\boldsymbol{h}; \boldsymbol{r}] \in \mathbb{R}^{2d}$. The deep architecture is composed of L rectified linear hidden layers fed with $\boldsymbol{z}^{(0)}$:

$$\begin{aligned} \boldsymbol{a}^{(\ell)} &= \boldsymbol{M}^{(\ell)} \boldsymbol{z}^{(\ell-1)} + \boldsymbol{b}^{(\ell)}, \ \ell = 1, \ldots, L, \\ \boldsymbol{z}^{(\ell)} &= \mathrm{ReLU}(\boldsymbol{a}^{(\ell)}), \ \ell = 1, \ldots, L, \end{aligned}$$

where $\boldsymbol{M}^{(\ell)}$ and $\boldsymbol{b}^{(\ell)}$ are the weight matrix and bias for the ℓ-th layer, respectively. The score in NAM is given by multiplying the output of the last hidden layer with the embedding of the tail entity as follows:

$$\phi_r(h,t) = \boldsymbol{t}^\top \boldsymbol{z}^{(L)}.$$

3.4.3 Model training and negatives generation

A latent feature model is trained by minimising a loss function defined over a target graph $\mathcal{G}$ using stochastic gradient descent (SGD) (cf. Algorithm 1). The open world assumption states that knowledge graphs only contain positive examples (i.e. facts); however, negative

examples are also needed to train any of the models previously presented. A solution—motivated by the Local Closed World Assumption (LCWA) (X. Dong, Gabrilovich, Heitz, et al., 2014)—is to generate negative examples by *corrupting* the triples in the graph (Rendle, Freudenthaler, Gantner, et al., 2009; Bordes, Usunier, García-Durán, et al., 2013; Nickel, K. Murphy, Tresp, et al., 2016). Given a positive triple $(h, r, t) \in \mathcal{G}$, corrupted triples (negative examples) can be generated by replacing either the head or tail with a random entity sampled uniformly from $\mathcal{E}$ (Bordes, Weston, Collobert, et al., 2011). More formally, negative examples for a triple are generated by a function:

$$\mathcal{C}(h, r, t) = \{(h', r, t) \mid h' \in \mathcal{E}\} \cup \{(h, r, t') \mid t' \in \mathcal{E}\}. \tag{3.2}$$

Let $\mathcal{D}^+$ be the set of positive examples, and $\mathcal{D}^-$ the set of negatives generated accordingly with function $\mathcal{C}$. The training consists of learning the embeddings of entities and relations (parameters Θ) that best explain $\mathcal{D}^+$ and $\mathcal{D}^-$ according to Equation (3.1). For that, models such as TransE (Bordes, Usunier, García-Durán, et al., 2013), DistMult (B. Yang, Yih, X. He, et al., 2015) and HolE (Nickel, Rosasco, and Poggio, 2016) minimise a pairwise ranking loss:

$$\mathcal{L}_{\text{hinge}} = \min_{\Theta} \sum_{\tau^+ \in \mathcal{D}^+} \sum_{\tau^- \in \mathcal{D}^-} \left[\gamma - \sigma(\phi_r(h, t)) + \sigma(\phi_r(h', t'))\right]_+, \tag{3.3}$$

where $\tau^+ = (h, r, t)$ is a positive example, $\tau^- = (h', r, t')$ a negative one, $[x]_+ = \max(0, x)$, and γ is the margin hyperparameter separating positives from negatives. This loss make the scores of positive triples higher than those of negative ones. The entity embeddings are also constrained to unit norm: $\forall \boldsymbol{h}, \boldsymbol{t} \in \mathcal{E} : \|\boldsymbol{h}\|_2 = 1, \|\boldsymbol{t}\|_2 = 1$. Whereas other models like ComplEx (Trouillon, Welbl, Riedel, et al., 2016) minimise the logistic loss:

$$\mathcal{L}_{\text{logistic}} = \min_{\Theta} \sum_{\tau \in \mathcal{D}^+ \cup \mathcal{D}^-} \log(1 + \exp(-y_\tau \cdot \phi_r(h, t))) \tag{3.4}$$

where $\tau = (h, r, t)$ is a training example in $\mathcal{D}^+ \cup \mathcal{D}^-$ (i.e. triple), and $y_\tau \in \{-1, 1\}$ is the label (negative or positive) associated with the example.

It worth pointing out that by minimising the pairwise ranking loss we do not assume that negative examples are necessarily false, just that they are more invalid than positive ones (Nickel, K. Murphy, Tresp, et al., 2016). Trouillon, Welbl, Riedel, et al. (2016) also showed that the logistic loss generally yields better results for compositional models such as DistMult or ComplEx, whilst the pairwise ranking loss is more suitable for translational models like TransE. In Mohamed, Nováček, Vandenbussche, and Muñoz (2019), we studied the behaviour of different latent feature models over WordNet, Freebase, and NELL datasets, when using different pointwise and pairwise versions of the loss functions reviewed. Our experiments validate that the selection of a loss function does have a considerable impact in the performance of knowledge graph embedding models. Moreover, we observed that there are strong relationships between the loss functions and evaluation metrics,

Algorithm 1 Learning the model parameters Θ via Projected SGD

Input: $\mathcal{D}^+$, epochs τ, initial learning rate $\eta \in \mathbb{R}$
Output: Optimal model parameters Θ

```
Initialise embeddings e and r
for i = 1, ..., τ do
    e ← e/||e||, ∀e ∈ E
    g_i ← ∇L(Θ)                    ◁ Compute the gradient of the loss function L on examples
    Θ ← Θ − η_i g_i                ◁ Update the model parameters via gradient descent
    Apply additional constraints or regularisation terms
end for
return Θ
```

which can assist practitioners when choosing a loss. This is an improvement over state-of-the-art approaches where loss functions have been selected in a rather non-systematic way.

The optimisation of Equations (3.3) and (3.4) can be done by stochastic gradient descent (SGD) (Robbins and Monro, 1951) in mini-batch mode as in Algorithm 1. In each iteration, a subset of positive triples is sampled from $\mathcal{D}^+$ and the negative examples are generated accordingly for each positive. Both positives and negatives are used during training to compute the loss function in each mini-batch. After the mini-batch, the embeddings are updated via gradient descent with constant or adaptive learning rates. In knowledge graph embeddings, AdaGrad (Duchi, Hazan, and Singer, 2011) is commonly used to tune the learning rate.

3.5 Summary

In this chapter, we have presented the state of the art for schema languages (Section 3.1), schema inference (Section 3.2), and knowledge graph completion approaches (Section 3.3). Because of the dynamic schema of most knowledge graphs, several schema inference and schema validation approaches have been proposed to provide certain consistency assurances to data consumers. However, validation is still an open problem and there are no scalable (or approximated) solutions able to deal with noisy input data (Labra Gayo, García-González, Fernández-Alvarez, et al., 2019). We provide a definition and describe existing tasks in the completion of knowledge graphs. Building upon existing work from databases, we pointed out the relevance of cardinality to expose the inherent structure of knowledge graphs, and how they can be used to give a sense of the completeness of relationships. Because cardinality is typically not found in knowledge graphs, in Chapter 4, we will introduce an approach that mines cardinality constraints from instance data—such cardinalities can be easily encoded in schema languages.

We reviewed completion approaches based on statistical relational learning that can scale to fairly large knowledge graphs (Section 3.4). Completion approaches can be roughly divided between graph-based feature and latent feature models. There are no conclusive ex-

Table 3.1: Summary of latent feature models with their scoring functions and constraints.

Model	Ent. Embedding	Rel. Embedding	Scoring Function	Constraints/Regularisation
SE (2011)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{M}_r^1, \boldsymbol{M}_r^2 \in \mathbb{R}^{d \times d}$	$-\|\boldsymbol{M}_r^1 \boldsymbol{h} - \boldsymbol{M}_r^2 \boldsymbol{t}\|_1$	$\|\boldsymbol{h}\|_2 = 1, \|\boldsymbol{t}\|_2 = 1$
RESCAL (2011)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{M}_r \in \mathbb{R}^{d \times d}$	$\boldsymbol{h}^\top \boldsymbol{M}_r \boldsymbol{t}$	$\|\boldsymbol{h}\|_2 \leq 1, \|\boldsymbol{t}\|_2 \leq 1, \|\boldsymbol{M}_r\|_F \leq 1, \boldsymbol{M}_r \in \mathbb{R}^{k \times k}$
UM (2012)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	—	$-\|\boldsymbol{h} - \boldsymbol{t}\|_2^2$	$\|\boldsymbol{h}\|_2 = 1, \|\boldsymbol{t}\|_2 = 1$
TransE (2013)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{r} \in \mathbb{R}^d$	$-\|\boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t}\|_\rho$	$\|\boldsymbol{h}\|_2 = 1 \|\boldsymbol{t}\|_2 = 1, \rho \in \{1, 2\}$
NTN (2013)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{r}, \boldsymbol{b}_r \in \mathbb{R}^k$, $\boldsymbol{M}_r \in \mathbb{R}^{d \times d \times k}$, $\boldsymbol{M}_r^1, \boldsymbol{M}_r^2 \in \mathbb{R}^{k \times d}$	$\boldsymbol{r}^\top \tanh(\boldsymbol{h}^\top \boldsymbol{M}_r \boldsymbol{t} + \boldsymbol{M}_r^1 \boldsymbol{h} + \boldsymbol{M}_r^2 \boldsymbol{t} + \boldsymbol{b}_r)$	$\|\boldsymbol{h}\|_2 \leq 1, \|\boldsymbol{t}\|_2 \leq 1, \|\boldsymbol{r}\|_2 \leq 1, \|\boldsymbol{b}_r\|_2 \leq 1, \left\|\boldsymbol{M}_r^{[:,:,i]}\right\|_F \leq 1, \|\boldsymbol{M}_r^1\|_F \leq 1, \|\boldsymbol{M}_r^2\|_F \leq 1$
MLP (2014)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{r} \in \mathbb{R}^d$	$\boldsymbol{w}^\top \tanh\left(\boldsymbol{M}^1 \boldsymbol{t} + \boldsymbol{M}^2 \boldsymbol{r} + \boldsymbol{M}^3 \boldsymbol{t}\right)$	$\|\boldsymbol{h}\|_2 \leq 1, \|\boldsymbol{t}\|_2 \leq 1, \|\boldsymbol{r}\|_2 \leq 1$
DistMult (2015)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{r} \in \mathbb{R}^d$	$\boldsymbol{h}\, \mathrm{diag}(\boldsymbol{r}) \boldsymbol{t}$	$\|\boldsymbol{h}\|_2 \leq 1, \|\boldsymbol{t}\|_2 \leq 1, \|\boldsymbol{r}\|_2 \leq 1$
HolE (2016)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{r} \in \mathbb{R}^d$	$\boldsymbol{r}^\top (\boldsymbol{h} \star \boldsymbol{t})$	$\|\boldsymbol{h}\|_2 \leq 1, \|\boldsymbol{t}\|_2 \leq 1, \|\boldsymbol{r}\|_2 \leq 1$
ComplEx (2016)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{C}^d$	$\boldsymbol{r} \in \mathbb{C}^d$	$\mathrm{Re}(\boldsymbol{h}^\top\, \mathrm{diag}(\boldsymbol{r}) \bar{\boldsymbol{t}})$	$\|\boldsymbol{h}\|_2 \leq 1, \|\boldsymbol{t}\|_2 \leq 1, \|\boldsymbol{r}\|_2 \leq 1$
NAM (2016)	$\boldsymbol{h}, \boldsymbol{t} \in \mathbb{R}^d$	$\boldsymbol{r} \in \mathbb{R}^d$	$\phi_r(h, t) = \boldsymbol{t}^\top \boldsymbol{z}^{(L)}$ $\boldsymbol{z}^{(\ell)} = \mathrm{ReLU}(\boldsymbol{a}^{(\ell)}), \boldsymbol{a}^{(\ell)} = \boldsymbol{M}^{(\ell)} \boldsymbol{z}^{(\ell-1)} + \boldsymbol{b}^{(\ell)}$ $\boldsymbol{z}^{(0)} = [\boldsymbol{h}; \boldsymbol{r}]$	—

periments that show a superiority of graph-based over latent feature models or vice versa for learning from knowledge graphs (Toutanova and D. Chen, 2015; Nickel, K. Murphy, Tresp, et al., 2016). On the other hand it has been shown that they are often complementary for tackling knowledge graph completion tasks. Therefore, we will use algorithms from both approaches in two different problems. In Chapter 6, we will evaluate how graph-based features can be used to improve the completeness of Biomedical knowledge graphs. And finally, in Chapter 7 we will propose an algorithm to embed cardinality as a background knowledge to guide knowledge graph embeddings to deliver more accurate predictions.

Part II

Latent Shapes in Knowledge Graph

CHAPTER 4

Approaches for Mining Cardinality

Contents

There is an increasing number of knowledge graphs available on the Web, generated in academia and industry alike. In this chapter, we address the problem of lack of structure in these knowledge graphs due to their schema-free nature required for open environments such as the Web. This problem was stated in our RQ (1), and here we address it with the use of cardinality constraints (Section 3.2.4) to extract and exhibit the structure of knowledge graphs. We propose a definition for relation cardinality bounds, and a data-driven method to extract such bounds from instance data in a bottom-up approach. These bounds can be used to unveil the structure that knowledge graphs data naturally exhibit, independently from any ontological information provided. Furthermore, we also show how these bounds can be used to assess two relevant data quality dimensions: consistency and completeness.

4.1 Problem statement

Cardinality (also known as multiplicity) covers a critical aspect of relational data modelling, referring to the number of times an instance of one entity can be related with instances of another entity. A *cardinality bound* is a restriction on the number of elements in the relation. In particular, knowledge graphs have relationships between entities (of a given type) connected by property types, and we want to specify bounds for such relationships. For example, we would like to express that a drug has only one molecular formula, but can be associated to a finite set (of known or unknown size, the latter denoted as ∞) of adverse drug reactions.

In practice, most knowledge graphs use multiple—sometimes overlapping—vocabularies (Polleres, Scharffe, and Schindlauer, 2007; Vandenbussche, Atemezing, Poveda-Villalón, et al., 2017) (e.g., SKOS, FOAF, DCAT) and avoid to include domain, range or cardinality restrictions because of the contradictions they can generate (Glimm, Hogan, Krötzsch, et al., 2012; Rivero, Hernández, Ruiz, et al., 2012). To illustrate this, let us consider a knowledge graph about countries with two different properties: `gov:hasPopulation` and `dbpedia-owl:populationTotal`.[1] These properties come from different ontologies/vocabularies and represent the same thing, i.e., population of a country; however, their semantics might not be the same or the expected domain/range values could be different. Similarly, two or more labels can be used to refer to the same entity: e.g., `http://www.geonames.org/2963597/ireland.html` and `http://dbpedia.org/resource/Ireland`. If one of these ontologies/vocabularies defines a constraint over the resources, this constraint might contradict the definition(s) made somewhere else producing an inconsistency and harming the reasoning capabilities over the data.

On the other hand, the lack of a central schema can cause a series of difficulties in the reusability of such data (Lausen, Meier, and Schmidt, 2008; Motik, Horrocks, and Sattler, 2009; Bosch and Eckert, 2015; Muñoz, 2016), where applications might need to rely on the fact that data satisfy a set of constraints. A schema indicates what are valid relations for an entity according to its type, what are the allowed values for the properties, and other constraints that instance data should satisfy. For instance, let us consider a software developer building a graphical user interface (GUI) that displays information about countries to end users. For that the developer must query the knowledge graph, in other words, she must write queries using the SPARQL query language to fill every property of every country in the GUI. However, when building the query to retrieve the population of Ireland she finds that, unexpectedly, the knowledge graph contains two mismatching population numbers for that country. These properties come from different vocabularies and their values, i.e., the populations, do not match. This inconsistency could have been avoided if she knew that some countries have more than one value for the relation population. Such situations—sometimes rare, sometimes very abundant—show a gap between the expected

[1] Henceforth, we use prefixes to replace namespaces according to `http://prefix.cc/` to shorten the length of URLs.

and real structure of knowledge graphs. Data users and knowledge engineers would benefit from having an understanding of what information is available to write queries, and to reuse or manage KBs (Neumann and Moerkotte, 2011; Schmidt and Lausen, 2013).

We deal with the problem of identifying relation cardinality bounds such as "a person has two parents" or "a book has minimum one author and maximum eleven". We call this problem the *relation cardinality mining problem* that we define as follows:

Relation cardinality mining problem

Input: a knowledge base $\mathcal{G}$ and optional context (entity type) τ.

Output: a set Σ of relation cardinality bounds that are satisfied by $\mathcal{G}$.

It is important to notice that unlike traditional databases, RDF and OWL assume the open-world semantics (OWA), and absence of the unique name assumption (nUNA). This makes the problem of extracting relation cardinality more complex than a naïve application of SPARQL queries using the `COUNT` operator. Take as example the constraint "a person must have two parents": if the data contain an entity of `Person` type with only one parent, *this does not cause a logical inconsistency, it just means it is incomplete, and in RDF/OWL incomplete is different from inconsistent*. To deal with these specifics, we propose a method which tackles two important challenges:

(1) *KG equality normalisation*, meaning that we must deal with `owl:sameAs` (or equivalent) axioms representing equality between entities to discover accurate cardinalities, and
(2) *outliers filtering*, where we should account for the probability of noise in the data, in order to discover robust cardinalities.

By doing so, the result of this work can provide users with 'shapes' of data that serve them to analyse completeness and consistency, and thus, contribute towards higher levels of quality in knowledge graphs (Hogan, Harth, Passant, et al., 2010; Schmidt and Lausen, 2013; Paulheim, 2017).

4.2 Notion of cardinality bounds

Different languages have their own notion of cardinality (see Section 3.2.4). Here, we propose a definition of relation cardinality for knowledge graphs that generalises the semantics of previous definitions.

Definition 4.1

The *predicate count* of r with respect to h, denoted $count(r, h)$, is defined as the number of triples in $\mathcal{G}$ with h as subject and r as relation:

$$count(r, h) = |\{(h, r, t) \mid \exists t, (h, r, t) \in \mathcal{G}\}| . \tag{4.1}$$

Considering that the *count* function counts the number of *different* objects appearing with a given subject–relation pair, it is easy to see how the output of the function is directly affected by the rewriting of the knowledge graph $\mathcal{G}$. (We introduced rewriting of knowledge graphs in Section 2.1.2, where sameAs-axioms are used to normalise the knowledge graph on equality.)

To illustrate the definition above, let us consider a knowledge graph about books and say that the book *Foundations of Databases* has three authors (Serge Abiteboul, Richard Hull and Victor Vianu) and one publisher (Addison-Wesley). This can be represented by the constraints $count(author, Foundations\ of\ Databases) = 3$ and $count(publisher, Foundations\ of\ Databases) = 1$.

Definition 4.2

A *relation cardinality bound* in knowledge graphs restricts the number of relation values associated with an entity in a given context. Such context could be a particular entity type or the whole knowledge graph. Formally, a cardinality bound φ is an expression of the form $card(P, \tau) = (min, max)$ where $P \subseteq \mathcal{P}$, τ is a context entity type, and where $min \in \mathbb{N}$ and $max \in \mathbb{N} \cup \{\infty\}$ with $min \leq max$. Here $|P|$ denotes the number of relations in φ, min is called the *lower bound*, and max the *upper bound* of φ. If τ is defined ($\tau \neq \varepsilon$), we say that φ is *qualified*; otherwise we say that φ is *unqualified*.

The semantics of this definition of relation cardinality bounds limits the maximum and minimum counts that a given set of relations can have in a given context—as in SHACL, DSP, ICV and OWL. The lower bound of a cardinality may take on values in $\mathbb{N}$, whilst upper bounds can be ∞ to represent that there is an unknown upper limit. In fact, each RDF constraint language has different default values for the minimum and maximum cardinalities. For instance, ShEx assigns a default cardinality of one to a predicate appearing in a shape without any explicit cardinality. On the other hand, SHACL assumes a lower cardinality of zero and upper of ∞. A constraint with such default values (i.e., zero and ∞) will always be satisfied by the data, thus it may be omitted from some data shapes leaving a gap in the explicitness of the shape and the structure of data. We deal with that gap in Section 4.3, where we introduce an algorithm to mine relation cardinality bounds from knowledge graphs.

An unqualified bound is independent of a type/context, i.e., it holds for a set of relations independent of its context, whereas a qualified bound holds only for a set of relations in combination with subject entities of a same given type. Herein, we focus on qualified constraints given their interestingness and relevance for structural analyses of knowledge graphs.

```
SELECT $this
WHERE {
    $this $PROPERTY ?value .
} GROUP BY $this
HAVING (COUNT(?value) <= $minCount)
```

Figure 4.1: SPARQL 1.1 definition of a minimum cardinality constraint.

Definition 4.3

Consider a knowledge graph $\mathcal{G}$. We say that φ is a cardinality bound in $\mathcal{G}$ or $\mathcal{G}$ satisfies φ for a set of properties $P_\varphi \subseteq \mathcal{P}$, a lower bound min_φ, and upper bound max_φ defined in φ, denoted by $\mathcal{G} \models \varphi$, if

$$\forall h \in (\mathcal{R} \cup \mathcal{B}),\ r \in P_\varphi\ (min_\varphi \leq count(r, h) \leq max_\varphi).$$

If φ is qualified to τ then to satisfy φ, $\mathcal{G}$ also needs to satisfy the condition that $\forall h \in (\mathcal{R} \cup \mathcal{B})\ (h, \texttt{rdf:type}, \tau) \in \mathcal{G}$.

Although the mining approach that we will present in Section 4.3 is able to compute an upper bound cardinality, such limit is uncertain when considering RDF's OWA. A cardinality is an expression (lower, upper) associated to the *observed* cardinality of a relation, which is likely to disagree with the expectation of users. For instance, even when the data show that an entity of type Person has at most two children, this might be wrong when considering other unseen same type instances. More certain cardinality bounds can be mined from reliable or complete knowledge graphs, rarely present on the Web and usually existent within specific domains. Therefore, we refer to relation cardinality bounds as descriptive *patterns* when they are automatically extracted from knowledge graphs, and as *constraints* (prescriptive patterns) when normatively assessed by a user and applied in order to restrict a knowledge graph.

In practice, cardinality bounds can be used to validate knowledge graphs using SPARQL 1.1 queries. For instance, Figure 4.1 shows the SPARQL query with aggregation proposed to validate a lower bound min_φ (`$minCount`). The query represents restrictions on the number of values, `?value` nodes, that the `$this` node may have for the given property. A validation result must be produced if the number of value nodes is more than `$minCount`, thus indicating that the data do not conform to the shape. Likewise, to validate an upper bound (max_φ) restriction for a property, we change the `HAVING` condition to '>=', and return a validation result if the number of values is less than `$maxCount`. Note that SHACL, ShEx, and other constraint languages only allow the definition of one condition at a time per property. Therefore, to validate our cardinalities with multiple properties, one must apply an SPARQL 1.1 query like the one in Figure 4.1 independently for each entity and property pair with a single bound (upper or lower). In Section 4.3 we will show how a single, but much more complex, SPARQL 1.1 query can be used to extract both minimum and maximum bounds at once.

Example 4.1

The following expressions define cardinality bounds for different entity types in different domains. Abusing of notation, we may write $card(p, \tau) = (min, max)$ when the constraint applies to a single relation.

1. $card(\{\texttt{mondial:name}, \texttt{mondial:elevation}\}, \texttt{mondial:Volcano}) = (1, 1)$,
2. $card(\texttt{mondial:hasCity}, \texttt{mondial:Country}) = (1, \infty)$,
3. $card(\texttt{dcterms:contributor}, \texttt{bibo:Book}) = (0, \infty)$,
4. $card(\texttt{dcterms:language}, \texttt{bibo:Book}) = (1, 2)$.
5. $card(\{\texttt{lmdb:editor}, \texttt{lmdb:filmid}\}, \texttt{lmdb:Film}) = (1, 9)$.

As suggested in the previous examples, when the upper bound is unclear we can use ∞ in the cardinality bound to express that uncertainty.

4.3 Cardinality mining algorithm

In this section, we introduce an algorithm for mining relation cardinality patterns from knowledge graphs. We also present two different implementations: one based on SPARQL 1.1 that uses a graph databases approach to normalise and extract cardinalities; and another based on Apache Spark[2] that applies a MapReduce or divide-and-conquer strategy to divide the data and run the normalisation step in parallel.

4.3.1 Algorithm

We present Algorithm 2 as an efficient and domain-agnostic solution to mine accurate and robust cardinality patterns from any knowledge graph. This algorithm is designed to mine qualified cardinalities, i.e., a context type is given; however, it can be easily adapted to mine unqualified cardinalities. From a data quality perspective, it is desirable that the mined relation cardinality bounds (see Section 4.1) are accurate and robust. Algorithm 2 outputs a set of relation cardinality patterns, which are called *accurate* because it considers the semantics of equality axioms (expressed by `owl:sameAs` and equivalent predicates), and *robust* because we perform an outliers detection and filtering over noisy cardinality counts.

Our mining algorithm (see Algorithm 2) has three main steps:

(1) ***KG normalisation:*** represented by the function $normalise : \mathscr{G} \rightarrow \mathscr{G}$, receives an unnormalised knowledge graph (with possibly multiple equal entities) and applies an on-the-fly (in memory) rewriting process to consider the semantics of the relation `owl:sameAs` (or other equivalent relation). We address the normalisation by querying all

[2] http://spark.apache.org/ (version 2.1.0)

Algorithm 2 Extraction of cardinality bounds

Input: a knowledge base $\mathcal{G}$; and a context τ
Output: a set Σ of cardinality bounds

1: $\mathcal{G}' \leftarrow normalise\ (\mathcal{G})$
2: $E \leftarrow \{h \mid (h, \texttt{rdf:type}, \tau) \in \mathcal{G}'\}$
3: $P \leftarrow pred(\mathcal{G}', \tau)$ ◁ *predicates for entities of type* τ
4: **for all** $r \in P$ **do**
5: $\quad \mathcal{G}'' \leftarrow triples(\mathcal{G}', E, r)$ ◁ *triples with entity type* τ *and relation* r
6: $\quad M\ \langle u, v \rangle \leftarrow cardPatterns\ (\mathcal{G}'')$ ◁ *map:* u *is an entity, and* v *a cardinality*
7: $\quad \Theta \leftarrow filterOutliers\ (\text{M})$ ◁ *set of inlier cardinalities*
8: $\quad \Sigma.add(card(\{r\}, \tau) = (min(\Theta),\ max(\Theta)))$
9: **end for**

Table 4.1: An axiomatisation for reduction on equality

$\dfrac{(h, r, t) \wedge (h', r', t') \wedge (h', \texttt{owl:sameAs}, h)}{(h, r, t), (h, r', t')}$ (subject-equality)	$\dfrac{(h, r, t) \wedge (h', r', t') \wedge (t', \texttt{owl:sameAs}, t)}{(h, r, t), (h', r', t)}$ (object-equality)

equal entities and building graphs connecting these nodes, which builds so-called cliques where all equal nodes are connected to each other. The cliques are used to normalise the whole KB with replacements. This step could be considered optional in cases where users want information about unnormalised bounds—at the cost of accuracy.

(2) ***Cardinalities extraction:*** performed by the function $cardPatterns : \mathscr{G} \rightarrow \mathcal{E} \times \mathbb{N}$, it is called to retrieve (entity=cardinality) pairs from the passed set of triples in the context of a given relation. The cardinalities for all relations are stored in a map, which either filtered from noisy values or returned directly to extract the constraints.

(3) ***Outliers filtering:*** represented by the $filterOutliers : \mathcal{E} \times \mathbb{N} \rightarrow \mathbb{N} \cup \emptyset$ function, receives a map of (entity=cardinality) pairs and applies grouping and unsupervised univariate statistical methods to identify and remove noisy or outside of a range values to ensure robustness. It returns an empty set when the (entity=cardinality) pair is filtered out and the cardinality otherwise. Similarly to normalisation, this step could be considered optional at the cost of robustness though.

Next, we present an example for the application of Algorithm 2, and describe each of its part in more details in Sections 4.3.2 to 4.3.4.

Example 4.2

Let us consider a KB with entities ex:s1, ex:s2, and ex:s3, and properties ex:p1 and ex:p2. First, we apply the *normalise* function that replaces entities by one representative element equivalence type induced according to the sameAs-cliques. Then, for each property we extract the (entity, cardinality) pairs using the function $cardPatterns$. For instance, we obtain $\{\texttt{ex:s1} = 1, \texttt{ex:s2} = 1, \texttt{ex:s3} = 2\}$ for property ex:p1, and $\{\texttt{ex:s1} = 3, \texttt{ex:s2} = 25, \texttt{ex:s3} = 3\}$ for property ex:p2. Next, the function $filterOutliers$ tries to identify outliers and determines that there are no outliers for property ex:p1, but that a cardinality of 25 is an outlier for ex:p2. Thus, 25 is removed from the patterns leaving $\{\texttt{ex:s1} = 3, \texttt{ex:s3} = 3\}$ as robust cardinalities for ex:p2. Finally, the cardinality bounds (min, max) are extracted from the remaining inlier cardinalities by using simple min and max functions.

4.3.2 Knowledge graph normalisation: rewriting approaches

Knowledge bases contain different types of axioms, being owl:sameAs and equivalent-semantic relations the most important when computing cardinalities. Regardless of the approach, by not considering these axioms a method loses its accuracy and cannot ensure that the relation cardinality bounds are consistent with the data and domain of knowledge. Unlike Rivero, Hernández, Ruiz, et al. (2012), we perform an on-the-fly normalisation of the graph in order to capture the semantics of sameAs-axioms without having to modify the underlying data. A naïve approach using a SPARQL query with COUNT operator will wrongly return two instances of ex:C1 instead of the expected count of 1 for the example in Figure 2.2 (left).

To overcome this issue we propose an axiomatisation with two rules (see Table 4.1), namely, *subject-equality* and *object-equality*. The axiomatisation imposes that duplicated elements are replaced by a representative element of equivalent type induced by owl:sameAs or similar predicates. This normalisation can be done replacing the underlying data (Motik, Nenov, Piro, et al., 2015) or on-the-fly (without modification) when needed. However, if the underlying data is modified, the links to other knowledge graphs stated by the sameAs-axioms are overwritten and lost.

Instead, here we follow an on-the-fly rewrite (line 1 of Algorithm 2) which performs the modifications in memory. A similar approach was previously used by Schenner, Bischof, Polleres, et al. (2014). In this rewriting approach, all the so-called *sameAs-cliques* are replaced by a selected representative entity (Motik, Nenov, Piro, et al., 2015). sameAs-cliques are built by connecting the triples returned by the $sameAsPairs$ function, generating complete graphs[3] with entities as nodes all of which are equal to each other. For each clique, all nodes (entities) are connected to each other, and a representative node can be chosen

[3] Any complete graph is its own maximal clique.

randomly. The representative is then used to rewrite the knowledge graph: replacing all the appearances of the equal entities in the clique by the representative everywhere in the knowledge graph. Note that for doing an on-the-fly rewriting, it is assumed that all cliques either fit in memory or are stored in an external fast-access index. This could be considered a limitation of our approach; however, in our experiments we did not find any case where these cliques did not fit in main memory.

In practice, the axiomatisation of Table 4.1 can be implemented on-the-fly either by using SPARQL 1.1 or programmatically. Next, we briefly introduce these two options:

4.3.2.1 SPARQL rewriting

The SPARQL query language has the limitation that it is unaware of the special semantics of `owl:sameAs` when evaluating triple patterns. This is a serious problem, especially when using the language to count the cardinality of properties. However, one can make use of several constructs in the language to overcome such limitation. We make use of a nested SPARQL 1.1 query with sub-selects (Polleres, Reutter, and Kostylev, 2016) as shown in Figure 4.2, which contains three sub-queries (`SQ-1`, `SQ-2` and `SQ-3`) with a wrapping query that aims to obtain the (entity=cardinality) pairs for a given property and entity type (Line 3). Under `SQ-1` are `SQ-2` and `SQ-3`, which perform the sameAs-clique generations for subjects and objects, respectively. Sub-query `SQ-2` implements the subject-equality rule, whereas sub-query `SQ-3` implements the object-equality rule. During the clique generation, a graph search to find equal entities is performed in all directions from a starting node using the property path `(owl:sameAs|^owl:sameAs)*`, which is a complex and resource-demanding query (check (Arenas, Conca, and Pérez, 2012; Kostylev, Reutter, Romero, et al., 2015) for a more detailed study on property paths in RDF). For each clique found, a representative is selected, and all "clone" entities are rewritten with the representative.

Clearly, the query used here is complex and resource demanding when executed with medium and large KBs. In terms of complexity, the evaluation of SPARQL queries can be solved in $O(|P| \cdot |\mathcal{G}|)$, where $|P|$ is the number of graph patterns in the query and $|\mathcal{G}|$ the number of triples in the knowledge base (Arenas, Gutiérrez, and Pérez, 2009). SPARQL is known to be in PSPACE-complete in general (Arenas, Gutiérrez, and Pérez, 2009). Hence, we also propose a more efficient and faster solution that works outside of a SPARQL endpoint.

4.3.2.2 Programmatic rewriting

Because of the complexity of the SPARQL solution, we propose a second rewriting approach that promises to be more time- and space-efficient. We thus frame the extraction of relation cardinality patterns as the well-known words count problem from linguistics. This problem has been one of the first to be addressed by modern parallelisation paradigms and frameworks. Thus, we can easily parallelise the algorithm using frameworks such as Apache Spark. By using Spark and the `filter` and `map` operations, we implemented a parallel and

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?first_sub (COUNT(DISTINCT ?first_obj) AS ?nbValue)
WHERE {
    {
        SELECT DISTINCT ?first_sub ?first_obj                                   % (SQ-1)
        WHERE {
            ?sub $relation ?obj .
            {
                SELECT ?sub ?first_sub                                          % (SQ-2)
                WHERE {
                    ?sub a $type .
                    ?sub ((owl:sameAs|^owl:sameAs)*) ?first_sub .
                    OPTIONAL {
                        ?notfirst ((owl:sameAs|^owl:sameAs)*) ?first_sub .
                        FILTER (STR(?notfirst) < STR(?first_sub))
                    }
                    FILTER (!BOUND(?notfirst))
                }
            }
            {
                SELECT ?obj ?first_obj                                          % (SQ-3)
                WHERE {
                    ?obj ((owl:sameAs|^owl:sameAs)*) ?first_obj .
                    OPTIONAL {
                        ?notfirst ((owl:sameAs|^owl:sameAs)*) ?first_obj .
                        FILTER (STR(?notfirst) < STR(?first_obj))
                    }
                    FILTER (!BOUND(?notfirst))
                }
            }
        }
    }
} GROUP BY ?first_sub
```

Figure 4.2: Query the cardinality of a relation (`$relation` variable) for every entity of a given type (`$type` variable).

efficient rewrite, where the sameAs-cliques are generated and used to normalise the knowledge graph triple by triple (see Figure 4.3, left). We generate the sameAs-cliques as follows: for each (h, `owl:sameAs`, t) triple we lexically compare h and t and select the minimum (say h), which becomes the *representative*; add a mapping from the representative entity to the other (say from h to t); if the non-representative (say t) in this axiom was the representative of other entities, then we update their mappings in cascade with the newly found representative (say h). We then apply a `map` operation over each initial triple in $\mathcal{G}$ and rewrite it according to the sameAs-cliques to obtain $\mathcal{G}'$ in $O(1)$.

4.3.3 Detection of cardinality patterns

After the normalisation step is finished, cardinalities can be collected for each relation (Algorithm 2, line 6). This ensures their accuracy, which is a major difference w.r.t. previous approaches such a Rivero, Hernández, Ruiz, et al. (2012). In the SPARQL-based approach, Figure 4.2 shows a query which performs both the normalisation of $\mathcal{G}$ and the detection of cardinality patterns in one place. However, complex SPARQL queries are hard to evaluate

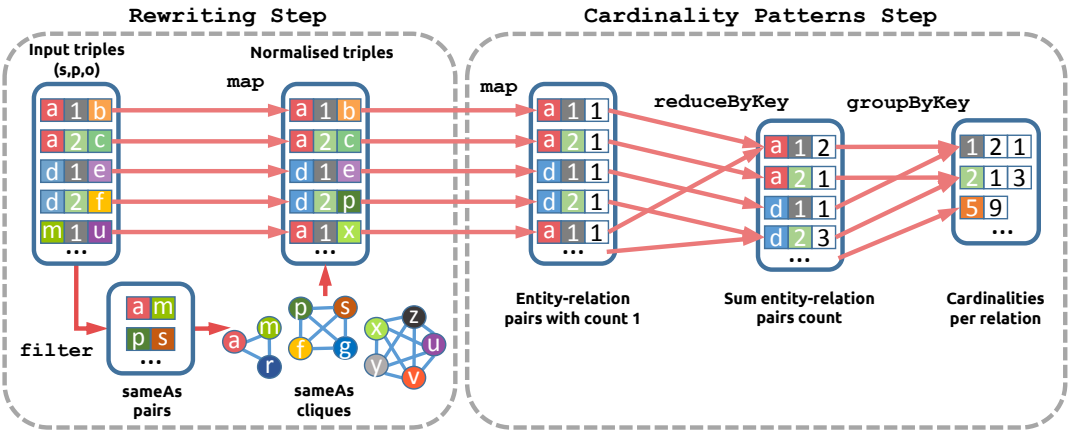


Figure 4.3: Cardinality patterns extraction using MapReduce approach.

and optimise, making this approach very inefficient and poorly scalable (Schmidt, Meier, and Lausen, 2010). On the other hand, the Spark-based approach can make use of multiple machines to scale and process large knowledge graphs in splits. We show a comparison between the two approaches in Section 4.4. Regardless of the rewriting approach, the output of the cardinality extraction is a map of (entity=cardinality) pairs for a given relation and entity type.

Users could take these cardinalities as patterns at this stage; however, several works have shown that knowledge graphs frequently contain noise and outliers (e.g., Hogan, Harth, Passant, et al. (2010), Paulheim and Bizer (2014), and Paulheim (2017)). In order to address this, we carry out a filtering of outliers from the cardinalities, which is described in the next section.

4.3.4 Outlier detection and filtering

Considering the adverse effects that outliers could cause in the method described so far, we now present techniques that can be used to detect and remove outliers from knowledge graphs (see Algorithm 2, line 7). Several supervised and unsupervised approaches can be used for the detection of outliers in numerical data (see Pearson (2005) for details); however, we did not find any labelled dataset for valid cardinality values. Therefore, we only consider unsupervised approaches for univariate data. We address the detection of outliers in a sequence of numbers as a statistical problem. Statistical outlier detection methods define rules for identifying points that do not respect the nominal behaviour of the data (i.e., appear to be anomalous) but they cannot explain the reason(s). Usually, the interpretation of outliers depends on the domain of knowledge and the nature of the data, thus, there are no universal rules for that. Interestingly, outlier detection approaches determine a lower and upper bound on the range of data, similarly to the semantics of a cardinality bound.

We studied three of the most commonly used methods for identifying outliers in univari-

ate data: ESD, Hampel and Bloxplot. The *extreme studentized deviation* (ESD) identifier (Rosner, 1983) is one of the most popular approaches. It computes the mean μ and standard deviation σ values and considers as outlier any value outside of the interval $[\mu - t \cdot \sigma, \mu + t \cdot \sigma]$, where $t = 3$ is usually used. The problem with ESD is that both the mean and the standard deviation are themselves sensitive to the presence of outliers in the data. *Hampel* identifier (Pearson, 2005) appears as an option, where the mean is replaced by the median med, and the standard deviation by the median absolute deviation (MAD). The range for outliers is defined as $[med - t \cdot MAD, med + t \cdot MAD]$. Since the median and MAD are more resistant to the influence of outliers than the mean and standard deviation, Hampel identifier is generally more effective than ESD. However, Hampel sometimes could be considered too aggressive, declaring too many outliers (Pearson, 2005). Box plot appears as a third option, and defines the range: $[Q1 - c \cdot IQD, Q3 + c \cdot IQD]$, where $Q1$ and $Q3$ are the lower and upper quartiles, respectively, and $IQD = Q3 - Q1$ is the interquartile distance—a measure similar to the standard deviation. The parameter c is similar to t in Hampel and ESD, and is commonly set to $c = 1.5$. Box plot is better suited for distributions that are moderately asymmetric, because it does not depend on an estimated "centre" of the data. Thus, in our evaluation we use the box plot rule to determine cardinality outliers.

Under the open-world assumption, we can expect that any cardinality mining and outlier detection algorithm will not be 100% accurate on the bounds. For example, if the property `birthDate` is missing for 80% of the entities, our algorithm will predict a lower bound of 0, even though every person should have a birth date. Similarly, if the knowledge graph contains only one parent for most people in it, the algorithm will infer a lower bound of 1 when the true value should be 2 (same logic applies for the upper bounds). On the other hand, if a property (e.g., `wonAward`) is present for 80% of the entities, our mining algorithm has more accurate data to work with and extract more realistic cardinality bounds. For this case, our algorithm will output a cardinality bound (0, 1) for the `wonAward` property in entities of type Actor. This bound shows that not all actors have the property, but those who have it, have it at most once. External knowledge or reasoning capabilities could be integrated with our approach for dealing with such cases; however, this is left as future work.

4.4 Experimental settings

We evaluate the application of our mining algorithm in its two variants against several real-world and synthetic knowledge graphs. These experiments aim to explore the usability of the mined cardinality bounds.

In our experiments, we distinguish between real-world and synthetic datasets. Commonly, real-world datasets are more heterogeneous in nature and not all relations appear for a given entity type. However, synthetic datasets are usually generated automatically by programs that randomly create instance data from an input ontology, thus, the resulting

Table 4.2: Datasets characteristics.

Dataset	№ triples	№ types	№ relations	№ sameAs
LinkedMDB	3,579,532	41	148	92,589
OpenCyc	2,413,894	7,613	165	360,014
UOBM	2,217,286	40	29	0
British National Library	210,820	24	45	14,761
Mondial	186,534	27	60	0
New York Times People	103,496	1	20	14,884
SWDF	101,321	62	132	759

instance data are more homogeneous. We hypothesise that these differences have twofold implications:

(a) synthetic knowledge graphs are usually more complete and consistent than real-world ones; and

(b) data shapes of incomplete entity types tend to have default bound values, i.e., cardinality bounds are usually 0 and ∞.

To evaluate these hypotheses, we perform an experiment in which we use the cardinality bounds to assess completeness and consistency of entity types in the knowledge graphs.

4.4.1 Datasets

We used seven datasets with different characteristics such as number of triples and sameAs-axioms. They are diverse in domain of knowledge, features, and represent both real-world and synthetic data. We present their characteristics in Table 4.2 and describe them as follows:

- LinkedMDB[4] is an open repository that describes movies, actors, directors, and so forth from the IMDB database.
- OpenCyc[5] is a large general KB released in 2012 that contains hundreds of thousands of terms in the domain of human knowledge covering places, organisations, business-related terms and people among others.
- UOBM[6] is a synthetic dataset that extends the Lehigh University Benchmark (LUMB), a university domain ontology, that contains information about faculties and students.
- British National Library[7] (BNL) is a dataset published by the National Library of the UK (second largest library in the world) about books and serials.

[4] http://data.linkedmdb.org/
[5] http://www.cyc.com/platform/opencyc
[6] https://www.cs.ox.ac.uk/isg/tools/UOBMGenerator/
[7] http://www.bl.uk/bibliographic/download.html

- Mondial[8] is a database compiled from geographical Web data sources such as CIA World Factbook, and Wikipedia.
- New York Times People[9] is a compilation of the most authoritative people mentioned in news of the New York Times newspaper since 2009.
- SWDF[10] is a small dataset containing information related to several semantic web related conferences and workshops.

4.4.2 Test settings

We implemented our cardinality mining Algorithm 2 using Python 3.5 and Apache Spark 2.1.0. We use an Intel Core i7 4.0 GHz machine with 32 GB of RAM running Linux kernel 3.2 to run experiments on different knowledge graphs. Although Spark can run on multiple machines, we only tested it on a single machine using multiple parallel processes—one per core using 8 cores in total.

4.5 Results and discussion

In this section, we evaluate the proposed approaches in quantitative and qualitative terms.

4.5.1 Quantitative evaluation

Intuitively, based on the scalability of Spark, one can foresee that the parallelised variant of our algorithm (see Figure 4.3) should outperform the other using SPARQL. To test this we ran both implementations over two selected datasets, the British National Library and Mondial datasets, where only the former contains `owl:sameAs` axioms. The times correspond only to the extraction of the cardinality bounds and do not include the loading of data in memory or in the RDF store (triplestore).

For the BNL dataset, we fix the entity type to $\tau = \texttt{Book}$, which co-occurs with 7 relations. We ran the code 10 times and obtained an average runtime of 253.908±0.351 sec for the SPARQL implementation, and 15.634±0.118 sec for the Spark one. This shows that the Spark implementation is 16x faster than the SPARQL implementation while performing the same task on the BNL dataset.

We repeat the same experiment over the Mondial dataset fixing the entity type $\tau = \text{River}$, which co-occurs with 8 relations. We ran again the code 10 times and obtained an average runtime of 117.739±0.651 sec for the SPARQL implementation, and 2.948±0.087 sec for the Spark one. This shows that the Spark implementation is 40x faster than using SPARQL,

[8] http://www.dbis.informatik.uni-goettingen.de/Mondial/
[9] https://datahub.io/dataset/nytimes-linked-open-data
[10] http://data.semanticweb.org/

Table 4.3: Evaluation of completeness and consistency per dataset: one type and five random properties per type.

Entity type	№ sameAs-cliques	№ triples before/after	Completeness ratio	Consistency ratio
Actor	92589	3579532/3536905	4/5	5/5
Fashion Model	118	1060/928	2/5	5/5
Research Assistant	0	135197/135197	4/5	5/5
Book	4515	97101/83556	2/5	3/5
Country	0	21766/21766	1/5	4/5
Concept	4979	58685/48780	2/5	5/5
InProceedings	759	101321/101302	0/5	5/5

while performing the same task with the Mondial dataset. The differences in the factors, 40x with the Mondial dataset and 16x with the BNL dataset, are due to the lower number of instances and the absence of sameAs-axioms in the data. These results show that the runtimes of the algorithm in either implementation are still small for two different but relatively small datasets. When dealing with much larger datasets such as DBpedia with 9.5 billion triples in its 2016-04 release, we can expect the runtimes to increase with the number of triples. Large datasets such as DBpedia grow in terms of entity types, triples per type, but also in terms of sameAs-axioms, where it will be interesting to test the performance of both implementations of Algorithm 2. Moreover, we believe that our Spark version will require several machines and run in its distributed manner. We leave such evaluations and other further optimisation as future work.

Finally, our experiments also show that the outlier detection method (i.e., box plot) does not add a significant overhead to the whole process and scales well (with the number of relations) for different data sizes.

4.5.2 Qualitative evaluation

After showing that the mining of cardinality bounds is efficient in time, and to show the benefits of studying cardinality constraints derived from automatically discovered bounds in knowledge graphs, we bring to the fore their use on the assessment of data quality. Specifically, we evaluate each entity type in the dimensions of completeness and consistency from a common sense point of view. Because the consideration of cardinality bounds is application-dependent, here we try to abstract (without loss of generality) from individual use cases. The cardinalities presented herein are considered robust bounds assessed to be a constraint by a knowledge engineer.

The characteristics of the studied datasets range between 1 up to 7,613 types and 20 up to 165 relations. To keep our study manageable, we selected randomly one entity type per dataset (7 in total) and five relations per type (35 in total). For each type, we show (see Table 4.3) the number of sameAs-cliques generated, and the number of triples before and

after the rewriting process.

We consider that a relation r in the context of a type τ is *complete* given a cardinality constraint if every entity h of type τ has the "right number" of triples (h, r, t); and *incomplete* otherwise. For example, a constraint might be that all books must have at least one relation `title`, but the same is not true for relation `comment`. Also, we consider that a relation r in the context of a type τ is *consistent* if the triples with predicate r and subject h (of type τ) comply with the cardinality bounds; and *inconsistent* otherwise. For example, a constraint might be that all books must have always one `title`; however, we found five books which violate this constraint having two titles. Based on a set of verified discovered robust bounds, in Table 4.3 we show the ratios of completeness and consistency found in the five relations per type. For example, 2/5 completeness ratio in the entity type Book indicates that 2 out of 5 relations presented complete data, and the rest was incomplete. We did the same to measure consistency. In general, we noticed a strong consistency and higher completeness on synthetic and curated datasets, where it is normal to define an ontology in which all instances are satisfied.

To further study the distribution of outliers in the cardinality bounds of each relation, we selected three entity types, namely, `foaf:Person`, `c4dm:Event` and `swrc:InProceedings` from the SWDF dataset. We plot the corresponding box plots for all relations in these entity types in Figures 4.4 to 4.6, respectively. These box plots provide us with the following insights:

(1) `foaf:Person`: In this entity type we can see that the box plots are quite flat (i.e. they have small width). This suggests that overall the relation cardinalities have a high level of uniformity (agreement among them). Their values are centred around 1, thus, the mean cardinality is usually one. However, several probable outliers (denoted by black dots) are present in most of the relations. For instance, the relation `swc:holdsRole` is the one with more of them, going up to 22. `swc:holdsRole` records the roles of a person (e.g., PC member, chair, speaker, keynote) in a given event (e.g., conference, workshop), and shows that there is a huge variation of the cardinalities for different entities. Some people repeat more as organisers of events than others. A similar behaviour can be seen in the relation `foaf:made` used to state when a person is the author of a paper or a proceeding editor. Regarding the consistency of data, we can mention that two similar ontologies `swc` and `swrc` are used in entity instances of this type; however, equivalent relations in these ontologies do not match in the instance data. For instance, this can be seen in the `swrc:affiliation` and `swc:affiliation` relations, where the latter contains more variability than the former, but the former has more outliers.

(2) `c4dm:Event`: In this entity type, we can also see quite flat box plots, mostly around the value 1, and with little outliers. Differences are observed in the relations `swc:isSuperEventOf` (used to state sub-events of a conference/workshop, such as coffee/lunch breaks or talks) and `dce:subject` (used to state the topics of an event). The box plot of these two relations is larger in width meaning that the cardinalities are different but still within some low boundaries. Moreover, two other interesting cases

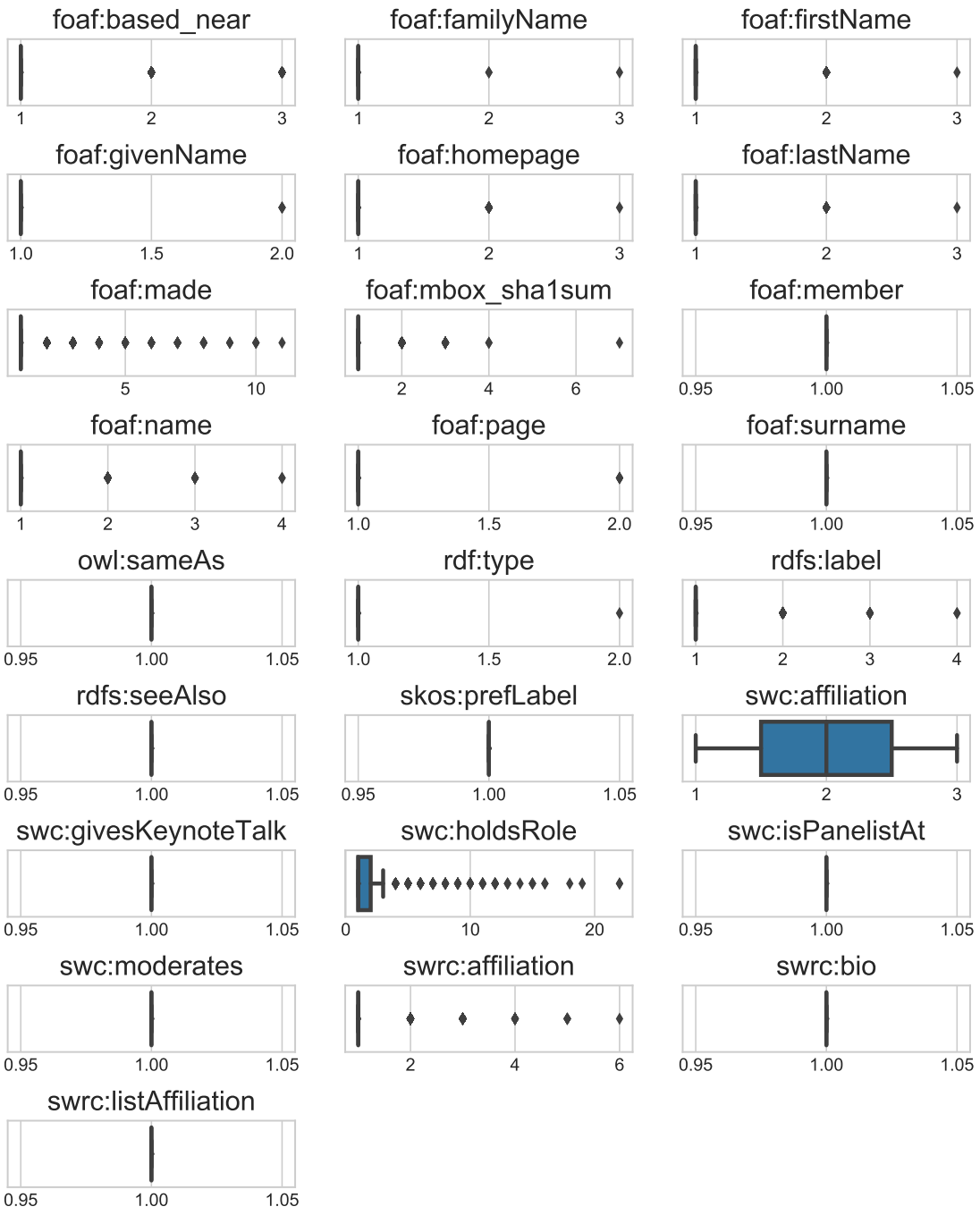


Figure 4.4: Box plot figures for each relation in the entity type `foaf:Person` of the SWDF KG.

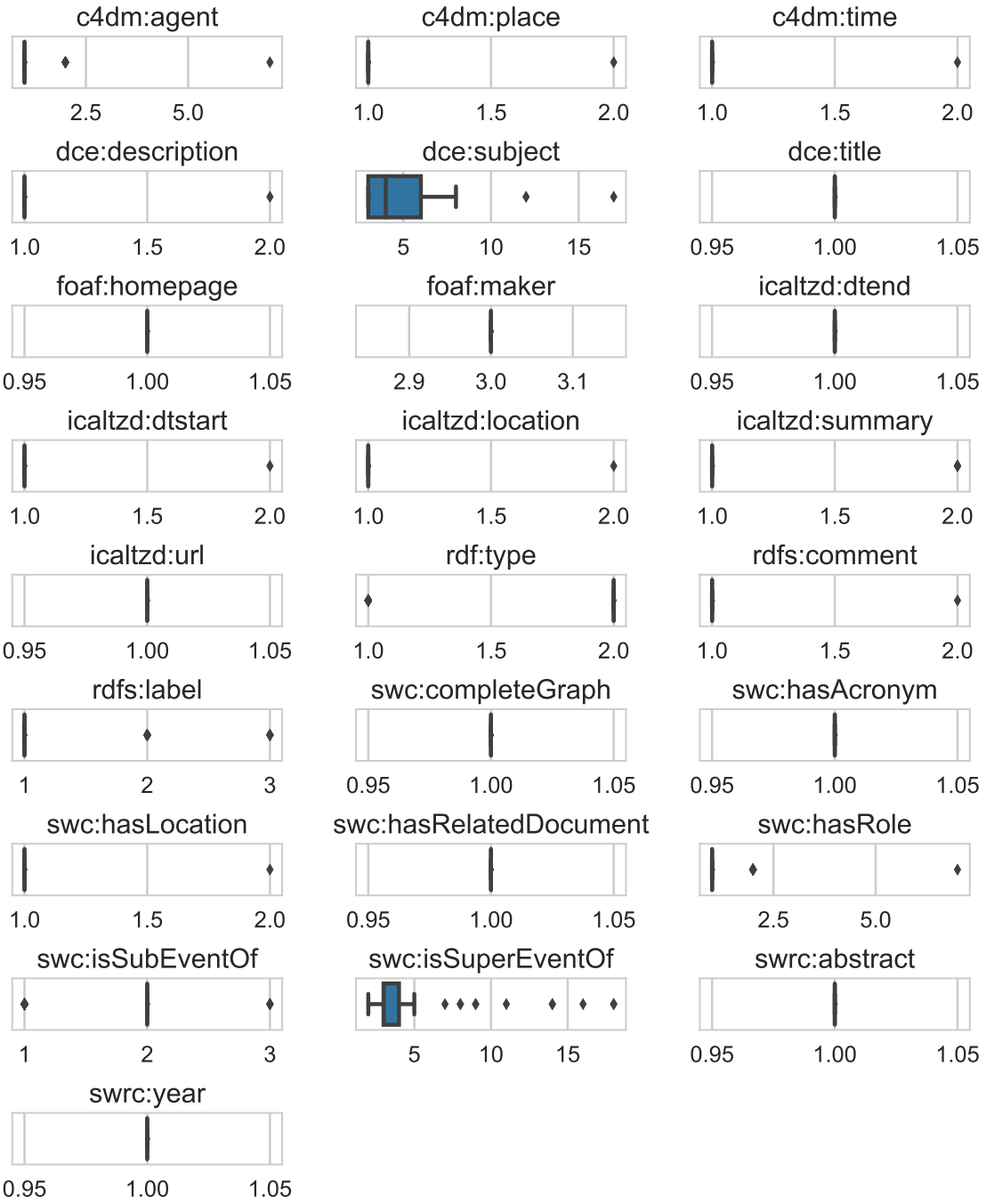


Figure 4.5: Box plot figures for each relation in the entity type `c4dm:Event` of the SWDF KG.

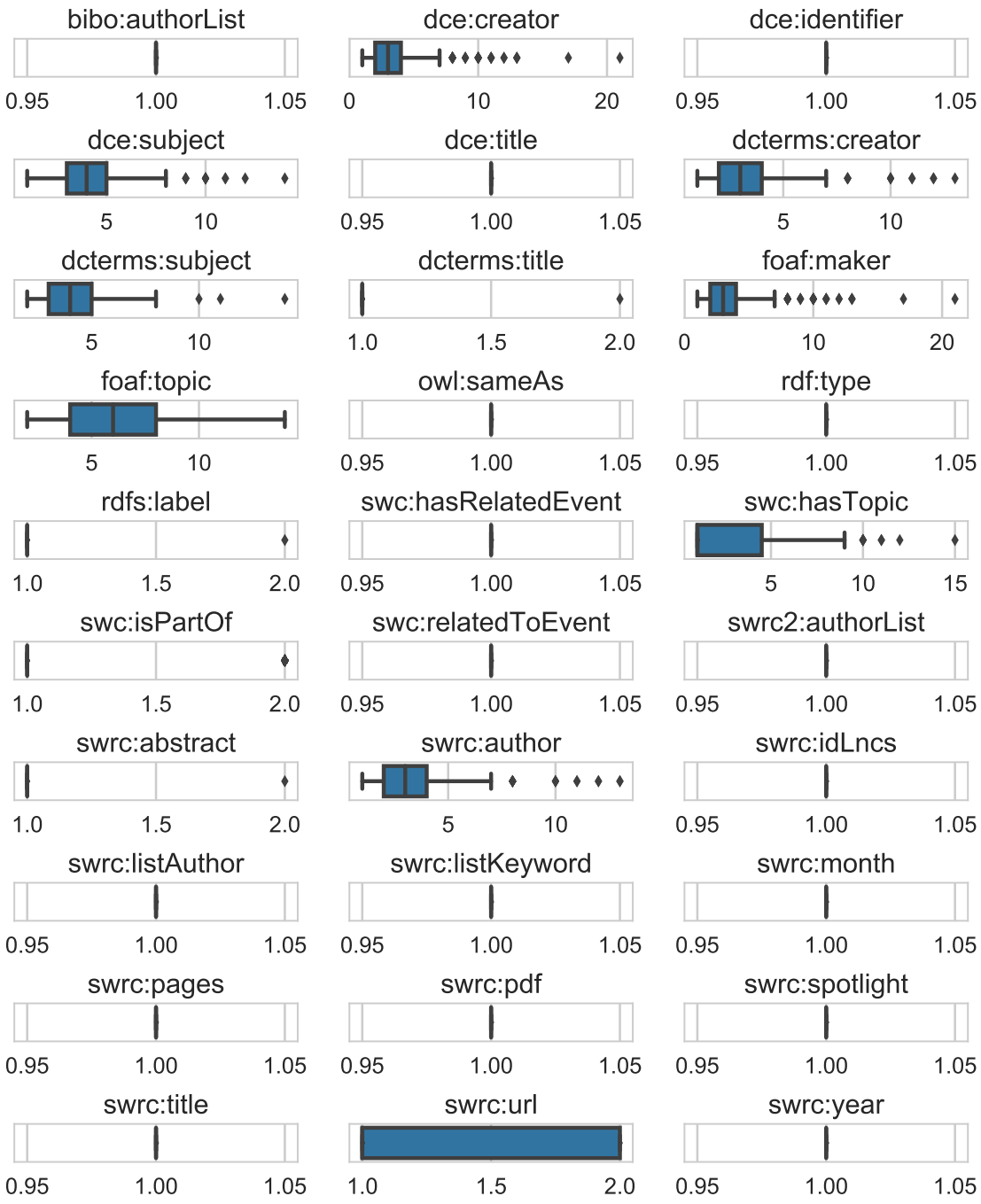


Figure 4.6: Box plot figures for each relation in the entity type `swrc:InProceedings` of the SWDF KG.

```
card(mondial:name, mondial:Volcano)=(1,1)
card(mondial:elevation, mondial:Volcano)=(1,1)
card(mondial:longitude, mondial:Volcano)=(1,1)
card(mondial:latitude, mondial:Volcano)=(1,1)
card(mondial:locatedIn, mondial:Volcano)=(1,3)
card(mondial:lastEruption, mondial:Volcano)=(0,1)
```

Figure 4.7: Subset of relation cardinality bounds for type `mondial:Volcano`.

are seen in the `rdf:type` and `swc:isSubEventOf` box plots, which present some outliers below the box plot. This is due to entities of this type have two values (e.g., `swc:TalkEvent`, `c4dm:Event`) for these relations in average, but few cases have only one.

(3) `swrc:InProceedings`: This entity type has more relations with wider box plots compared to the previous two and it also contains more outliers, indicating that cardinalities have high variability among entities. Some insights that we can collect by looking at the box plots are: (a) more than one relation coming from different ontologies is used to express the same attribute; and (b) box plots for relations that express the same attribute do not match in most cases. For instance, to illustrate (a) we have the case of the relations `dce:creator`, `dcterms:creator`, `swrc:author`, and `foaf:maker` that are used to represent the authors of a paper. However, some sets of entities contain only a subset of these relations. We believe that this depends on the metadata provided (when provided) by the conferences and workshops, which is not always the same. Likewise, the relations `dce:subject` and `dcterms:subject` show discordance in their data across entities.

The information collected from the box plots and cardinality bounds could be used to complete missing relations or spot inconsistencies in the data. Furthermore, we argue that by detecting cardinality inconsistencies and incompleteness we can determine structural problems at instance level. For example, if the knowledge graph is being automatically generated from a textual source, e.g., PubMed articles, structural problems such as missing labels could indicate issues in the pipeline generating the knowledge graph. In turn, this can be used to guide repair methods and move towards better quality of knowledge graphs.

4.6 Summary

Cardinality bounds turned out to be a powerful tool to expose the inherent structure or shape of knowledge graphs. In this chapter, we provided a definition of cardinality bounds that builds upon similar research done in XML and relational databases Section 4.2. With a definition in place, in Section 4.3 we proposed an algorithm for mining accurate and robust cardinality bounds from instance data. We identified that two factor can significantly impact the mining of cardinality: the unique name assumption and outliers. To deal with UNA, we proposed a normalisation that removes duplicated entities and edges in the graph

according to their declaration as semantically equal. Moreover, we applied statistical outlier detection and removal methods to obtain robust cardinality bounds. Two implementations of this algorithm are compared using real-world and synthetic datasets. Our experimental results (Section 4.4) show that the use of Apache Spark can be up to 40x faster than using SPARQL for collecting the cardinality patterns (Section 4.5). We believe that cardinality bounds can play an important role in applications such as query re-writing, validation, and representation learning. In Chapter 5, we will study how cardinality could be used with machine learning models to propose an approximated validation algorithm. Similarly, cardinality can be used to build graph-based feature models to complete Biomedical knowledge graphs (Chapter 6). Finally, in Chapter 7 we will test the use of cardinality bounds to encode common sense in distributed representations learnt from knowledge graphs.

CHAPTER 5

Approximate Validation of Knowledge Graphs

Contents

Knowledge graphs are a rich representation of information that provide a huge deal of flexibility to users. That flexibility, however, comes at a price when data consumers require to *validate* the data consistency. For example, a business use case could require to verify that all customers have a contact email, or it may be required that customers have at least one contact phone number and address for billing. Validation is usage dependant — R. Y. Wang and Strong (1996) highlighted the importance of the concept "fitness for use" when measuring *data quality*. However, validation of knowledge graphs at scale is an open problem that remains unresolved. Current validation approaches have two main shortcomings: (1) their output of validation is a Boolean value, in some cases accompanied by a report pointing the sources of inconsistency (error); and (2) they rely on technologies that do not scale well, such as SPARQL and regular expressions. Therefore, it becomes impractical to validate large knowledge graphs. These shortcomings make the validation problem of knowledge graphs daunting, let alone the cases dealing with numerous sources of inconsistency such as Web data. In this chapter, we tackle the problem of validation using heuristics or approximate algorithms to help when accuracy-efficiency trade-offs can be taken.

5.1 Problem statement

In the last decade, knowledge base construction (KBC) systems have facilitated the automatic creation of very large knowledge graphs. For example, the never-ending language learning (NELL) (Carlson, Betteridge, Kisiel, et al., 2010; Mitchell, Cohen, Jr., et al., 2018) knowledge graph comprises over 50 million facts[1]; Wikidata (Erxleben, Günther, Krötzsch, et al., 2014; Tanon, Vrandecic, Schaffert, et al., 2016), a shared knowledge graph providing structured data to the Wikipedias in different languages, keeps growing continuously and has ca. 57 million facts[2]; or the Google's Knowledge Graph, introduced in May 2012 and built automatically with information from multiple sources, as of October 2016 it was estimated to contain over 70 billion facts. Validating knowledge graphs of such magnitude is a complex and resource demanding task that has not received enough attention thus far from the research community. We believe that in scenarios like that, data consumers are willing to exploit accuracy-efficiency trade-offs to achieving the validation. In other words, users would be willing to gain efficiency by allowing occasional errors in the data.

Data consumers pose different requirements for the characteristics that a knowledge graph must have to be used depending on their their use cases, but currently there are few tools or solutions to assist them determining the suitability of a given knowledge graph. R. Y. Wang and Strong (1996) highlighted the importance of the concept "fitness for use" when measuring *data quality*: There is no one-fits-all solution and the selection of a data source depends on the specific use case at hands. In Chapter 4, we introduced a bottom-up approach to extract the cardinality constraints that data naturally exhibits, helping data consumers to understand the shape and characteristics their data have and analyse whether these are acceptable or not for their use case(s). Conversely, in this chapter, we address a top-down approach a.k.a. *knowledge graph validation*, where a data consumer has a set of restrictions the data must satisfy and she wants to check whether a given knowledge graph adheres to this set of constraints (Kontokostas, Westphal, Auer, et al., 2014; Labra Gayo, E. Prud'hommeaux, Boneva, et al., 2018).

In the area of database management systems, this problem is well known as *consistency checking*, where a relational database is checked against a set of integrity constraints (Abiteboul, Hull, and Vianu, 1995) — which are limitations imposed on the data that are supposed to be satisfied *all the time* by instances of the database. For a knowledge graph $\mathcal{G}$, this problem has been translated (Boneva, Labra Gayo, Hym, et al., 2014; Kontokostas, Westphal, Auer, et al., 2014) as the process of determining whether $\mathcal{G}$ is deemed valid w.r.t. a "shape" S if all relations in $\mathcal{G}$ match the structural model defined in S, denoted by $S \models \mathcal{G}$, analogous to the processes in relational and XML databases. Shapes indicate the structure of entities: allowed relations, their data types, and cardinalities; however, they can (and usually are) overloaded with more complex constraints defined using SPARQL query language (Labra Gayo, E. Prud'hommeaux, Boneva, et al., 2018).

[1] As of June 2019, http://rtw.ml.cmu.edu/rtw/
[2] As of June 2019, https://www.wikidata.org/wiki/Wikidata:Statistics

The $S \models \mathcal{G}$ notion of validation is strict and its strictness is often required in many use cases; however, some users may have more flexible needs in terms of accuracy in favour of a better validation time and scalability. For example, consider a case where the size of the knowledge graph is very large (as in the example of KBC mentioned above) and strict validation is impractical.

A few approaches have been proposed for strict validation of knowledge graphs (see our review of the most popular constraint languages for knowledge graphs in Section 3.1.2), among them SHACL is the W3C recommendation for validating RDF knowledge graphs against a set of constraints. Although SHACL is the current W3C recommendation, Corman, Reutter, and Savkovic (2018a) have shown that validating RDF knowledge graphs against SHACL constraints is NP-hard in the size of the graph, thus, intractable for large Web-scale knowledge graphs. When the optimal solution cannot be reached, or if an optimal solution is not necessary, or it is highly expensive, researcher and practitioners have exploited approximate approaches that provide a "good" solution at a lower cost. Such approaches are known as "approximation algorithms" (Johnson, 1974) that are polynomial by relaxing the need for accuracy guarantees. In this work, we introduce the problem of *approximate structure validation* that aims to apply approximate and scalable methods to validate a knowledge graph without the need for 100% accuracy.

Definition 5.1

We define *approximate structure validation* given a knowledge graph $\mathcal{G}$ and a shapes graph S, as the process of determining the level σ in which the structure of an entity e can be mapped from some input shape in S whose target node is e. For each $e \in \mathcal{E}$ in $\mathcal{G}$, $\sigma_i \in [0, 1]$ is computed and $\mathcal{G}$ is deemed valid w.r.t. S with a $\sigma = \frac{1}{|\mathcal{E}|} \sum_{i \in |\mathcal{E}|} \sigma_i$, denoted by $S \models_{\sigma} \mathcal{G}$. We refer to σ as the global conformance score that refers to the aggregated level of conformity w.r.t. the schema S.

Henceforth, we will define an approach for computing the conformance score as the probability of a knowledge graph $\mathcal{G}$ to comply with a schema S.

We can thus state the approximate graph validation problem as follows:

Approximate structure validation problem

Input: a knowledge graph $\mathcal{G}$ and a shapes graph S.

Output: Whether $S \models_{\sigma} \mathcal{G}$ with a conformance score σ.

A σ value close to 0 indicates that $\mathcal{G}$ does not satisfy most (or none) of the structure defined by S, whilst a value of σ closer to 1 indicates that $\mathcal{G}$ satisfy most (or all) of the structure defined by S. Intuitively, we can notice that when $\sigma = 1$, the knowledge graph $\mathcal{G}$ is deemed to be valid for all structure constraints, i.e., $S \models \mathcal{G}$.[3] To that end, we hypothesise

[3]Note that the $\models$ notation used here is borrowed from databases area and does not mean entailment or

that an approximate solution for validation can use machine learning approaches to learn the structure of knowledge graphs and entity types. The validity of a knowledge graph $\mathcal{G}$ is the implication of the validity of all entities in $\mathcal{G}$. Using the learnt models we can predict the probability $\Pr(y = 1 \mid \boldsymbol{e})$, where $\boldsymbol{e}$ is the latent shape representation of an unseen entity e. This probability represents the level to which e satisfies the constraints in a given schema. An entity that follows all constraints will get a probability closer to 1, while an entity that does not will get a lower probability. (Note that one of the challenges we tackle in this chapter is the definition of such latent shape representations.)

Henceforth, we focus on the approximate structure validation problem for knowledge graphs that covers cases where users may prefer a faster but approximate solution that fulfils their validation requirements.

5.2 Related work

Although, the research stream on the validation problem in knowledge graphs is fairly new, the interest on it has been growing due to the relevance of knowledge graphs in artificial intelligence use cases. In this section, we cover the work done validating knowledge graphs, and the approximate approaches for validation proposed in the field of databases.

Knowledge graph validation. The validation of knowledge graphs is a much sought-after feature that has attracted increasing attention due to its critical role in the quality assessment of the information they contain (Labra Gayo, E. Prud'hommeaux, Boneva, et al., 2018). Data consumers are more and more in need of means to check whether a knowledge graph conform to some intended structure. In Chapter 3, we have already reviewed the most relevant constraint languages and how they are used for validating knowledge graphs. When compared to previous approaches, we noticed that there are no works studying approximate solutions to the validation problem on knowledge graphs. However, the idea of approximate validation is not new. For instance, Labra Gayo, García-González, Fernández-Alvarez, et al. (2019) mentions the possibility of using probabilistic reasoning as an approach to define approximate validation algorithms for knowledge graphs, although no further details are provided.

The purpose of constraint languages is to define the structure and constraints that a given knowledge graph is expected to conform. Current approaches for validation make use of so-called *shapes graphs*, usually defined using grammars (e.g., ShEx) (Boneva, Labra Gayo, and E. G. Prud'hommeaux, 2017) or SPARQL queries (e.g., SPIN) (Kontokostas, Westphal, Auer, et al., 2014), to check the consistency (veracity) of knowledge graphs. The expressiveness of shapes graphs is bounded by the expressiveness of the constraint language they are written on, i.e., the constraints that conform a given shapes graph may be varied. T. Hartmann (2016) compiled a set of use cases from the Dublin Core Metadata Initiative

semantic consequence.

Table 5.1: Coverage of validation approaches w.r.t. constraint types (T. Hartmann, 2016).

DSP	ReSH	ShEx	SHACL	OWL 2	SPIN (SPARQL)
17.3% (14)	25.9% (21)	29.6% (24)	51.9% (42)	67.9% (55)	100% (81)

(DCMI) Application Profiles Task Group, the W3C RDF Data Shapes working group, and the W3C Shapes Constraint Language (SHACL) working group, and identified 81 types of constraints required by various stakeholders. These constraints should be expressible by constraints languages in order to meet the requirements imposed over knowledge graphs. Using this set of 81 constraints, T. Hartmann (2016) evaluated the expressiveness of the most common constraint languages and concluded that SPARQL (E. Prud'hommeaux and Seaborne, 2008) and SPARQL Inferencing Notation (SPIN) (Knublauch, J. A. Hendler, and Idehen, 2011) are the two most powerful and widely used languages for constraints formulation and validation (Fürber and Hepp, 2010). Table 5.1 table shows the coverage of validation languages w.r.t. the 81 constraint types. (Note that the SHACL specification was still not finished at the time when Hartmann's work was completed, and its coverage may be considerably different.)

To generate our ground-truth, we will use SHACL as validation language given it is the current W3C recommendation. Furthermore, following the initial motivation of shapes graphs, in this chapter and thesis, we restrict the scope of shapes to just structure definitions for a set of entities. That is, we focus on the presence and cardinality of relations for entities. For example, saying that a person must have a birth date and/or can have zero or more children indicates structure, whilst checking the format of her birth date (e.g., `YYYY-MM-DD`) does not, under our definition.

Approximate validation in databases. While no concrete approach has been proposed thus far for RDF knowledge graphs, approximate validation methods have been studied for other data models. In databases, $I \models \Sigma$ denotes that a relation I satisfies a set of integrity constraints Σ, if $I \models \sigma$ for each $\sigma \in \Sigma$ (Abiteboul, Hull, and Vianu, 1995). This definition results in a Boolean result for the validation operation (true or false) indicating whether the whole document is valid or not w.r.t. the set of constraints. A relaxation to this definition has been proposed for XML. In Tekli, Chbeir, Traina, et al. (2015), the authors propose an approximate XML structure validation between an XML document (tree) D and an XML (regular tree) grammar G, denoted by $G \approx_{\sigma} D$. Tekli et al. use a similarity score σ to denote that D approximately conforms to G with a similarity score σ obtained by computing the XML document/grammar similarity using a tree edit distance. In particular, σ underlines the degree of membership of document D to the grammar language $L(G)$, which represents the set of all the possible XML documents that can be generated using the XML grammar G. Their approach is the most similar to ours in terms of their goal, but they solve a very different problem dealing with the tree structure of XML documents.

Approximation algorithms are efficient algorithms to find approximate solutions to NP-

hard problems (Williamson and Shmoys, 2011). In this thesis work, we hypothesise that an approximate solution to the validation problem can use machine learning approaches that learn the structure of knowledge graphs and entity types. In the next section, we propose our method for approximate structure validation of knowledge graphs using graph features for learning structure representations.

5.3 An algorithm for approximate structure validation

With scalability in mind, our approximate validation approach uses a divide-and-conquer technique, where: (a) the knowledge graph is split into smaller units (e.g., subgraphs); (b) a machine learning model is build to determine the validity of an entity w.r.t. a given class T; (c) and finally, the validation is achieved by passing each unit to the corresponding class model m_T and getting its prediction probability $\Pr(y = 1 \mid \boldsymbol{e})$ based on the extracted latent shape for entity e. We now describe the details of our approximate structure validation algorithm for knowledge graphs.

5.3.1 Approximate validation

Although popular approaches for validating knowledge graphs use SPARQL to express data constraints or perform reasoning to fully capture the semantics of data (Bosch, Acar, Nolle, et al., 2015), in this dissertation, we consider knowledge graphs as a data model not limited to RDF only. Additionally, knowledge graphs can be built at Web scale from different sources and not necessarily follow an OWL ontology. Thus, we consider that validation of knowledge graphs should not be restricted to Semantic Web technologies in all cases. In this chapter, we introduce an approximate approach that relaxes the strict notion of validation based on machine learning.

As we have already pointed out before, current approaches use a "binary" view of validation, where the validity is measured over the *whole* knowledge graph, disregarding the size and type of the structural issues that cause violations. Another shortcoming is scalability that stems from this binary view, where knowledge graphs are atomically validated, making validation unfeasible for large knowledge graphs. In our approach, we first divide a knowledge graph into smaller units, create machine learning models to classify these units w.r.t. the classes in the knowledge graph, and run the validation for each one of these units by passing them to their respective model. Henceforth, we consider a subgraph as a unit structure extracted around a target entity. Let $\mathsf{Graph}^d(\mathcal{G}, e)$ be the subgraph of depth d around an entity (node) e. Note that if we merge all n subgraphs extracted from $\mathcal{G}$, the output is the original knowledge graph $\mathcal{G}$, i.e., $\mathcal{G} = \bigcup_{i=1}^{n} \mathsf{Graph}^d(\mathcal{G}, e)$. Building upon this equality, we can claim that $S \models\!\!\approx_\sigma \mathcal{G}$ is equivalent to $S \models\!\!\approx_\sigma \mathsf{Graph}^d(\mathcal{G}, e), \forall e \in \mathcal{E}$. Intuitively, this assumption will allow us to implement more efficient approaches by, for example, val-

idating each subgraph against its corresponding shape graph in parallel.

Example 5.1

Example shapes graph S_1 containing two shapes and their corresponding property shapes:

```
@prefix schema: <http://schema.org/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

schema:Hotel
a sh:NodeShape ;
sh:targetClass schema:Hotel ;
sh:property [ sh:path schema:name ;
sh:datatype xsd:string ;
] ;
sh:property [ sh:path schema:url ;
sh:minCount 1 ;
sh:maxCount 3 ;
] ;
sh:property [ sh:path schema:priceRange ;
sh:in ( "€" "€€" "€€€" ) ;
] ;
sh:property [ sh:path schema:address ;
sh:node schema:AddressShape ;
] .

schema:AddressShape
a sh:NodeShape ;
sh:closed true ;
sh:property [ sh:path schema:streetAddress ;
sh:datatype rdf:langString ;
] ;
sh:property [ sh:path schema:addressRegion ;
sh:datatype xsd:string ;
] ;
sh:property [ sh:path schema:postalCode ;
sh:or ( [ sh:datatype xsd:string ] [ sh:datatype xsd:integer ] ) ;
] .
```

In this chapter we propose an agnostic validation algorithm that propagates the validity from one entity to its closest neighbours. For this, we make the following assumptions:

1. entities used in the training of our machine learning models are correctly typed, i.e., if the statement (A, `rdf:type`, C, i, s) in the knowledge graph, entity A belongs to class C in reality;

2. structure validity can be captured by similarity functions, i.e., if entity A is valid and entity B has a high similarity score with A, then B is said to be valid with that similarity score.

Example 5.1 shows a shapes graph S_1 that contains two node shapes `schema:Hotel` and

schema:AddressShape for validating entities of the target type schema:Hotel. This is a common shapes graph that allows to validate Web content generated using Schema.org (R. V. Guha, Brickley, and Macbeth, 2016), which is a set of vocabularies that allow application developers to exchange structured data on the Web.

Schema.org allows Web developers to use structured data markup embedded in Web pages to annotate entities (e.g., people, places, events, products, offers) and relationships between entities and actions.[4] Annotations embedded in web pages are then extracted to generate knowledge graphs with data from many different sites. Such is the adoption and value of schema.org annotations that they serve as data sources for Google's Knowledge Graph project, providing background information, e.g., logos, contact, and social information (R. V. Guha, Brickley, and Macbeth, 2016). Example 5.2 shows a subset of the triples extracted by The Web Data Commons project (Meusel, Petrovski, and Bizer, 2014) runs large-scale processes to extract structured data from the Common Web Crawl and makes the datasets publicly available.[5]

Example 5.2

Below an extract of a knowledge graph $\mathcal{G}_1$ retrieved from the 2018-12 Common Web Crawl and encoded in Turtle:

```
@prefix schema: <http://schema.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix ex: <http://example.com/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

ex:HotelElCiervo
    a schema:Hotel ;
    schema:name "Hotel El Ciervo" ;
    schema:url <http://hotelsambencant.cat/europa/espanya/catalunya/vielha/hotel_el_ciervo.html> ;
    schema:priceRange "€" ;
    schema:address [ schema:streetAddress "Passeig Sant Oren\u00E7, 3"@es ;
        schema:postalCode 25530
    ] .

ex:MonteParadiso
    a schema:Hotel ;
    schema:name "Monte del Paradiso" ;
    schema:priceRange "€€" ;
    schema:address [ schema:streetAddress "Localita Costa di Monte Acuto"@it ;
        schema:addressRegion "Umbria"@it ;
        schema:postalCode "06019"
    ] .
```

In Example 5.2, we can see two focus nodes in the data graph $\mathcal{G}_1$, namely, ex:HotelElCiervo and ex:MonteParadiso, which belong to the class schema:Hotel defined in Example 5.1. When validating these focus nodes against the shapes graph S_1, we have that $S_1 \not\models \mathcal{G}_1$ because ex:MonteParadiso does not comply with two of the constraints in S_1:

[4] https://schema.org/ (Accessed May 11, 2019)
[5] http://webdatacommons.org/structureddata/ (Accessed Feb 13, 2019)

(1) it does not have at least one value for the relation `schema:url`; and
(2) it uses a language tag `@it` for the `schema:addressRegion` relation (Line 22), which is supposed to have a datatype `xsd:string`, thus, not accepting a language tag.

Note that even though the entity `ex:MonteParadiso` is not valid w.r.t. the defined schema S_1, it does have all the relations required in the structure of `schema:Hotel`. One could say that even when the strict validation is not satisfied (Boolean value), the entity does conform with most of the structure satisfying 5 out of 7 property shapes (∼71.4%) defined in the `schema:Hotel` shapes graph.

Therefore, for achieving our goal of approximate validation, we follow the validation strategy of ShEx that defines associations between nodes and shapes as input to the validation process. The ShapeMap Language (E. Prud'hommeaux and Thomas Baker, 2019) divides the validation of a knowledge graph into `node@shape` pairs, such as

`ex:HotelElCiervo@schema:Hotel`, `ex:MonteParadiso@schema:Hotel`,

where the shape map determines that validation is performed over each pair: entity (node) validated against the given shape. This strategy has several advantages when compared to other approaches that will stop after the first error is found and return a binary value. Shape maps assume certain independence between the validation of individual entities that allows us to formalise Claim 5.1.

Claim 5.1

Let S be a schema (shapes graph), $\mathcal{G}$ a knowledge graph (data graph), then $S \approx_\sigma \mathcal{G}$ is equivalent to $S \approx_\sigma \mathsf{Graph}^d(\mathcal{G}, e), \forall e \in \mathcal{E}$, where d represents the depth of the paths in the subgraph.

Proof. Let $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \ell)$ be a knowledge graph. By definition, each $\mathsf{Graph}^d(\mathcal{G}, e)$ generates a graph $\mathcal{G}' = (\mathcal{E}', \mathcal{R}', \ell')$, where $\mathcal{E}' \subseteq \mathcal{E}$ and $\mathcal{R}' \subseteq \mathcal{R}$. For a large enough depth d, a subgraph could contain all the entities (nodes) in $\mathcal{E}$ if the graph is connected, meaning that $\mathcal{G} \equiv \mathcal{G}'$. If the graph is not connected, then by covering all entities in $\mathcal{E}$, one should reach the equivalence between both graphs at some point.

It is clear from our definition in claim 5.1 that to capture hierarchical or composite relationships, such as *great-grand-father*, it is required a value of $d > 1$. For example, let's assume a knowledge graph containing only triples with the *hasFather* relation and we want to capture the relationship *hasGreatGrandFather*. To achieve this, we should at least use a value $d = 3$ to capture such cases, e.g., (*Bart Simpson*, *hasGreatGrandFather*, *Orville J. Simpson*) can be represented by the following path:

$$(\textit{Bart Simpson}, \textit{hasFather}, \textit{Homer Simpson}) \wedge$$
$$(\textit{Homer Simpson}, \textit{hasFather}, \textit{Abraham Jedediah Simpson II}) \wedge$$
$$(\textit{Abraham Jedediah Simpson II}, \textit{hasFather}, \textit{Orville J. Simpson}).$$

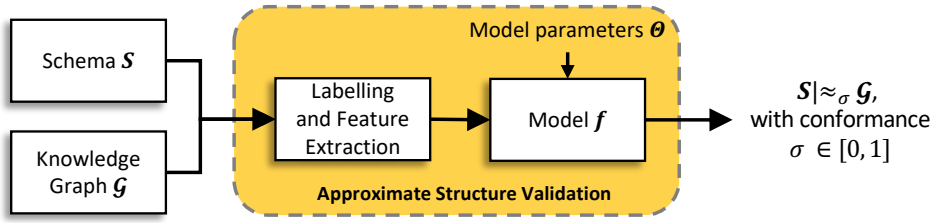


Figure 5.1: Approximate structure validation model with machine learning.

Note that here we focus on the problem of *structural validation of RDF*. Although the schema.org vocabulary introduced by Google can potentially generate an infinite number of constraints using existing standards and serialised using RDF, not all of these constraints are considered structural in this work.[6] For example, we will not validate that cycle-free relations such as `schema:children` (i.e., children must be born after their parents) or irreflexive relations such as `schema:children`, `schema:knows`, etc., are satisfied. Our proposed approach is non-deterministic and unaware of everything but the paths included in the subgraphs $\text{Graph}^d(\mathcal{G}, e,)$. Similarly, we do not validate cases like relation `schema:deathDate` must be after `schema:birthDate` or that the `schema:email` must match a certain regular expression. We do not consider these cases as part of the structural validation of RDF.

Because of how we frame the problem and task, we do not validate specific constraints one by one, but many *structural* constraints at the same time. We refer to these structural constraints as the latent shape of entities. We cast the validation problem as a similarity problem between entities, and say that if entity A is valid w.r.t. a shape $\mathcal{S}$ with target class $\mathcal{C}$, then every entity X whose similarity to A is close to 1 should also be valid w.r.t. $\mathcal{S}$ with a conformance score σ.

5.3.2 Machine learning framework

To actually provide a fast and scalable approximate validation, we use machine learning algorithms to infer the conformance score given a knowledge graph and a shapes graph. Our goal is then to find the best model to obtain $S \mid\!\approx_\sigma \mathcal{G}$, as shown in Figure 5.1.

Following a probabilistic approach, we define a function $f_S\colon \mathbb{R}^k \rightarrow [0,1]$, where k is the length of the vector representation of entities. Given an entity e and its vector representation denoted $\boldsymbol{e}$, $f_S(\boldsymbol{e})$ returns the probability with which entity e satisfy the shapes graph S, i.e., its conformance score. The design of f_S depends on the input shapes graph S and can be learnt from sample data using a range of alternative machine learning approaches, such as supervised decision tree or semi-supervised label propagation. Building upon our Claim 5.1, we say that $S \mid\!\approx_{\sigma=1} \mathcal{G}$ (or $S \models \mathcal{G}$) if $f_S(\boldsymbol{e}_i) = 1$ for every entity $e_i \in \mathcal{E}$. In other words, the knowledge graph $\mathcal{G}$ is said to satisfy or be valid w.r.t. S, because every

[6] We refer the reader to https://w3c.github.io/data-shapes/data-shapes-ucr/ for more details.

entity in $\mathcal{G}$ conforms with the schema.

We define a *domain* $\mathcal{D}$ that consists of two components: a feature space $\mathcal{X}$ and a marginal probability distribution $\Pr(X)$, where $X = \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\} \in \mathcal{X}$. Henceforth, $\boldsymbol{e}_i$ is a vector representation of the i-th entity, $\mathcal{X}$ is the space of all entity vectors, and X is a particular learning sample of size n. We will later discuss methods to obtain a vector representation of entities in a knowledge graph in Section 5.3.3.

Given a specific domain $\mathcal{D} = \{\mathcal{X}, \Pr(X)\}$, a *task* consists of two components: a label space $\mathcal{Y}$ and an objective predictive function $f_S(\cdot)$. The task is denoted $\mathcal{T} = \{\mathcal{Y}, f_S(\cdot)\}$. $f_S(\cdot)$ is not observed but can be learnt from the training data, which consists of pairs $\{\boldsymbol{e}_i, y_i\}$ where $\boldsymbol{e}_i \in X$ and $y_i \in \mathcal{Y}$. Also note that, in the semi-supervised setup, the label y will be known only for a subset of the entities, while in the unsupervised setup y will be unknown for every entity in the knowledge graph. The function f_S can be used to predict the corresponding label, $f_S(\boldsymbol{e})$, for a given entity $\boldsymbol{e}$ with vector representation $\boldsymbol{e}$. Therefore, the conformance score σ is equals to $f_S(\boldsymbol{e})$.

To tackle this task, we can actually use different types of learning, namely, supervised, semi-supervised and unsupervised. We provided a brief overview of each of these types in Section 2.4. Here, we focus on analysing how these types deal (or not) with the nuances of the validation problem we are trying to solve.

(1) *Unsupervised learning:* Assumes that we do not actually know the labels of any example and that data can be clustered instead given their feature vectors and a similarity metric. Evaluation in this case is more challenging since we depend on a manual review of the results. We could use this type of learning to identify whether the proximity of vector representations of semantically close entities is reflected in the clustering.

(2) *Supervised learning:* Assumes we have a large number of labelled samples that can be used to learn good models that allow to identify the correctness of an entity w.r.t. a shapes graph. If we knew the validity of a large number of entities, it would be easy to learn a model per class or a multi-class model.

(3) *Semi-supervised learning:* This type of learning assumes that we have a limited set of labelled samples and the rest of the training data is unlabelled. The number of unlabelled samples is usually larger than the number of labelled ones. In practice, this means that we can know the conformity of few entities w.r.t. the schema, and unlabelled data can be added to help improve learning. For example, we could propagate labels based on similarity, a labelled node will propagate its valid/invalid label to all its closest neighbours (according to a given similarity metric).

Independent of the learning type, we are able to learn a function f_S that provides a conformance score given an entity vector $\boldsymbol{e}$. f_S could also be used to provide a binary classifier by setting a threshold on the probabilities returned. For instance, if $f_S(\boldsymbol{e}) \geq 0.5$, we say that entity e is valid, and invalid otherwise.

5.3.3 Feature vectors extraction

For our machine learning framework, we need to represent entities using k-dimensional vectors, so that models can be trained based on the entity features and validation labels. The following are the criteria we define for choosing an approach to extract feature vectors from a knowledge graph:

(a) to capture structural patterns in the neighbourhood of entities;
(b) to generate a k-dimensional vector for any entity;
(c) to provide an interpretable representation for any entity;
(d) to be efficient to compute vectors for entities.

Several approaches have been proposed for vectorisation (also referred to as *propositionalisation*) of entities and relations in a RDF knowledge graph (Nickel, Tresp, and Kriegel, 2012; Ristoski and Paulheim, 2014, 2016a,b). In Section 3.4, we reviewed the most common approaches, which we have divided into two groups: (i) graph-based feature models, and (ii) latent feature models. From these groups, we observe that both types of approaches exploit link patterns in the graph structure; however, latent feature models require to know beforehand the whole sets of entities and relations in order to obtain vector representations for them. Latent approaches by design do not provide interpretable features. This limitation does not satisfy our criterion (c). Therefore, for our approximate validation experiments we decided to use algorithms that provide graph-based (observable) features.

Distributional Semantics (Z. S. Harris, 1954) proposes to represent the meaning of a word by its context or usage. The the so-called Distributional hypothesis states that "linguistic items with similar distributions have similar meanings." Based of the idea of Distributional Semantics, we have decided to represent an entity based on its subgraph or neighbourhood around the entity. A subgraph $\mathsf{Graph}^d(\mathcal{G}, u)$ contains all nodes and edges reachable from u following paths of length $\leq d$ (see Definition 2.4). Because the subgraph of a node u could be very large, we bound a subgraph using a sampling strategy H and denote it by $\mathsf{Graph}^d(\mathcal{G}, u)^H$. The problem of sampling subgraphs has been well studied (Cormen, Leiserson, Rivest, et al., 2009). Sampling a subgraphs for an entity u is a form of local search in the graph. Generally, two types of sampling strategies H are available for generating a subgraph $\mathsf{Graph}^d(\mathcal{G}, u)^H$: Breadth-First Strategy (BFS) and Depth-First Strategy (DFS) (Cormen, Leiserson, Rivest, et al., 2009). We define them extending the definitions in Grover and Leskovec (2016) to labelled graphs.

Breadth-first Strategy In BFS, the neighbourhood $\mathsf{Graph}^d(\mathcal{G}, u)^{BFS}$ consists of edges which can be reached immediately from the source node u. And repeats the traversal process from those neighbours.

Depth-first Strategy In DFS, the neighbourhood $\mathsf{Graph}^d(\mathcal{G}, u)^{DFS}$ consists of edges sequentially sampled at increasing distance from the source node u.

A subgraph of depth d is generated using a mix between DFS and BFS from a node. It is easy to see that the subgraph generation step also suffers from scalability issues on large-scale knowledge graphs, where a simple DFS or BFS search can become very expensive, and return non-representative subgraphs if taken separately. Applying only DFS would lead to very deep subgraphs which might not consider all neighbour relations; and applying only BFS would lead to very wide subgraphs with not enough depth. To cope with the problem of incomplete search, we apply a DFS with BFS flavour by considering the following two restrictions: (1) from a given node, we extract a maximum of 10 instances for any same relation, to avoid neglecting under represented relations in nodes with highly common relations; and (2) in each iteration of DFS, we take a sample of 100 edges, to keep a manageable final size for a subgraph. Note that such parameters are implementation decisions that can be tuned by users. In this way, we try to keep instances for all neighbour relations (even the under-represented ones, such as one-to-one relations that otherwise could be discarded), and we try to keep a representative enough subgraph while keeping an adequate size.

Formally, the previous strategy can be simulated by a *random walk* of fixed length ℓ (Lao and Cohen, 2010). For any relation path $P = \langle r_1, \ldots, r_\ell \rangle$ and a seed node $s \in Dom(P)$, a *path constrained random walk* defines a distribution $h_{s,P}$ recursively as follows. If P is the empty path, then define

$$h_{s,P} = \begin{cases} 1\,, \text{if } u = s \\ 0\,, \text{otherwise} \end{cases} \tag{5.1}$$

If $P = \langle r_1, \ldots, r_\ell \rangle$ is non empty, then let $P' = \langle r_1, \ldots, r_{\ell-1} \rangle$, and define

$$h_{s,P} = \sum_{u' \in Ran(P')} h_{s,P'}(u') \cdot \Pr(u \mid u'; r_\ell), \tag{5.2}$$

where $\Pr(u \mid u'; r_\ell) = \dfrac{r_\ell(u', u)}{|r_\ell(u', \cdot)|}$ is the probability of reaching node u from node u' with a one step random walk with edge type r_ℓ. $r_\ell(u', u)$ indicates whether there exists an edge with type $r \in \mathcal{R}$ that connect u' to u. Using these random walks we are able to build neighbourhood subgraphs $\mathsf{Graph}^d(\mathcal{G}, u)$ for every entity $u \in \mathcal{E}$. Subgraphs in this case play the role of latent shapes graph that expose the inherent structure of entities. We hypothesise that these latent shapes graph are similar for entities that belong to the same entity type (class), thus, they can be used for validating a knowledge graph.

Next, we assume that there exists a subgraph $\mathsf{Graph}^d(\mathcal{G}, u)$ for a given entity u, and we want to obtain a k-dimensional vector representation out of all subgraphs. We use a simple approach for vectorising subgraphs, we list all paths P in $\mathsf{Graph}^d(\mathcal{G}, u)$ obtained during the random walks for every entity $u \in \mathcal{E}$. To perform better random walks, we can also assume that the traversal of the knowledge graph can be done considering inverse edges as discussed in Section 2.2.1. Each path is then considered as an individual feature-value mapping with a clear interpretation. The value of a feature in such mapping can be defined in two ways:

(1) binary value encoding the existence or not of a given path, and
(2) cardinality or number of occurrences of a given path in the subgraph.

We then are able to obtain a vector for each entity in the graph based on subgraphs. In this case, the length k of the vector representations is equals to the maximum number of distinct paths found across all subgraphs. It could be possible to apply filtering techniques to the paths in a way that we obtain fixed length vectors, but we leave that as future work. We will test dimensionality reduction techniques for visualising the entities later in our experiments.

5.4 Experimental settings

Herein, we evaluate our approach on real-world datasets extracted from the Web. These are noisy and not previously validated datasets to show the effectiveness of our approach.

5.4.1 Datasets

To test our approach, we selected three datasets of different size from the Web Data Commons Microdata corpus.[7] These datasets are extracted directly from Web annotations generated by Web developers using the Schema.org vocabulary. Schema.org data contains annotations like product data, event data, or address data using a vocabulary agreed by big vendors in the search engine business, namely, Google, Microsoft, Yahoo, and Yandex.[8] Since its definition, Schema.org has been adopted by many organisations and websites (R. V. Guha, Brickley, and Macbeth, 2016).

Note that no data validation has been applied to the data generated by Web Data Commons, thus, they are likely to contain noise and errors. (Although some syntactic validation for Schema.org usage could have been applied.) With our Web scope in mind, we selected a subset of three Schema.org classes used in domains and comprising different numbers of triples: RiverBodyOfWater from the JSON-LD corpus, and Library and Continent from the Microdata corpus.[9] Each subset contains all instances of a specific class (and associated classes) as generated by the extraction tool (Meusel, Petrovski, and Bizer, 2014). Therefore, multiple classes can be found in a single dataset, where a relation exists with instances of the main class in the Schema.org annotations.

It is worth to point out that initially we analysed a couple of these classes, but many were left out of our analysis because their N-Quads[10] were ill-formed and could not be parsed properly. The errors we came across were mostly about ill-formed URIs used for naming

[7] http://webdatacommons.org/structureddata/
[8] https://schema.org/
[9] http://webdatacommons.org/structureddata/2018-12/stats/schema_org_subsets.html
[10] N-Quads is a line-based, plain text format for encoding an RDF dataset.

Table 5.2: Schema.org Microdata datasets used for approximate validation experiments. We omit the http://schema.org/ prefix from the class names. In parentheses, the values after filtering classes with more than 10 entities.

Class name	№ Triples	№ Classes	№ Typed Entities	Top-5 classes (Entity Count)
RiverBodyOfWater	37,265 (35,920)	24 (13)	7,646 (7,601)	ImageObject (1,615) ListItem (1,614) Organization (1,036) RiverBodyOfWater (556) GeoCoordinates (546)
Library	545,264 (459,377)	213 (139)	110,798 (110,574)	Library (19,523) PostalAddress (14,132) Book (7,306) WebPage (6,089) Offer (5,655)
Continent	2,198,788 (1,679,553)	48 (27)	535,828 (535,787)	City (298,536) AdministrativeArea (175,391) Place (28,904) Country (10,303) Continent (8,155)

Note: The entity count in the top-5 classes represents the number of distinct URIs found per each class type (without removing duplicated triples) and not the number of real world entities.

resources. Table 5.2 shows the characteristics of the selected datasets covering different sizes to test the scalability of our approach.

5.4.2 Test Settings

In this section we describe the experiments we have performed to demonstrate the potential of our approach.

Dataset Labelling. For labelling the entities and determine their correctness, we validated them against the Schema.org SHACL shapes graph. SHACL vocabulary (containing the shapes graph) is available for download as Turtle file from https://schema.org/docs/developers.html and http://datashapes.org/schema.ttl. To obtain a label of an entity e in a knowledge graph we have two options to frame the problem:

(a) Consider the probability of entity e being valid w.r.t. the whole Schema.org shapes graph. Such probabilities lie in the range $[0, 1]$. This will only tell us whether e is valid or not giving the conformance score σ for that. This is known as classical binary classification.

(b) Consider the probability of entity e being valid w.r.t. any of the shapes graphs in the Schema.org vocabulary. This will provide us a probability for e, $\Pr(label(e) = Class)$, with the likelihood of e to belong to a class in the Schema.org vocabulary. This is known as multi-class classification.

For our experiments, we decided to combine these two options. That is, we will validate the whole knowledge graph against the SHACL shapes graph of Schema.org and validate that

Table 5.3: Feature extraction for the different datasets. We use X and ✓ for "no" and "yes" answers.

Class name	Length	Binary/Cardinality	Inverse	№ Features	Time (H:M:S)
RiverBodyOfWater	1	Cardinality	X	50	0:00:01.230082
RiverBodyOfWater	1	Binary	X	50	0:00:01.204800
RiverBodyOfWater	1	Cardinality	✓	62	0:00:01.256128
RiverBodyOfWater	1	Binary	✓	62	0:00:01.330430
RiverBodyOfWater	2	Cardinality	X	119	0:00:01.451450
RiverBodyOfWater	2	Binary	X	119	0:00:01.382525
RiverBodyOfWater	2	Cardinality	✓	1,256	1:51:04.558503
RiverBodyOfWater	2	Binary	✓	1,256	1:51:09.186025
RiverBodyOfWater	3	Cardinality	X	176	0:00:01.433272
RiverBodyOfWater	3	Binary	X	176	0:00:01.444603
Library	1	Cardinality	X	684	0:00:22.755177
Library	1	Binary	X	684	0:00:21.936524
Library	1	Cardinality	✓	850	0:00:24.536830
Library	1	Binary	✓	850	0:00:23.407407
Library	2	Cardinality	X	1,413	0:00:27.455912
Library	2	Binary	X	1,413	0:00:27.758297
Library	3	Cardinality	X	1,629	0:00:29.973990
Library	3	Binary	X	1,629	0:00:29.628192
Continent	1	Cardinality	X	62	0:01:07.296898
Continent	1	Binary	X	62	0:01:08.496949
Continent	1	Cardinality	✓	73	0:01:19.823801
Continent	1	Binary	✓	73	0:01:09.588959
Continent	2	Cardinality	X	73	0:01:09.588959
Continent	2	Binary	X	73	0:01:10.888404
Continent	3	Cardinality	X	111	0:01:09.059367
Continent	3	Binary	X	111	0:01:08.520166

Note: Feature extraction for the Library and Continent classes using paths of length longer than 2 with inverse are not reported due to time out after 12 hours.

all entities are valid, and we will determine the probability of validity (conformance score) of entity e against every shape graph of a class.

We use pySHACL[11] one of the most complete SHACL implementations that passes 119/121 (98%) of the validation tests[12] to validate each knowledge graph and obtain its correctness. We did not find any invalid entities in our three selected datasets. However, we have noticed that the SHACL shapes graph for Schema.org contains mostly domain and range constraints, and constraints such as cardinality are not included. Here is where our latent shapes graphs obtained during the feature engineering part will play an important role for capturing cardinality and providing a better similarity of an entity w.r.t. other entities of the same class. Thus, if an entity e belongs to class x_i, then we expect to have a high probability on the i-th dimension. Similarly, if e does not belong to any of the classes in $\mathcal{G}$, then its predictions will be close to zero for every dimension. Given the classification nature of the problem, we would like to test the three different types of learning we mentioned in Section 5.3.2, namely, unsupervised, semi-supervised, and supervised learning.

Unsupervised learning. The purpose of unsupervised learning here is to demonstrate the

[11] https://github.com/RDFLib/pySHACL
[12] https://w3c.github.io/data-shapes/data-shapes-test-suite/

"goodness" of the features extracted for each entity. We would like to test two state-of-the-art methods to visualise vector representations, t-Distributed Stochastic Neighbour Embedding (t-SNE) (Maaten and G. Hinton, 2008) and UMAP (McInnes, Healy, Saul, et al., 2018). t-SNE and UMAP are dimensionality reduction techniques that can be used for visualisation of higher dimensionality vectors. They search for a low dimensional projection (embeddings) of the data that keeps similar entities as close as possible. They do not require any labelled data and remove all triples (A, `rdf:type`, B), thus, we treat them as unsupervised learning. Our aim here is to identify whether entities with similar latent shapes and validity are close to each other in the low dimensional space.

Semi-supervised learning. Once we are able to identify that entities with similar latent shape are well captured by our proposionalisation, we would like to use a semi-supervised algorithm known as Label Propagation. A label propagation algorithm takes previously known labels (entity types in our case) and propagates them to unlabelled points based on a diffusion function. The set of labelled samples is generally considered to be small compared to the set of unlabelled samples. This is a desirable characteristic if we think of propagating validation (or compliance against a schema), where we only need to know the true value for a few samples initially. We compare the performance of four algorithms (decision tree, kNN, label propagation, and naïve Bayes) when given different portions of labelled data.

Supervised learning. Here, we would like to compare the performance of a supervised learning algorithm when trained using feature vectors generated by different path configurations. We test the Decision Tree algorithm for that, which is frequently used to classify datasets with a low number of variables building interpretable classification rules for that. The generation of latent shapes proposed here has the following input: (a) the length of the paths, (b) whether to count or mark the existence of paths, (c) whether to use inverse edges in the walks or not. And the output are vectors of dimension k, where k is determined by the distinct number of paths found in the random walks.

5.5 Results and discussion

Feature extraction. For feature extraction, we tested combinations of three parameters following our criteria defined in Section 5.3.3:

(1) binary vs. cardinality of paths (generating binary or continuous vectors),
(2) length of the paths (generating paths of a length d), and
(3) inclusion of inverse edges in the paths (considering inverse relations in the path navigation).

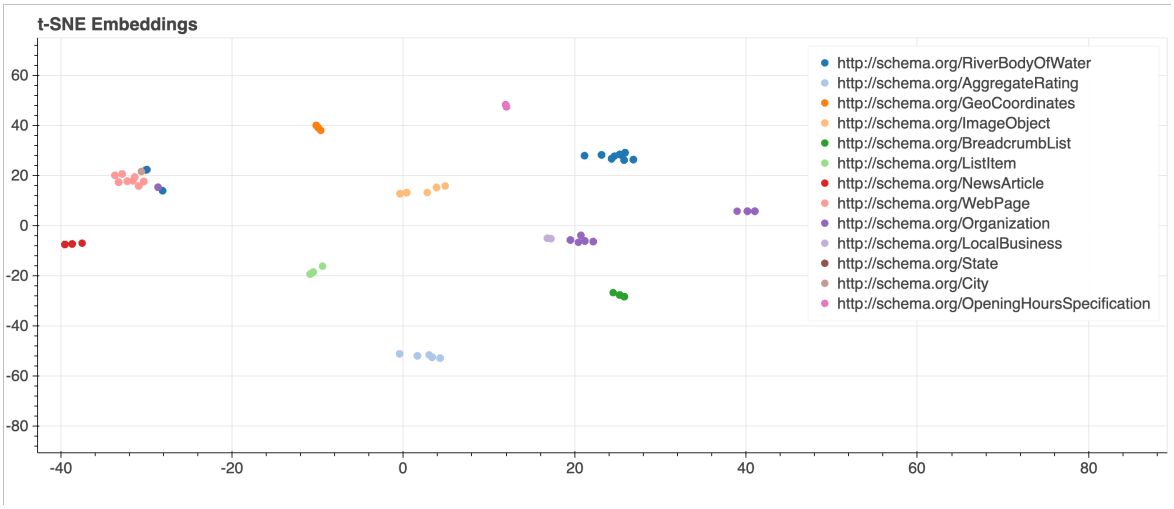
In terms of runtime, we did not notice major differences in extraction for the cardinality parameter (1) when extracting binary or cardinality features. Since paths were already enumerated, the decision of counting a path or applying an indicator function did not add any

overhead. Table 5.3 shows the different combinations of features and their runtime. For parameter (2), we have noticed some minor differences in runtime when computing longer paths in the setting that does not consider inverse relations. However, when adding inverse relations in parameter (3), we have noticed a considerable overhead due to the inverse edges when compared with the same configurations (length and count) without inverse edges. This is directly related to the depth of the graphs and the number of incoming edges for entities, which are only considered if inverse edges are added. Inverse edges add complexity to the random walks, making the process of paths enumeration very expensive. In our experiments, we passed from just above one second to almost two hours when adding inverse relations to the RiverBodyOfWater class. Not only runtime increased, but also the number of features went from 119 (without inverse edges) to 1,256 when considering paths of length 2 with inverse edges.

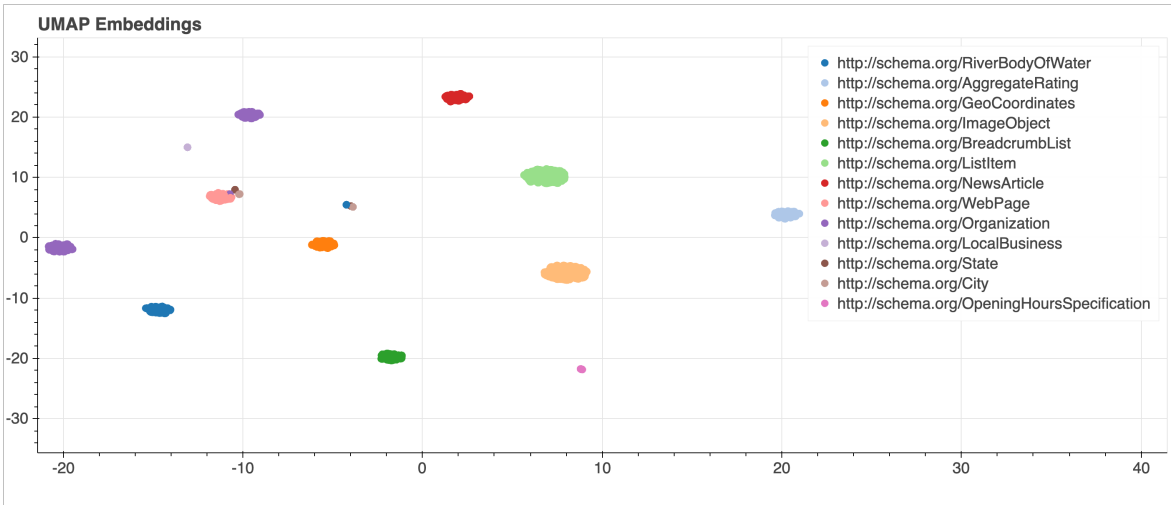
The complexity added by inverse edges is also associated to the number of relations in a knowledge graph. The Library knowledge graph has 975 different relationship, the RiverBodyOfWater knowledge graph has 62 different relationships, and the Continent knowledge graph has 95 different relationships. We observed that even by applying our pruning of relations from a given entity, it becomes time demanding to get path features using paths of length > 2 with inverse. In such a scenario, the dimensionality of the feature space increases too fast compared to the number of observed entities in a knowledge graph. The previous makes available data very sparse, since some features will only have a few training examples, making it hard for machine learning algorithms to capture any pattern. This is known as the *curse of dimensionality* (Goodfellow, Bengio, and Courville, 2016). To deal with this, we do not generate features with paths of length > 2 with inverse relations. We use unsupervised learning to identify how useful are the features we can generate for each class in a knowledge graph when limiting our extraction to paths of length ≤ 2. Future work, could go into the pre-processing of the generated dataset with the following goals: (a) filter out poorly-defined entities, whose features are mostly null because they do not have edges that allow a good classification; and (b) filter out features or paths that are rare and not present in many entities. These measures could help cleaning and improving the quality of training data, and helping to alleviate the curse of dimensionality.

Unsupervised learning. We evaluated the representation power of the latent shapes graphs using unsupervised learning techniques to reduce the dimensionality of the feature vectors. Using t-SNE and UMAP and the Euclidean distance, we reduced the dimensionality of our feature vectors to two. We apply these techniques to feature vectors obtained from paths of length 1 with inverse relations. In our experiments, UMAP scaled to the two larger datasets without problems, but t-SNE ran out of the limit time of 1 hour. Even though both methods require to compute a pairwise similarity between every two types entity in the knowledge graph, we found the UMAP implementation[13] to be much faster than the scikit-learn im-

[13] https://umap-learn.readthedocs.io/



(a) t-SNE embeddings visualisation



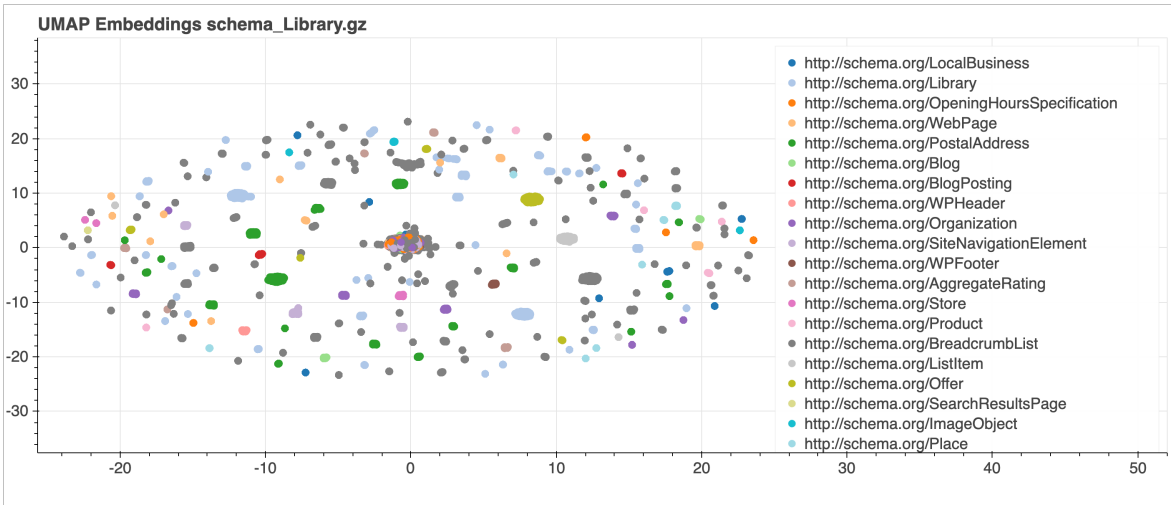
(b) UMAP embeddings visualisation

Figure 5.2: Plotting the 2D embeddings generated by t-SNE and UMAP for the RiverBodyOfWater class.

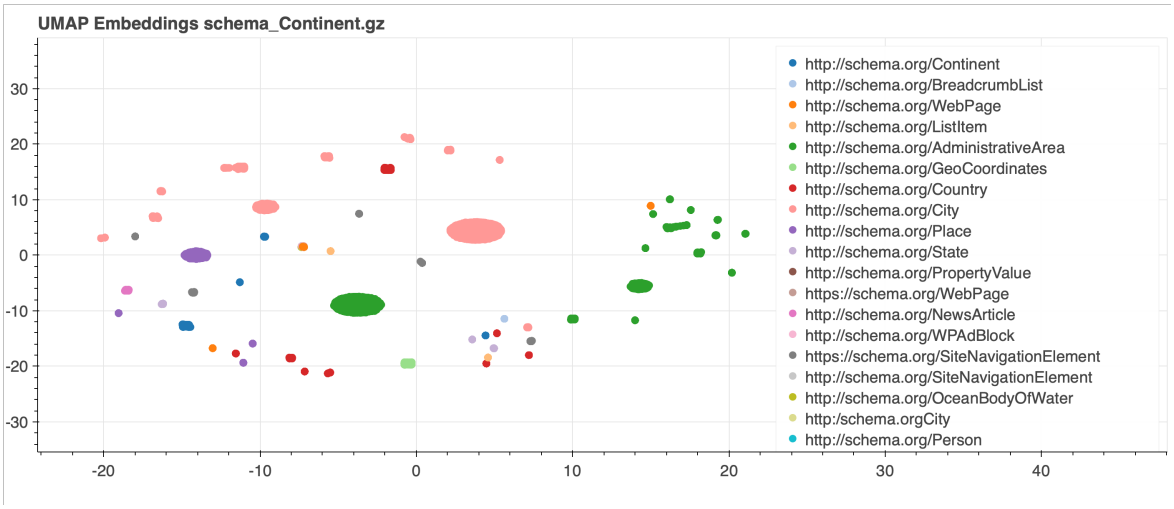
plementation of t-SNE[14]. Figure 5.2 shows a comparison between the embeddings obtained from t-SNE and UMAP for the RiverBodyOfWater class. For the larger classes Library and Continent, we only present the results from UMAP.

We can observe that the different classes present in each one of the datasets are clearly distinguishable in the RiverBodyOfWater dataset that contains the smaller number of classes. Whereas for the other two datasets, some classes are well clustered in the low-dimensionality space and other are smaller clusters distributed in different areas. In Figure 5.2, it is nice to see how related classes are close to each other in the 2D space, for example, `schema:LocalBusiness` is close to `schema:Organization` and `schema:State` is close to `schema:City`. We can then assume that the features are good representations of the enti-

[14] https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html



(a) UMAP embeddings visualisation for Library



(b) UMAP embeddings visualisation for Continent

Figure 5.3: Plotting the 2D embeddings generated by UMAP for the Library and Continent classes. Note that only the first 19 classes are displayed in the legend.

ties in each class. This is a clear example of the latent schema graphs concept we analyse in this thesis.

Supervised learning. We use the previously extracted feature vectors in each dataset to classify entities in each class. Overall, the model training part took much less time than the feature generation that we stored after computing them. (This is mainly because of the complexity added by the random walks.) For this, we apply the decision tree classification algorithm to the data and obtain the corresponding confusion matrix. We divided the data using a 80–20% split, where 80% is for training and 20% is left for testing. Figure 5.4 shows the confusion matrix obtained by evaluating the decision tree model in the Library dataset using feature vectors with length 3, no inverse edges, and cardinality features. Similar results were obtained using the other feature vectors—we obtained similar results even using

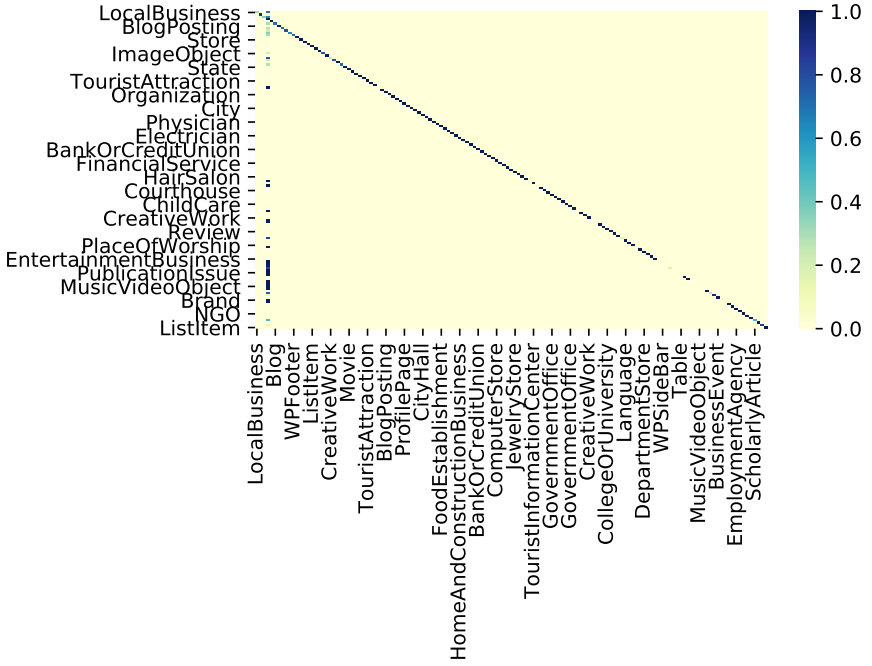
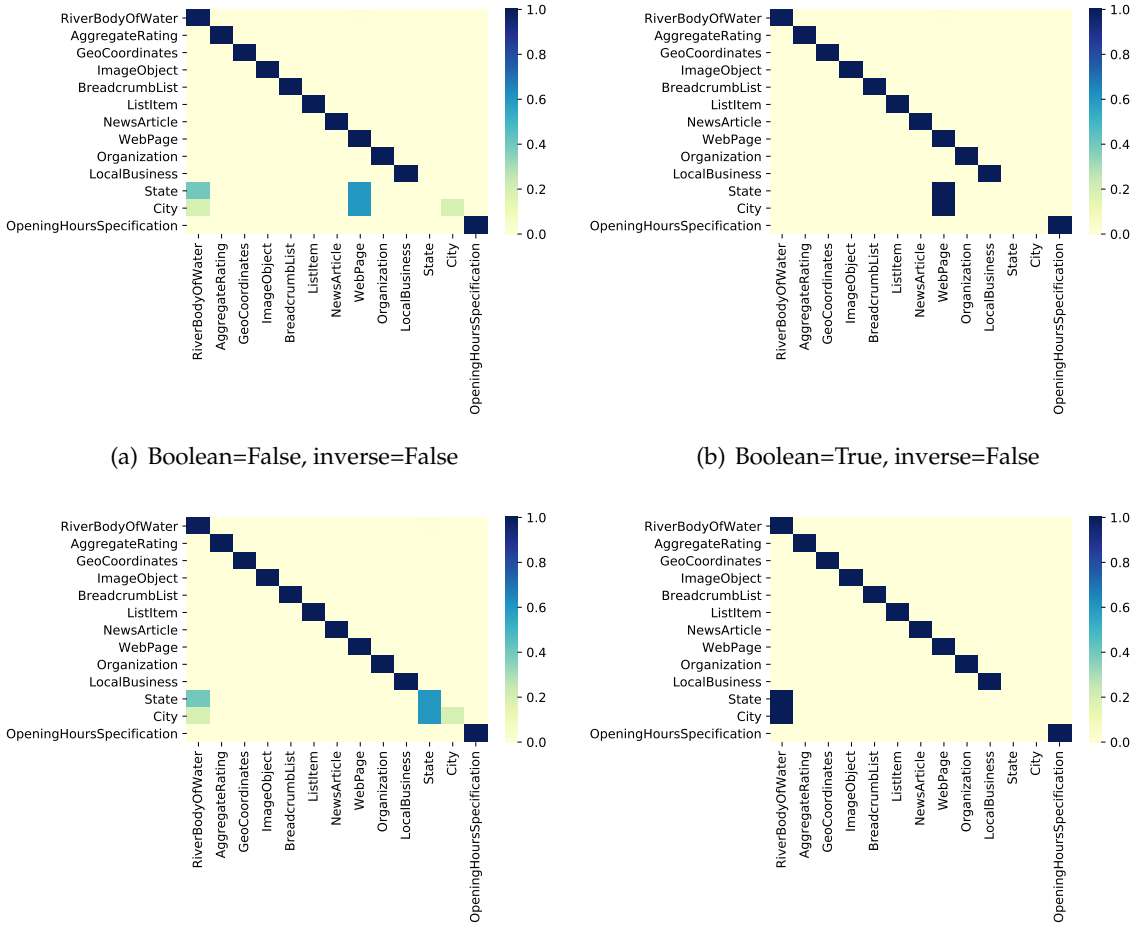


Figure 5.4: Confusion matrix with the results of paths of length 3, no inverse edges, and cardinality features over Library dataset.

the simpler configurations with paths of length 1, no inverse edges, and binary features. The worst results were observed in entities predicted as `schema:WebPage`, which originally only contains 3,044 entities. Most entities that do not appear with any differentiator feature were then predicted as Web pages. We observed that the definition of `schema:WebPage` instances is skewed, because relations in this class such as `schema:WebPage:url`, `schema:WebPage:mainContentOfPage`, and `schema:WebPage:name` are only defined for less than 23% of the instances, making it a default class for instances not well defined. Thus, the model wrongly classified most of these instances. Likewise, for the Continent dataset, instances of not well-defined classes were misclassified as `schema:City`.

Furthermore, we analysed the effects of the different configurations for feature extraction. We used the RiverBodyOfWater dataset, which contains a small number of classes to visually these effects. Figure 5.5 shows the confusion matrices obtained by different configurations using paths of length 1. The best macro-F1 of 0.9066 is given by the feature vectors using cardinality paths and inverse edges. The second best macro-F1 of 0.8699 is given using cardinality features and no-inverse edges. These results highlight the importance of cardinality for generating features for entities in a knowledge graph. Cardinality provides a better representation in the cases observed here. The two classes with most misclassified examples were `schema:State` and `schema:City`, which were classified by the models as `schema:WebPage` and `schema:RiverBodyOfWater`. That is an effect of poorly described state and city entities in the dataset.

Semi-supervised learning. In a real scenario, we can assume that the labels would be avail-



(a) Boolean=False, inverse=False

(b) Boolean=True, inverse=False

(c) Boolean=False, inverse=True

(d) Boolean=True, inverse=True

Figure 5.5: Confusion matrices given by the decision tree over the RiverBodyOfWater dataset with paths of length 1 and different settings of inverse and cardinality features.

able for a small number of cases. Here, we emulate the effects of having different ratios of unlabelled data on the classification problem. For that we measured macro-F1 metric using incremental ratios of unlabelled data between 0.1 and 0.9. Figure 5.6 shows the results of four machine learning algorithms (decision tree, kNN, label propagation, and naïve Bayes) when given limited labelled data. As expected, the performance decreases when the size of unlabelled examples increases. However, we also observed that the drop in performance was not very severe—at 90% of unlabelled data we still obtain macro-F1 above 0.75. These results demonstrate the predictive power of the feature vectors based on latent shapes.

These experimental results using Schema.org data showed that our approach is significantly more efficient than traditional validation, allowing users to validate noisy knowledge graphs. We found that the feature extraction part could still be improved to make it more scalable. A possibility is to extend the parallelisation proposed by Kyrola (2013) to labelled graphs. For scenarios like this, it would also be interesting to test latent feature models to obtain the feature vector representations as future work.

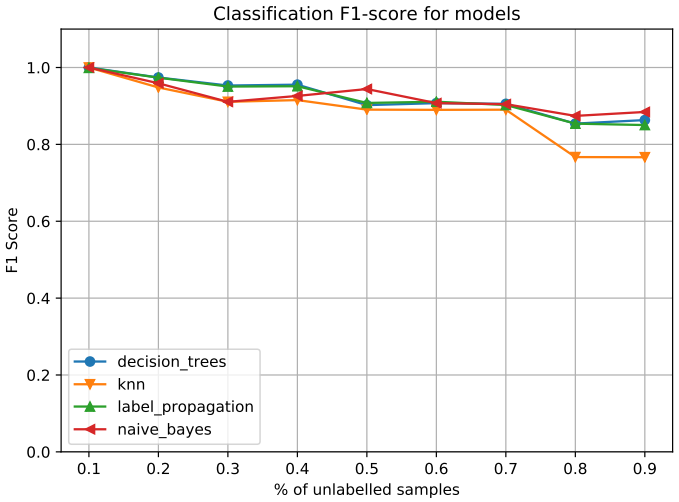


Figure 5.6: Macro-F1 measured over different percentages of unlabelled data for the RiverBodyOfWater dataset.

5.6 Summary

The idea of approximated approaches to measure the (in-)consistency of a knowledge graph has been already mentioned in Labra Gayo, García-González, Fernández-Alvarez, et al. (2019). In this chapter, we have demonstrated some initial results of an approximate solution for validating the structure of knowledge graphs (Section 5.1), based on path expressions and relation cardinality. We have taken inspiration from the cardinality patterns and shapes analysis reviewed in Chapter 3 to propose the use of multi-class machine learning to tackle the approximate validation task introduced here. In Section 5.2, we have reviewed the related work for this problem and we have described the details of our approach in Section 5.3. We developed a solution that extracts local patterns for each node and give a vector representation that can then be used to train machine learning models. We evaluated different approaches to extract such vector representation and identified that features generated from short paths of length 1 considering cardinality and inverse relations provide some of the best performance results for the classification task (Section 5.4). We discuss our results in Section 5.5 and conclude that the use of machine learning algorithms and vector representations of entities in knowledge graphs facilitate the development of more efficient solutions for mining tasks. The problem of validating entities that belong to multiple classes is still open and we think multi-label learning approaches could provide new possibilities to reach a full validation.

Part III

Knowledge Graph Mining Applications

CHAPTER 6

Knowledge Graphs to Improve Adverse Drug Reactions Prediction

Contents

In this chapter, we study the use of knowledge graphs to train prediction models in the Bioinformatics domain. This study will help us answering the RQ (3) of this thesis. More specifically, we address the problem of predicting *adverse drug reactions* (ADRs)—also known as side effects—using machine learning models that are trained with feature vectors extracted from a knowledge graph. ADRs can cause significant clinical problems and represent a major challenge for public health and the pharmaceutical industry. Early discovery of potential ADRs can limit their effect on patient lives and also make drug development pipelines more robust and efficient. Reliable *in silico* (performed on computer) predictions of ADRs can be helpful in this context, and thus, it has been intensely studied. Recent works using machine learning have achieved promising results using relational data sources, but we believe knowledge graphs can provide more interesting features never used before. We review the most relevant works and propose a multi-label approach that benefits from knowledge graph link features, achieving new state-of-the-art results.

6.1 Problem statement

Timely identification of adverse drug reactions (ADRs) is highly important in the domains of public health and pharmaceutical industry. It can substantially limit the detrimental effect of the adverse reactions on patient lives. Discovering potential adverse reactions in the early stages of research can also make the drug development pipelines more robust and efficient. Therefore a reliable automated prediction of adverse drug reactions is much desired and has become an intensely studied research topic in the last fifteen years.

During a drug development process, pharmacology profiling leads to the identification of potential drug-induced biological system perturbations including primary effects (intended drug target interactions) as well as secondary effects (off-target drug interactions) mainly responsible for ADRs (Bowes, Brown, Hamon, et al., 2012). Computational (a.k.a. in silico) prediction of ADRs during the development cycle of a drug (before the drug is licensed for use) can reduce the cost of drug development and provide a safer therapy for patients. However, current state-of-the-art methods suffer from limitations: They work with datasets that have been manually pre-processed, and the prediction methods are adapted to the experimental data in a very focused manner.

Several KGs have been made available to represent data in the life sciences domain, including biological interaction-based features for drugs such as drug target, pathways, enzymes, transporters or protein-protein interactions (Mizutani, Pauwels, Stoven, et al., 2012; Yamanishi, Pauwels, and Kotera, 2012; L.-C. Huang, X. Wu, and J. Y. Chen, 2013). Although biomedical KGs are abundant nowadays, they also suffer from quality issues such as incompleteness. In Muñoz, Nováček, and Vandenbussche (2016, 2019), we investigated the use of knowledge graphs as a convenient uniform representation of relevant biomedical data. We use the notion of neighbourhood mixtures in a knowledge graph to generate feature vectors for drugs, which are used to learn multi-label models. Casting the problem as multi-label learning, we are able to account that one drug can have multiple ADRs at the same time. We hypothesise that the information provided by neighbourhood mixtures can help to improve completeness of the *side_effect* relation between drugs and ADRs.

We follow a drug profile approach and state the prediction problem as follows:

Adverse drug reactions prediction

Input: a set $\mathcal{D} = \{(\boldsymbol{x}_i, Y_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i \in \mathcal{X}$ are drugs feature vector and $Y_i \in \mathcal{Y}$ are adverse drug reactions

Output: a model $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ that predicts the *side_effect* relation between drugs and ADRs

The representation of drugs usually consist of different feature sets, such as chemical, biological, and phenotypic spaces. Additionally, in this work, we consider drug representations obtained from a knowledge graph and compare the prediction results.

6.2 Related work

Generally, a drug can have multiple adverse reactions, thus, the ADR prediction problem can be naturally formulated as a multi-label learning problem (Tsoumakas and Katakis, 2006). Multi-label learning addresses a special variant of classification in machine learning where multiple labels (i.e., ADRs) are assigned to each example (i.e., drug).

The ADR prediction problem can then be solved either by transforming it into a set of binary classification problems, or by adapting existing machine learning techniques to their multi-label setting.[1] Most of the current ADR prediction methods, however, do not fully exploit the convenient multi-label formulation, as they simply convert the main problem into a set of binary classification problems (M. Zhang and Z. Zhou, 2014). This is problematic for two main reasons. Firstly, transforming the multi-label problem into a set of binary classification problems is typically very computationally expensive for large numbers of labels (which is the case in ADR prediction). Secondly, using binary classifiers does not accurately model the inherently multi-label nature of the main problem.

W. Zhang, F. Liu, L. Luo, et al. (2015) proposed a multi-label learning method called FS-MLKNN that integrates feature selection and k-nearest neighbours. Unlike most previous works, W. Zhang, F. Liu, L. Luo, et al. (2015) method does not generate binary classifiers per label, but uses an ensemble learning instead. They performs an iterative feature selection using an expensive evolutionary algorithm to filter out irrelevant features. Different to them, we do not perform feature selection to modify the train data; and we use standard multi-label learning models. We follow the philosophy of algorithm adaptation: fit algorithms to data (M. Zhang and Z. Zhou, 2014). Table 6.1 lists the reviewed approaches along with the features they use.

6.3 ADRs prediction using knowledge graphs

We model the prediction of ADRs for a drug as a multi-label learning problem, and use knowledge graphs to build feature matrices that can be passed to the learning methods. Our drug centric approach aims to compute a similarity graph represented as an adjacency matrix, which can serve to compute drug-drug similarity. Using the assumption that similar drugs share similar sets of ADRs, we use the similarity between drugs to propagate ADRs between very similar drugs. To build the adjacency matrix from a knowledge graph, we use the previously introduced definition of neighbourhood mixture and compute similarity as follows: Given a knowledge graph $\mathcal{G}$, let $\pi^{\leftarrow}(\mathcal{G}, u) = \left\{(r, v^{-1}) \mid \exists r \in \mathcal{R} \exists v \in \mathcal{E},\ (v, r, u) \in \mathcal{G}\right\}$ be the function that extracts the finite set of incoming relations r from v to u, and let $\pi^{\rightarrow}(\mathcal{G}, u) = \{(r, v) \mid \exists r \in \mathcal{R} \exists v \in \mathcal{E},\ (u, r, v) \in \mathcal{G}\}$ be the function that extracts the finite set of outgoing relations r from u to v. Hence, we can define the function to extract the neigh-

[1] See https://en.wikipedia.org/wiki/Multi-label_classification for more details and a list of examples.

Table 6.1: Multi-source feature sets used by state-of-the-art methods.

	Atias and Sharan (2011)	Pauwels, Stoven, and Yamanishi (2011)	Mizutani, Pauwels, Stoven, et al. (2012)	Yamanishi, Pauwels, and Kotera (2012)	M. Liu, Yonghui Wu, Yukun Chen, et al. (2012)	Bresso, Grisoni, Marchetti, et al. (2013)	L.-C. Huang, X. Wu, and J. Y. Chen (2013)	Jahid and Ruan (2013)	Zhang et al. (2015); (Dec. 2016); (2016)	Rahmani, Weiss, Méndez-Lucio, et al. (2016)	Muñoz, Nováček, and Vandenbussche (2016)
Chemical space											
Drug compound substructure	✓	✓	✓	✓	✓	✓	✓	✓	✓		
Biological space											
Drug target				✓	✓		✓		✓	✓	✓
Pathway			✓		✓				✓		✓
Enzymes					✓				✓		✓
Transporters					✓				✓		✓
PPi							✓			✓	
Phenotypic space											
Indication					✓				✓		✓
Cell line response	✓										

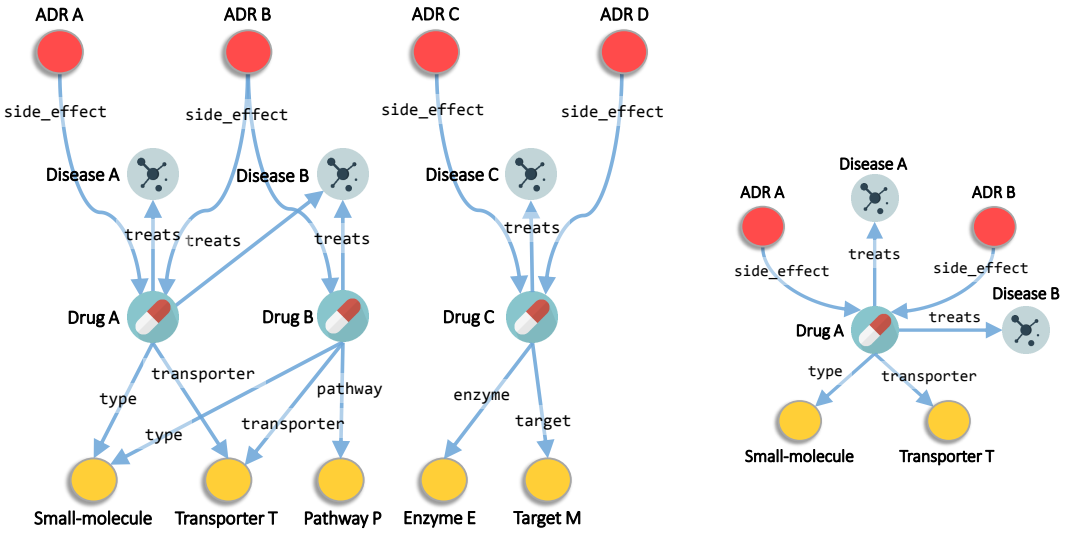
bourhood mixture for an entity u in the knowledge graph $\mathcal{G}$ as:

$$\mathcal{N}_u = \pi^{\leftarrow}(\mathcal{G}, u) \cup \pi^{\rightarrow}(\mathcal{G}, u). \tag{6.1}$$

Example 6.1

Consider the example knowledge graph in Figure 6.1(a) consisting of three drugs and their relationships with other entities. The neighbourhood mixture for *Drug A* is given by a set—depicted as a subgraph of $\mathcal{G}$ in Figure 6.1(b):

$$\begin{aligned}\mathcal{N}_{DrugA} = \{&(ADR\ A, \textit{side\_effect}^{-1}), (ADR\ B, \textit{side\_effect}^{-1}),\\ &(Disease\ A, \textit{treats}), (Disease\ B, \textit{treats}),\\ &(Small\_molecule, \textit{type}), (Transporter\ T, \textit{transporter})\}\end{aligned}$$



(a) Knowledge graph.

(b) Neighbourhood mixture for *Drug A*.

Figure 6.1: Example knowledge graph and neighbourhood mixture with drug-related entities and their relations.

Example 6.2

Consider the KG in Figure 6.1(a), and the three drugs *Drug A*, *Drug B*, and *Drug C*, with corresponding neighbourhood mixtures $\mathcal{N}_{DrugA}$, $\mathcal{N}_{DrugB}$, and $\mathcal{N}_{DrugC}$, extracted as shown in Example 6.1 for $\mathcal{N}_{DrugA}$. We have 12 unique pairs in the mixtures, which are then interpreted as 12 different features. The design matrix is then,

$$\left(\begin{array}{c|cccccccccccc} \textit{Drug A} & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \textit{Drug B} & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \textit{Drug C} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{array}\right).$$

Intuitively, the more features two entities have in common, the more similar they are. This is the motivation of our algorithm KG-SIM-PROP (Muñoz, Nováček, and Vandenbussche, 2016), which is a similarity-based method that propagates the labels of the k nearest neighbours to a drug using the 3w-Jaccard similarity metric. Similarly to kNN, KG-SIM-PROP requires a similarity matrix to work. For all other algorithms, we can feed directly the design matrix. KG-SIM-PROP obtains prediction scores for each drug using a weighted average of the labels coming from the nearest neighbours. Let $\boldsymbol{x}_i$ be a drug, k be the number of neighbours considered, $\boldsymbol{s}$ the similarity vector of $\boldsymbol{x}_i$ to each of the k neighbours, and $\boldsymbol{t}$ the vector of labels assigned to neighbour drugs. We compute the vector of ADRs predictions for $\boldsymbol{x}_i$ as follows:

$$p(\boldsymbol{x}_i) = \frac{\boldsymbol{s} \cdot \boldsymbol{t}}{\sum_i \boldsymbol{s}_i}. \tag{6.2}$$

To construct a similarity matrix, we apply pairwise similarity between the feature vector of one drug and the feature vector of all other drugs—this can be computed in parallel. 3w-Jaccard (Choi, Cha, and Tappert, 2010) similarity is used since it assigns higher weight to common features, and lower weight to discriminant features, i.e., those only present in one drug. The 3w-Jaccard between two drugs $\boldsymbol{x}_i$, and $\boldsymbol{x}_j$ is defined as:

$$S_{\text{3W-JACCARD}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{3a}{3a + b + c}, \tag{6.3}$$

where $a = \left|\mathcal{N}_{\boldsymbol{x}_i} \cap \mathcal{N}_{\boldsymbol{x}_j}\right|$, $b = \left|\mathcal{N}_{\boldsymbol{x}_i} - \mathcal{N}_{\boldsymbol{x}_j}\right|$, and $c = |\mathcal{N}_{\boldsymbol{x}_j} - \mathcal{N}_{\boldsymbol{x}_i}|$, with $0 \leq S_{\text{3W-JACCARD}} \leq 1$. We construct the $\boldsymbol{W}$ similarity matrix using the 3w-Jaccard similarity between every pair of drugs: the similarity graph is represented as an adjacency matrix $\boldsymbol{W} \in \mathbb{R}^{N \times N}$ with $w_{ij} = w_{ji} = S_{\text{3W-JACCARD}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $w_{ii} = 0$.

Now, our learning problem can be formally stated as: Given a drug $\boldsymbol{x}$ and a finite-size vector Y with its initially known adverse reactions (i.e., labels), seek to find a discriminant function $f(\boldsymbol{x}, Y) = \hat{Y}$, where $\hat{Y}$ is a finite-size vector representation of the labelling function $\hat{Y} = [f(\boldsymbol{x}, y_1), \ldots, f(\boldsymbol{x}, y_Q)]^\top$ for $y_i \in \mathcal{Y}$.

6.4 Biomedical knowledge graphs

Various publicly available data sources can be used to extract drug profiles, and define similarity between drugs (Tan, Y. Hu, X. Liu, et al., 2016). Each data source describes a specific aspect of the pharmacological space of a drug such as its chemical, biological or phenotype properties. For instance, SIDER database (Kuhn, Letunic, L. J. Jensen, et al., 2015) presents information of side effects and indication for marketed drugs. PubChem Compound data (Kim, Thiessen, Bolton, et al., 2015) contains chemical structure description of drugs. DrugBank (Law, Knox, Djoumbou, et al., 2014) provides detailed information about drugs such as their binding proteins and targets, enzymes or transporters thus informing on drugs' mechanism of action and metabolism. KEGG Genes, Drug, Compound and Disease databases (Kanehisa, Furumichi, Tanabe, et al., 2017) describe further information about molecular interaction of drugs and their signalling pathways.

Several state-of-the-art methods have published the results of their data integration activity using multiple data sources, including the ones already mentioned. Table 6.2 describes the characteristics of datasets used by previous works in the ADR prediction task, namely, Liu's (M. Liu, Yonghui Wu, Yukun Chen, et al., 2012), Bio2RDF (Muñoz, Nováček, and Vandenbussche, 2016), and SIDER 4 (W. Zhang, F. Liu, L. Luo, et al., 2015) datasets. Liu's and SIDER 4 datasets are a collection of matrices with features in the chemical, biological and phenotypic spaces of drugs, combined with information on their associated ADRs. Additionally, we extracted the most recent ADRs for newly marketed drugs from Aeolus, which is a curated and annotated machine-readable version of FAERS database meant to facilitate research in drug safety (Banda, Evans, Vanguri, et al., 2016). In particular, the cases

Table 6.2: Characteristics of datasets used in the ADR prediction task.

Dataset	№ drugs	№ side effects
Liu's dataset	832	1,385
Bio2RDF dataset	1,824	5,880
SIDER 4 dataset	1,080	5,579
Aeolus dataset	750	181

Table 6.3: Number of triples in the Bio2RDF datasets used in our experiment.

Knowledge Graph	Content	№ triples
DrugBank	Drug types, chemical information	5,151,999
SIDER	Side effects of drugs	5,578,286
KEGG	Drugs, genes and pathway maps	4,387,541

(i.e., ADR events) in the FAERS reports are deduplicated and the drug and outcome (i.e. effect) concepts are mapped to standard vocabulary identifiers (RxNorm and SNOMED-CT, respectively). We use Aeolus to generate an updated version of the SIDER 4 dataset that includes also the latest ADRs as observed in the population

The Bio2RDF project aims to make available biomedical databases in the form of RDF (Belleau, Nolin, Tourigny, et al., 2008; Dumontier, Callahan, Cruz-Toledo, et al., 2014). Bio2RDF makes available over 30 databases including PubChem, DrugBank, SIDER and KEGG. Here, we use the release 4 of Bio2RDF and represent its data using a knowledge graph. Figure 6.2 shows a fragment of the Bio2RDF knowledge graph that integrates three databases, namely, DrugBank, SIDER and KEGG. Usually, connections between databases are made using identifiers such as PubChem compound or Chemical Abstracts Service (CAS) number.

Table 6.3 shows the characteristics of the Bio2RDF datasets considered to build knowledge graphs. Using the method described in Section 6.3, we can build design matrices from Bio2RDF datasets, which are similar to the ones provided in Liu's and SIDER 4 datasets.

6.5 Experimental settings

To test the use of neighbourhood mixtures and knowledge graphs for predicting ADRs, we select different multi-label learning models. These models learn how to assign sets of ADRs (labels) to each drug (example). We investigate state-of-the-art multi-label learning models that accept multiple labels, namely, Decision Trees, Random Forests, Nearest Neighbours (kNN), Multi-Layer Perceptron, and KG-SIM-PROP which we proposed in Muñoz, Nováček, and Vandenbussche (2016). The above models inherently support multi-labels, however, other models such as Logistic Regression can be adopted following the *one-vs-all* strategy, where the system builds as many binary classifiers as input labels and samples having label y are considered as positive or negative otherwise.

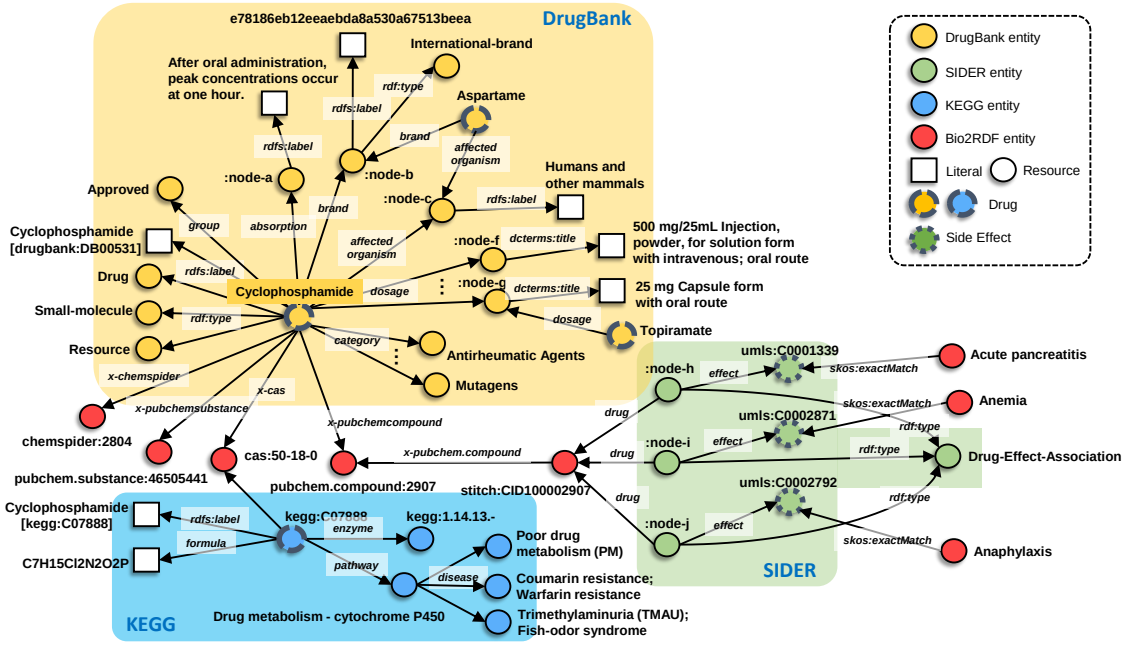


Figure 6.2: A Bio2RDF fragment around the cyclophosphamide drug, showing the connections between three main (colour-coded) databases: DrugBank, SIDER and KEGG.

We perform a comparison between the above models in terms of performance based on several multi-label ranking evaluation metrics (see Section 2.6.2). The performance of all models is evaluated using a 5-fold cross-validation (CV). First, all drugs are randomly split into five 5 equal sized subsets. Then, for each of the k folds, one part is held-out for testing and the learning algorithm is trained on the remaining four parts. In this way, all parts are used exactly once as validation data. The selection of best hyperparameters for each model is performed in each fold on the training set during the 5-fold CV, and the best model is applied over the test set for validation. The five validation results are then averaged over all rounds. We use the average precision (AP), area under the curve (AUC-ROC), and the area under the precision-recall curve (AUC-PR) to evaluate the models, because they can be used to evaluate models regardless of any threshold. However, because of the existing unbalance of the labels (i.e., an ADR is more commonly found as a negative value than as a positive one among drugs), the AUC-PR gives a more informative picture of the models performance (Davis and Goadrich, 2006). Thus, we set AUC-PR as our target metric (in the grid searches) for each of the rounds. Additionally, we compute other label-based metrics (M.-L. Zhang and Z.-H. Zhou, 2006; Tsoumakas, Katakis, and Vlahavas, 2010), namely, one-error, coverage, and ranking loss. The last type of measures we use are the general ranking evaluation metrics hits at K (Hits@K), and precision at K (P@K). Among the measures we used, the Hits@K and P@K are arguably the most accurate scores in terms of evaluating the benefit of ADR discovery for certain types of end-users like clinical practitioners.

In Figure 6.3, we show a typical flow chart for the processes of training, testing and evaluating machine learning models. For a given model, its output is used to generate ADR

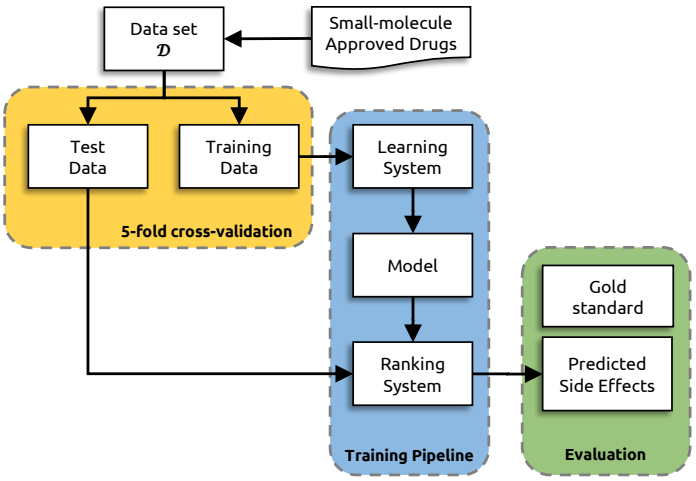


Figure 6.3: Machine learning flow chart for training and testing of a model.

predictions. These predictions are evaluated using Liu's, SIDER 4, Bio2RDF, and Aeolus datasets as gold standard.

6.6 Results and discussion

All models are implemented using the Scikit-Learn Python package (Pedregosa, Varoquaux, Gramfort, et al., 2011). We compare the models with the state-of-the-art results in four different settings, involving the different datasets.

6.6.1 Comparison on Liu's dataset

Liu's dataset was proposed in T. Liu (2011), includes multi-source data with different types of features about drugs, and has been considered as a benchmark in W. Zhang, F. Liu, L. Luo, et al. (2015) and W. Zhang, Yanlin Chen, Tu, et al. (2016). We compare the results reported in W. Zhang, Yanlin Chen, Tu, et al. (2016) for four existing methods (Liu's method, FS-MLKNN, LNSM-SMI, and LNSM-CMI) with six multi-label learning models selected by us. This comparison aims to demonstrate the flexibility and potential of using knowledge graphs with neighbourhood mixture for the link prediction task.

Table 6.4 shows the values of evaluation metrics for each model. We found out that the methods FS-MLKNN, LNSM-SMI and LNSM-CMI recently proposed by Zhang et al. (W. Zhang, F. Liu, L. Luo, et al., 2015; W. Zhang, Yanlin Chen, Tu, et al., 2016) perform best on the Liu's dataset; the multi-layer perceptron, however, comes second by rather small margin in all but one metric. The FS-MLKNN, LNSM-SMI and LNSM-CMI methods require large numbers of neighbours to work properly (400 as reported). The KG-SIM-PROP and kNN

Table 6.4: Comparison of the models' performance on the Liu's dataset. ($\star$) is our model proposed in Muñoz, Nováček, and Vandenbussche (2016) and (†) are traditional models with default parameters.

Model	Evaluation Criterion					
	AP ↑	AUC-PR ↑	AUC-ROC ↑	R-Loss ↓	One-Error ↓	Cov-Error ↓
Liu's method	0.2610	0.2514	0.8850	0.0927	0.9291	837.4579
FS-MLKNN	0.5134	0.4802	0.9034	0.0703	0.1202	795.9435
LNSM-SMI	0.5476	0.5053	0.8986	0.0670	0.1154	789.8486
LNSM-CMI	0.5329	0.4909	0.9091	0.0652	0.1250	776.3053
KG-SIM-PROP $\star$	0.4895±0.0058	0.4295±0.0078	0.8860±0.0075	0.1120±0.0139	0.1610±0.0164	1100.9985±65.8834
kNN†	0.5020±0.0078	0.4417±0.0081	0.8892±0.0085	0.1073±0.0053	0.1538±0.0181	1102.3548±41.4641
Decision Trees†	0.2252±0.0137	0.1989±0.0181	0.6634±0.0316	0.6519±0.0242	0.5493±0.0374	1377.1316±8.3936
Random Forests†	0.4626±0.0163	0.4331±0.0261	0.8342±0.0218	0.2525±0.0176	0.2007±0.0154	1284.3111±27.0454
MLP†	0.5196±0.0069	0.4967±0.0204	0.9003±0.0057	0.0874±0.0009	0.1454±0.0166	954.0372±22.2870
Linear Regression†	0.2854±0.0088	0.2595±0.0196	0.6724±0.0232	0.6209±0.0137	0.4267±0.0103	1380.0763±4.0209

Note: For each metric, we report the standard deviation values (when available). The values for the first four models were taken from (W. Zhang, Yanlin Chen, Tu, et al., 2016) The evaluation metrics are AP, AUC-PR curve, AUC-ROC, R-loss, one-error and Cov-error. ("↑" indicates that the higher the metric value the better, and "↓" indicates that the lower the metric value, the better.) Coloured values represent the best performing method across a given metric.

methods can work with as little as 30 neighbours which makes them more applicable to sparse datasets. The better results of LNSM-SMI and LNSM-CMI may be attributed to their consideration of neighbourhood as an optimisation problem via the linear neighbourhood similarity used. This provides them more accuracy on the similarity computation, but at the cost of efficiency. We report that FSMLKNN was the slowest method with more than two weeks running time on a single machine. This is mainly due to its multiple feature selection steps based on genetic algorithms. From the multi-label ranking methods, the slowest was kNN with 13 hours and 18 minutes, followed by linear regression with 9 hours and 26 minutes. Both multi-layer perceptron and KG-SIM-PROP took ca. 2 hours and 16 minutes, while the decision trees were the fastest with only 16 minutes.

On the other hand, KG-SIM-PROP and kNN employ widely used off-the-shelf similarity metrics between feature vectors to determine the neighbourhoods. Conversely, methods that do not consider a similarity, namely, decision trees, random forests, and linear regression, are among the worst performing methods.

In addition to the metrics reported in previous works, we report the ranking performance of the multi-label learning to rank methods in Table 6.5. Results show that multi-layer perceptron gives the best rankings across all metrics. This may indicate that non-linear methods (such as deep neural nets) are better suited to the ADR prediction problem. Deep learning methods have shown to excel in applications, where there is an abundance of training data, and sources such as Bio2RDF could serve for this purpose. The use of deep learning methods for the prediction of ADRs is still an open problem.

Table 6.5: Ranking performance of the proposed models on the Liu's dataset.

Model	Evaluation Criterion						
	P@3	P@5	P@10	HITS@1	HITS@3	HITS@5	HITS@10
KG-SIM-PROP	0.9333±0.1333	0.8400±0.2332	0.9200±0.1166	0.8390±0.0164	2.4351±0.0240	3.8691±0.0671	7.0734±0.0746
*k*NN	0.9333±0.1333	0.9200±0.0980	0.9400±0.0800	0.8450±0.0173	2.4568±0.0316	3.9027±0.0452	7.1744±0.0581
Decision Trees	0.4667±0.2667	0.4400±0.2653	0.4800±0.1470	0.4171±0.0176	1.1971±0.0570	1.9651±0.0940	3.8076±0.1941
Random Forests	0.9333±0.1333	0.9200±0.0400	0.9200±0.0400	0.8101±0.0088	2.3353±0.0594	3.7451±0.0779	6.9434±0.0982
MLP	1.0000±0.0000	0.9600±0.0800	0.9600±0.0490	0.8546±0.0166	2.4676±0.0295	3.9773±0.0544	7.3633±0.1451
Linear Regression	0.3333±0.2981	0.4000±0.1265	0.4400±0.1347	0.5745±0.0469	1.6262±0.0716	2.6394±0.0782	5.1851±0.0823

Note: The evaluation metrics are P@X (precision at 3, 5, and 10), and HITS@X (hits at 1, 3, 5, and 10). (For all metrics, the higher the value of the metric, the better.) Coloured values represent the best performing method across a given metric.

6.6.2 Comparison on Bio2RDF dataset

Several authors have found that combining information from different sources can lead to improve the performance of computational approaches in bioinformatics (see, among others, Polikar (2006) and R. Yang, C. Zhang, R. Gao, et al. (2015)). We study the use of KGs such as Bio2RDF to easily generate feature sets combining diverse data sources. In order to test this hypothesis, we use the DrugBank, SIDER and KEGG datasets from Bio2RDF, and build two versions: *version 1 (v1)* containing only DrugBank and SIDER, and *version 2 (v2)* containing the previous datasets plus KEGG. Tables 6.6 and 6.7 shows the performance of six multi-label learning methods using the set of 832 drugs and 1,385 side effects from Liu's dataset, but replacing the feature vectors of drugs with those extracted from the Bio2RDF v1 (or Bio2RDF v2) dataset. Originally, Liu's dataset contained a set of 2,892 manually integrated features coming from six sources. These are replaced by 30,161 and 37,368 features in Bio2RDF v1 and v2, respectively. In the evaluation of the models using Bio2RDF v1 KG, we obtained slightly lower results than for Bio2RDF v2 KG, showing that the addition of different information—including different sources such as KEGG—can help to improve the prediction power or the models.

Results using both version of Bio2RDF show that the methods perform better with the Bio2RDF features, than with the original Liu's dataset features, confirming our assumption that combination of various feature sources may increase the performance. This can be explained by the fact that Bio2RDF provides a richer representation of drugs and their relationships than the traditional feature sets. This is an important finding, as the Bio2RDF features can be constructed automatically, while the features in the Liu's and Zhang's datasets require non-trivial manual efforts. Furthermore, our results also indicate that having extra information about pathways provides better performance as shown in Table 6.7, where Bio2RDF v2 is built by adding KEGG dataset to Bio2RDF v1. To further explore the influence of possible feature set combinations on the results, we integrated the original Liu's dataset features with Bio2RDF v2, leading to 40,260 features in total. Table 6.8 shows the performance results obtained when combining feature sets from Liu's and Bio2RDF v2 datasets. This yields slightly better results in terms of the AP and AUC-PR metrics.

Table 6.6: Predictive power of the proposed models using drugs in the Liu's dataset and features from the Bio2RDF v1 dataset (DrugBank + SIDER).

Model	Evaluation Criterion					
	AP ↑	AUC-PR ↑	AUC-ROC ↑	R-Loss ↓	One-Error ↓	Cov-Error ↓
KG-SIM-PROP	0.5011±0.0106	0.4485±0.0115	0.8935±0.0096	0.1058±0.0122	0.1586±0.0177	1095.3082±55.47904
kNN	0.4977±0.0107	0.4210±0.0228	0.8848±0.0062	0.1211±0.0113	0.1658±0.0206	1127.7254±45.6342
Decision Trees	0.1964±0.0116	0.1710±0.0138	0.6301±0.0250	0.7220±0.0194	0.5673±0.0144	1377.2001±6.9189
Random Forests	0.4317±0.0107	0.3843±0.0143	0.8097±0.0102	0.3037±0.0088	0.2212±0.0139	1314.5006±17.6714
MLP	0.5099±0.0159	0.4546±0.0169	0.9010±0.0061	0.0791±0.0022	0.1430±0.0160	892.8340±20.4758
Linear Regression	0.2847±0.0083	0.2482±0.0137	0.6404±0.0248	0.6726±0.0141	0.3467±0.0238	1383.3808±3.2383

Note: The evaluation metrics are AP, AUC-PR curve, AUC-ROC, R-loss, one-error and Cov-error. ("↑" indicates that the higher the metric value, the better, and "↓" indicates that the lower the metric value, the better.) Coloured values represent the best performing method across a given metric.

Table 6.7: Predictive power of the proposed models using drugs in the Liu's dataset and features from the Bio2RDF v2 dataset (DrugBank + SIDER + KEGG).

Model	Evaluation Criterion					
	AP ↑	AUPR ↑	AUROC ↑	R-Loss ↓	One-Error ↓	Cov-Error ↓
KG-SIM-PROP	0.5118±0.0101	0.4604±0.0097	0.8954±0.0054	0.1051±0.0109	0.1466±0.0214	1091.9749±51.4537
kNN	0.5083±0.0124	0.4341±0.0277	0.8835±0.0086	0.1281±0.0031	0.1478±0.0027	1155.2053±36.5165
Decision Trees	0.2069±0.0176	0.1742±0.0266	0.6258±0.0242	0.7140±0.0233	0.5469±0.0385	1370.7402±7.5913
Random Forests	0.4438±0.0162	0.3993±0.0256	0.8153±0.0171	0.2883±0.0225	0.2103±0.0169	1295.7516±20.2287
MLP	0.5278±0.0106	0.4725±0.0284	0.9002±0.0074	0.0795±0.0028	0.1322±0.0298	909.7297±19.7920
Linear Regression	0.2919±0.0109	0.2587±0.0165	0.6441±0.0261	0.6665±0.0166	0.3557±0.0306	1383.3796±3.2407

Note: The evaluation metrics are AP, AUC-PR curve, AUC-ROC, R-loss, one-error and Cov-error. ("↑" indicates that the higher the metric value, the better, and "↓" indicates that the lower the metric value, the better.) Coloured values represent the best performing method across a given metric.

Table 6.8: Predictive power of the proposed models using a combination of features from both the Liu's dataset and the Bio2RDF v2 dataset.

Model	Evaluation Criterion					
	AP ↑	AUC-PR ↑	AUC-ROC ↑	R-Loss ↓	One-Error ↓	Cov-Error ↓
KG-SIM-PROP	0.5012±0.0079	0.4471±0.0097	0.8882±0.0089	0.1184±0.0139	0.1526±0.0177	1127.3234±51.2769
kNN	0.5020±0.0808	0.4482±0.0101	0.8883±0.0089	0.1184±0.0139	0.1502±0.0208	1127.1279±51.3701
Decision Trees	0.2080±0.0190	0.1728±0.0149	0.6306±0.0239	0.6944±0.0215	0.5444±0.0289	1372.1095±9.6089
Random Forests	0.4609±0.0174	0.4331±0.0127	0.8357±0.0117	0.2627±0.0134	0.1995±0.0241	1308.7285±24.9798
MLP	0.5281±0.0088	0.4870±0.0269	0.8946±0.0067	0.0835±0.0034	0.1418±0.0158	937.8773±36.9387
Linear Regression	0.3031±0.0108	0.2681±0.0169	0.6578±0.02424	0.6431±0.0147	0.3617±0.0273	1381.7218±4.0156

Note: The evaluation metrics are AP, AUC-PR curve, AUC-ROC, R-loss, one-error and Cov-error. ("↑" indicates that the higher the metric value, the better, and "↓" indicates that the lower the metric value, the better.) Coloured values represent the best performing method across a given metric.

6.6.3 Comparison on SIDER 4 dataset

To further evaluate the practical applicability of the multi-label learning models, we performed an experiment using the SIDER 4 dataset (W. Zhang, F. Liu, L. Luo, et al., 2015). The intuition behind this experiment is to test the predictive power of the models under a simple train and test set up. SIDER 4 dataset contains 771 drugs used for training, which are also present in Liu's dataset, and 309 newly added drugs used for testing. First, we run all methods on the original SIDER 4 dataset features and labels, and compare them against the results provided by W. Zhang, Yanlin Chen, Tu, et al. (2016). Table 6.9 shows the results

Table 6.9: Comparison of the predictive power of the models on the SIDER 4 dataset.

Model	Evaluation Criterion					
	AP ↑	AUC-PR ↑	AUC-ROC ↑	R-Loss ↓	One-Error ↓	Cov-Error ↓
Liu's method	0.1816	0.1766	0.8772	0.1150	0.9870	1587.5663
FS-MLKNN	0.3649	0.3109	0.8722	0.1038	0.1851	1535.9223
LNSM-SMI	0.3906	0.3465	0.8786	0.0969	0.2013	1488.2977
LNSM-CMI	0.3804	0.3332	0.8852	0.0952	0.1916	1452.7184
KG-SIM-PROP	0.3375	0.2855	0.8892	0.1398	0.2233	4808.3689
*k*NN	0.3430	0.2898	0.8905	0.1392	0.2168	4086.0777
Random Forests	0.3004	0.2599	0.8235	0.3318	0.2848	5362.6117
MLP	0.3546	0.2899	0.8943	0.0922	0.1309	4054.0356

The values for the first four models were taken from (W. Zhang, Yanlin Chen, Tu, et al., 2016). The evaluation metrics are AP, AUC-PR curve, AUC-ROC, R-loss, one-error and Cov-error. ("↑" indicates that the higher the metric value the better, and "↓" indicates that the lower the metric value the better.) Coloured values represent the best performing method across a given metric.

Table 6.10: Predictive power of the proposed models on drugs in the SIDER 4 dataset using Bio2RDF v2 dataset features.

Model	Evaluation Criterion					
	AP ↑	AUC-PR ↑	AUC-ROC ↑	R-Loss ↓	One-Error ↓	Cov-Error ↓
KG-SIM-PROP	0.3438	0.2876	0.8764	0.17460	0.2427	4969.0647
*k*NN	0.3416	0.2835	0.8728	0.1777	0.2395	5002.6084
Random Forests	0.2384	0.2061	0.7651	0.4567	0.4304	5440.0712
MLP	0.3529	0.2857	0.9043	0.0852	0.1909	3896.3625

Note: The evaluation metrics are AP, AUC-PR curve, AUC-ROC, R-loss, one-error and Cov-error. ("↑" indicates that the higher the metric value, the better, and "↓" indicates that the lower the metric value, the better.) Coloured values represent the best performing method across a given metric.

of the different methods over the SIDER 4 dataset. The state-of-the-art method LNSM-SMI gives the best average precision and AUC-PR, and the LNSM-CMI method the best coverage error. However, our multi-layer perceptron is the best performing model in AUC-ROC, ranking loss, and one-error. These results suggest better relative suitability of some multi-label learning methods for applications where a ranking function is required. Examples of such applications are use cases, where experts can only review a few prediction candidates and need the relevant ones to appear at the top of the list. Such use cases are indeed realistic, as there are often hundreds of predictions for every single drug. The results of multi-layer perceptron show some improvements when using features coming from the Bio2RDF v2 dataset (see Table 6.10).

6.6.4 Comparison on Aeolus dataset

We further evaluate the models considering both the SIDER 4 and Aeolus datasets (Banda, Evans, Vanguri, et al., 2016). Aeolus dataset provides us with relations between drugs and ADRs that were not previously known during the training or testing steps. The reason for the SIDER 4 and Aeolus experiments is the nature of the labels. The classic approach for validating ADR predictions follows the closed world assumption (missing predictions are false), but the actual problem follows the open world assumption (missing predictions

may be just unknown at the moment). Therefore there is always the possibility that those false positive predictions become true positives in a near future. Here, we hope to reflect this phenomenon by using the complementary Aeolus data that is frequently updated and contains information based on manually validated reports.

To test this point, we updated the matrix $\boldsymbol{Y}$ of ADRs of the test set using a version of Aeolus dataset generated after the release of the SIDER 4 dataset. We found 142 drugs in the intersection of SIDER 4 testing set and Aeolus. Whenever a new drug-ADR relationship is reported in the Aeolus dataset for any of the 309 drugs in the test set, this is reflected by modifying the SIDER 4 dataset. Aeolus introduces 615 new ADR relations in total with an average of 4.3 per drug. For example, Aeolus provides two new ADRs for triclosan (DB08604), an aromatic ether widely used as a preservative and antimicrobial agent in personal care products: odynophagia and paraesthesia oral. While these changes because of the Aeolus dataset are not crucial for drugs with many previously known ADRs (for instance, nilotinib (DB04868) has 333 ADRs in SIDER 4, and Aeolus only adds 3 new ADRs), they can have high impact on drugs with few known ADRs (such as triclosan or mepyramine both with only one ADR). In total, Aeolus provides at least one new ADR for 46% of drugs in the SIDER 4 test set. Interestingly, most of the new ADRs added by Aeolus dataset are related to the digestive system (e.g. intestinal obstruction, gastric ulcer, etc.), which we believe is because of the disproportionate FAERS reporting (Szarfman, Machado, and O'neill, 2002; Harpaz, Vilar, DuMouchel, et al., 2013) frequency for this type of events.

We ran the models once more and evaluated them against the new gold standard with the updates provided by Aeolus dataset. Table 6.11 shows the results of the updated dataset using Aeolus data for the four best performing multi-label models, where in comparison to Table 6.9 there are marginally lower results across all metrics. For instance, the AP of multi-layer perceptron drops by 0.92% and AUC-ROC by 1.85%. This observation is not consistent with our assumption that new knowledge about relations between drugs and ADRs can increase the true-positive rate by confirming some of the previous false positives as being true. We believe that this could be because of two reasons. (A) The added ADRs are under represented across drugs. We observed this in SIDER 4, where 37.5% (2093 of 5579) of ADRs are present at most once in either the training or test set. This makes those ADRs hard to predict. (B) There is a 'weak' relation between the drugs and the introduced ADRs. This weak relation comes from the original split in training and test set provided in SIDER 4 data set; we found out that 50.15% (2798 of 5579) ADRs are only present in the training set and not in the test set, compared with a 7% (392 of 5579) of ADRs that are only present in the test set. A proper assessment may require stratifying the experimental dataset and/or more representative extensions, which we leave to explore as future work.

Table 6.11: Predictive power of the proposed models on the SIDER 4 dataset with updated ADRs from the Aeolus dataset.

Model	Evaluation Criterion					
	AP ↑	AUC-PR ↑	AUC-ROC ↑	R-Loss ↓	One-Error ↓	Cov-Error ↓
KG-SIM-PROP	0.3272	0.2791	0.8796	0.1619	0.2233	5040.0615
*k*NN	0.3324	0.2834	0.8808	0.1613	0.2168	5038.6570
Random Forests	0.2883	0.2447	0.8059	0.3717	0.3366	5478.8479
MLP	0.3437	0.2836	0.8858	0.1050	0.1909	4339.7540

Note: The evaluation metrics are AP, AUC-PR curve, AUC-ROC, R-loss, one-error and Cov-error. ("↑" indicates that the higher the metric value, the better, and "↓" indicates that the lower the metric value, the better.) Coloured values represent the best performing method across a given metric.

6.7 Summary

In this chapter, we have shown how the structure provided by neighbourhood mixtures can be used to tackle the problem of ADR prediction in bioinformatics. We defined the problem in Section 6.1, were we frame the prediction of adverse drug reactions as a multi-label learning problem. Recently, multi-label learning has been used to address many of the classification problems in the area of bioinformatics. In Section 6.2, we reviewed previous approaches proposed to solve the ADR prediction problem. We proposed the use of knowledge graphs as an heterogeneous data source that turned to be a rich format to obtain feature vectors (Section 6.3). Neighbourhood mixtures allow us to combine heterogeneous data, and straightforwardly apply and evaluate different classes of machine learning models. The biomedical knowledge graphs that we considered as data sources were presented in Section 6.4, integrating different types of information on drugs and side effects. Our experiments (described in Section 6.5) showed that our multi-label learning models provide a simple and effective solution to predict potential side effects. Furthermore, the competitive performance of our multi-layer perceptron model (Section 6.6) provides unprecedented flexibility and scalability to provide good ranking metrics that can help experts during decision making processes.

CHAPTER 7

Cardinality Regularisation of Neural Link Predictors

Contents

Neural link predictors are models that learn distributed representations (embeddings) of entities and relations in a knowledge graph. They are remarkably powerful in the link prediction and knowledge graph completion tasks, mainly due to the learnt representations that capture important statistical dependencies in the data. Recent works have focused on either designing new scoring functions or incorporating extra information into the learning process to improve the representations. Yet these representations are mostly learnt from the observed links between entities, ignoring common sense or schema knowledge related to the relations, such as cardinality information—a fundamental aspect of the topology of data. In this chapter, we propose to learn better representations by incorporating a regularisation term inspired by *relation cardinality constraints*, which can be added to any existing neural link predictor without impacting their efficiency or scalability.

7.1 Problem statement

Cognitive development of children indicates that we learn the cardinality-related question *"How many?"* at 3.5 years of age (Wynn, 1990). This ability helps us to recognise physical and abstract things by counting; for example, we all know that a hand has five fingers, a car has four wheels or a meeting has more than two participants. This kind of background knowledge is not obvious for machines to acquire, even in contexts where it can be useful, such as Question Answering, Web Search, and Information Extraction (Tandon, Varde, and Melo, 2017).

One fundamental application area for cardinalities relates to the completion of knowledge graphs. For instance, consider Freebase (Bollacker, R. P. Cook, and Tufts, 2007), the core of the Google Knowledge Graph project, where X. Dong, Gabrilovich, Heitz, et al. (2014) reports that 71% of the people described in it have no known place of birth. By leveraging cardinality information about the *bornIn* relationship (i.e., each person must have a place of birth), we can quantitatively assess the degree of incompleteness in Freebase and focus the resources on predicting a single place of birth for each person. Yet *link prediction models* aimed at identifying missing facts in KGs do not consider such background knowledge, yielding potentially inconsistent and inaccurate predictions with high probability.

To tackle this problem, our RQ (4) (see Section 1.2) focuses on investigating the effects of incorporating cardinality information as constraints during the learning of models used to complete knowledge graphs. We aim to use cardinality constraints to impose boundaries on the number of predictions with high probability, thus, structuring the embedding space to respect common sense cardinality assumptions. Our experimental results on Freebase, WordNet and YAGO show that, given suitable prior knowledge, the proposed method consistently improves the predictive accuracy of downstream link prediction tasks.

In this work, we focus on a certain class of link prediction models, namely *Neural Link Predictors* (Nickel, K. Murphy, Tresp, et al., 2016). Such models learn low-dimensional distributed representations—also referred to as *embeddings*—of all entities and relations in a knowledge graph. Neural link predictors are currently the state of the art solution for tasks such as link prediction (Bordes, Usunier, García-Durán, et al., 2013; B. Yang, Yih, X. He, et al., 2015; Trouillon, Welbl, Riedel, et al., 2016), entity disambiguation and entity resolution (Bordes, Glorot, Weston, et al., 2014), taxonomy extraction (Nickel, Tresp, and Kriegel, 2012; Nickel and Kiela, 2017), and probabilistic question answering (Krompaß, Nickel, and Tresp, 2014). Recently, the focus has been on either designing new scoring functions, or incorporating additional background knowledge during the learning process. We refer readers to Nickel, K. Murphy, Tresp, et al. (2016) and Q. Wang, Mao, B. Wang, et al. (2017) for a review of these models.

Neural link predictors proposed in the literature miss to leverage prior knowledge in the form of relation cardinality information. For instance, prior knowledge encoding cardinality statements such as *"a person should have at most two parents"* or *"a patient should be taking between 1 and 5 drugs at a time"* are not taken into account by neural link prediction

Table 7.1: Top-5 predictions for parents of parent of *edgar_allan_poe* given by DistMult (B. Yang, Yih, X. He, et al., 2015).

Triples	Probability
(*edgar_allan_poe*, *hasParent*, *edgar_allan_poe*)	0.989
(*edgar_allan_poe*, *hasParent*, *eliza_poe*)	0.979
(*edgar_allan_poe*, *hasParent*, *virginia_eliza_clemm_poe*)	0.974
(*edgar_allan_poe*, *hasParent*, *julia_ward_howe*)	0.890
(*edgar_allan_poe*, *hasParent*, *benjamin_franklin*)	0.889

models. Such knowledge can be provided by domain experts, or automatically extracted from data (L. Galárraga, Razniewski, Amarilli, et al., 2017; Muñoz and Nickles, 2017). And it is expected that such cardinality constraints will be satisfied by both the knowledge graph and algorithms analysing the graph (e.g., link predictors). We believe that these constraints can impose common sense knowledge upon the structure of the embeddings space, thus, helping us to learn better representations that boost the performance of downstream tasks.

Formally, we define the problem as follows:

Cardinality regularisation problem

Input: a neural link predictor model $\phi_r : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$ with parameters Θ, a set Φ of cardinality bounds, and a knowledge graph $\mathcal{G}$

Output: a cardinality-aware neural link predictor ϕ'_r, where the link predictions over $\mathcal{G}$ using Θ satisfy all cardinality constraints in Φ

Cardinality constraints are one of the most important constraints in conceptual modelling (Olivé, 2007, Chapter 4) as they explicit the topology of data. However, no existing link prediction model considers them during learning. For instance, in the prediction of parents (relation *hasParent*) for the entity *Edgar Allan Poe*, we expect to predict at most two parents, namely, *Eliza Poe* and *David Poe Jr.* To illustrate this, let us analyse the predictions of a state-of-the-art neural link prediction model, DistMult (B. Yang, Yih, X. He, et al., 2015), using the Freebase FB13 dataset (Bordes, Weston, Collobert, et al., 2011), containing entities of the Freebase type *deceased people* and their relations. Table 7.1 shows the top-5 predicted parents for *edgar_allan_poe*. As we can see, most of the predictions are given a high probability with 22 entities scored higher than 0.8 despite the fact that most predictions are incorrect. Nevertheless, evaluation results are good due to the evaluation protocol of link prediction models based on learning to rank, where correct predictions (e.g., *eliza_poe*) should be ranked higher than incorrect of false ones (e.g., *benjamin_franklin*).

7.2 Related work

Early works in neural link prediction such as TransE (Bordes, Usunier, García-Durán, et al., 2013), RESCAL (Nickel, Tresp, and Kriegel, 2011), and DistMult (B. Yang, Yih, X. He, et al., 2015)) learn distributed representations using simple operations (addition, multiplication) to score the triples in a knowledge graph. While more recent research has focused on either (i) generating more elaborated scoring functions that better capture the nature of relations, or (ii) improving existing models with background knowledge. The former considers models like HolE (Nickel, Rosasco, and Poggio, 2016), where the scoring function is inspired by associative memory; ComplEx (Trouillon, Welbl, Riedel, et al., 2016) that uses complex-valued embeddings to model asymmetric relations; and ConvE (Dettmers, Minervini, Stenetorp, et al., 2018) that builds a multi-layer convolutional network to predict links. The latter focus is characterised by the incorporation of additional information such as entity types, relation paths, and logical rules. We refer the readers to (Nickel, K. Murphy, Tresp, et al., 2016; Q. Wang, Mao, B. Wang, et al., 2017) for a deeper review of neural link predictors.

Our work aligns with the second category that focuses on adding background knowledge. Almost every paper incorporating background knowledge agree that such prior knowledge improves link prediction models; however, none of them has considered integrity constraints such as cardinality so far. We build upon Muñoz and Nickles (2017), where we proposed mining algorithms for cardinality constraints from knowledge graphs. The use of these constraints to improve the accuracy of link prediction models is only suggested as future work. In the same vein, L. Galárraga, Razniewski, Amarilli, et al. (2017) use fine-grain cardinality information to learn rules that can be used to prune 'unnecessary' predictions; however, this is done only after the predictions are generated. Jiawei Zhang, Jianhui Chen, Zhu, et al. (2017) propose a method where a single cardinality bound (i.e., one-to-one, one-to-many or many-to-many) can be imposed in link prediction over uni-relational graphs (e.g., organisational charts). However, their work cannot be directly applied to knowledge graphs due to their more complex multi-relational nature.

7.3 Regularisation based on cardinality

In this chapter, we propose an efficient approach for incorporating the notion of cardinality to the training of any neural link prediction model, without affecting their efficiency and scalability. The proposed approach adjusts the embedding of entities and relations during the learning phase using a regularisation term that penalises predictions with high probability that are above or below the imposed cardinality. By doing so, the notion of relation cardinality will be captured during training to learn better link prediction models, i.e., that comply with available background knowledge (Q. Wang, B. Wang, and L. Guo, 2015), producing more accurate predictions.

Example 7.1

Given a cardinality bound $\varphi_{hasParent} = (0, 2)$, encoding the constraint *"a person should have at most two parents"*, we would like to ensure that the embeddings learnt by a neural link predictor yield predictions for the *hasParent* relation within the boundaries. In other words, we want to have the sum of probabilities over all possible parent entities of *Edgar Allan Poe* precisely between zero and two.[a] We express this constraint over the triple $\tau = (edgar\_allan\_poe, hasParent, t)$ as:

$$0 \leq \sum_{t \in \mathcal{E}} p(y_{hrt} = 1 \mid \Theta) \leq 2, \tag{7.1}$$

where t is an entity and the conditional probabilities $\forall t \in \mathcal{E}$ are given by the neural link prediction model.

[a] Note that by considering a zero lower bound, we account for the possible incompleteness of the KG.

The inequality term in Example 7.1 expresses a supervision signal, not based on labelled data, but that can be incorporated in the training of neural link prediction models. Again, such cardinality boundaries can be provided by experts, gathered from literature (Mirza, Razniewski, Darari, et al., 2017), or extracted from the knowledge base (L. Galárraga, Razniewski, Amarilli, et al., 2017; Muñoz and Nickles, 2017). Specifically, we propose to leverage cardinality bounds (e.g., the one in Equation (7.1)) to define a regularisation term that encourages models to respect the available cardinality constraints.

Let $\Phi = \{\varphi_r = (\varphi_r^{\downarrow}, \varphi_r^{\uparrow})\}_{r \in \mathcal{R}}$ be the set of cardinality constraints for each relation in a given knowledge graph $\mathcal{G}$, where $\varphi_r^{\downarrow}$ and $\varphi_r^{\uparrow}$ are the lower and upper bound for relation r, respectively.

Given $r \in \mathcal{R}$ and $h \in \mathcal{E}$, let $\mathcal{A}_{hr}[\mathcal{E}] \triangleq \{(h, r, t) : \forall t \in \mathcal{E}\}$ be the set of all possible triples with relation r and subject h, where the object was selected from $\mathcal{E}$. For instance, r denotes the relation *hasParent*, and h denotes the entity *edgar_allan_poe*.

Thus, we can define the following hard constraint on the conditional probability of all triples in the set $\mathcal{A}_{hr}[\mathcal{E}]$:

$$\varphi_r^{\downarrow} \leq \left(\mathcal{X}_{hr}[\mathcal{E}] \triangleq \sum_{x_{hrt} \in \mathcal{A}_{hr}[\mathcal{E}]} p_\Theta(y_{hrt} = 1 \mid \Theta) \right) \leq \varphi_r^{\uparrow}. \tag{7.2}$$

Because of the hardness of Equation (7.2) it becomes impractical to incorporate directly in neural link predictors. We propose a relaxation or *soft constraint* by defining a continuous and differentiable loss function that penalises violations to such constraint. Specifically, we define a function G_{hr} that is strictly positive if the cardinality constraint for a given entity h and relation r is violated, and zero otherwise. Given a cardinality constraint φ_r, the function

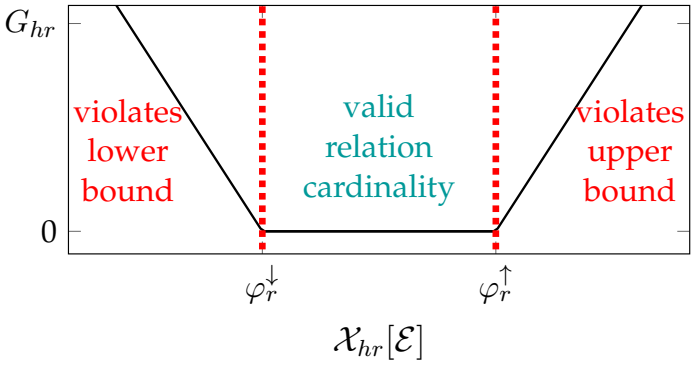


Figure 7.1: Value of the regularisation term G_{hr} based on the lower and upper bounds of a cardinality constraint $\varphi_r = (\varphi_r^{\downarrow}, \varphi_r^{\uparrow})$.

$G_{hr}[\mathcal{E}; \Phi]$ (or G_{hr} for simplicity) is defined as follows:

$$G_{hr}[\mathcal{E}; \Phi] = \max(0, \varphi_r^{\downarrow} - \mathcal{X}_{hr}[\mathcal{E}]) + \max(0, \mathcal{X}_{hr}[\mathcal{E}] - \varphi_r^{\uparrow}). \tag{7.3}$$

Figure 7.1 shows the values of G_{hr} (see Equation (7.3)) based on $\mathcal{X}_{hr}[\mathcal{E}]$ and a cardinality bound $\varphi_r \in \Phi$. Notice that for the general case where the upper bound corresponds to ∞ and lower bound to 0, the loss G_{hr} vanishes.

With this intuition in mind, we define a cardinality-regularised objective function, denoted by $\mathcal{L}^C(\Theta)$, which is applicable to any neural link prediction model:

$$\mathcal{L}^C(\Theta) = \mathcal{L}(\Theta) + \lambda \sum_{\Phi} G_{hr}, \tag{7.4}$$

where $\lambda \in \mathbb{R}_+$ weights the relative contribution of the expectation term, and $\mathcal{L}(\Theta)$ can be either the pairwise ranking loss or the logistic loss. This regularised loss Equation (7.4) can be minimised using stochastic gradient descent (SGD) (Robbins and Monro, 1951) in mini-batch mode, outlined previously in Algorithm 1.

Although our approach considers both upper and lower bounds, we noticed that the latter cannot be meaningfully imposed in all cases. For instance, given a constraint $\varphi_{spouse} = (1, 1)$, the desired effects of G_{hr} can yield inconsistent results if the knowledge graph is incomplete, and does not contain the *spouse* link of every person. In such cases, a zero lower bound can be used to address the knowledge graph incompleteness.

Our approach is intuitive and easy to implement for any neural link prediction model. However, its usefulness is limited by the cost of computing the sum in Equation (7.2): the set $\mathcal{A}_{hr}[\mathcal{E}]$ can easily be large in some KGs and become too expensive to compute the sum of marginal probabilities. In the following sections, we propose two techniques to overcome this problem and approximate the sum of probabilities via estimation and sampling.

7.3.1 Lower bound estimation

Instead of dealing with the whole set of entities, we can sample a much smaller subset of all entities $\mathcal{S} \subseteq \mathcal{E}$ and obtain the following lower bound:

$$\mathcal{X}_{hr}[\mathcal{S}] \leq \mathcal{X}_{hr}[\mathcal{E}]. \tag{7.5}$$

The tightness of the bound in Equation (7.5) is determined by the selection of the entities in $\mathcal{S}$. In this work, we consider *uniform sampling*. More specifically, a random set of indices $\mathcal{S} \equiv \{i_1, \ldots, i_S\}$ is taken uniformly, where $i_s \in \{1, \ldots, |\mathcal{E}|\}$, and form the following lower bound:

$$\sum_{x_{hrt} \in \mathcal{A}_{hr}[\mathcal{S}]} p(y_{hrt} = 1 \mid \Theta) \leq \mathcal{X}_{hr}[\mathcal{E}],$$

where the sum is over all elements in $\mathcal{S}$ with no repetitions.

7.3.2 Sum estimation

Instead of approximating a lower bound to $\mathcal{X}_{hr}[\mathcal{E}]$, we can also approximate its value directly by *sampling*.Let us consider the sum over a large collection of elements $Z \triangleq \sum_c z_c$. We consider two standard methods for approximating sums via Monte Carlo estimates, namely Importance Sampling (IS) and Bernoulli Sampling (Botev, B. Zheng, and Barber, 2017).

Importance Sampling. In Importance Sampling (IS), based on the identity $Z = \sum_c \frac{q(c) z_c}{q(c)}$, a set of indices $\mathcal{S} \equiv \{i_1, \ldots, i_S\}$ is selected from a distribution q, where $i_s \in \{1, \ldots, |\mathcal{E}|\}$, and yield the following approximation:

$$Z \approx \frac{1}{S} \sum_{s \in \mathcal{S}} \frac{z_s}{q(s)},$$

where $q(s)$ defines the probability of sampling s from $\mathcal{S}$.

Bernoulli Sampling. An alternative to IS is Bernoulli Sampling (BS), considering the following identity:

$$Z = \sum_c z_c = \mathbb{E}_{\mathbf{s} \sim \mathbf{b}} \left(\sum_c \frac{s_c}{b_c} z_c \right),$$

where each independent Bernoulli variable $s_c \in \{0, 1\}$ denotes whether z_c will be sampled or not, and $p(s_c = 1) = b_c$ is the probability of sampling z_c. This leads to the following approximation:

$$Z \approx \sum_{c : s_c = 1} \frac{z_c}{b_c},$$

where the sum is computed over the components with non-zero elements in the vector $\mathbf{s}$. Note that, when calculating an approximation to Z, IS relies on sampling with replacement, while BS relies on sampling without replacement.

By using our regularisation term with sampling, we add a time complexity $O(m \times n)$, where m is the total number of (sampled) triples in the regularisation and n the number of triples per batch. Since m is usually much smaller than the number of triples in a batch, we ensure that the time complexity of neural link predictors is not sensibly affected during training, and not affected at all at test time. The proposed method does not increase the space complexity of the models, since the proposed regulariser does not change the number of model parameters Θ.

7.4 Experimental settings

In this section, we investigate the benefits of cardinality regularisation for the state-of-the-art neural link prediction models. We compare the performance of original and regularised losses in the link prediction task across different benchmark datasets, which are partitioned into train, validation and test sets of triples (see Table 7.2 for the characteristics).

7.4.1 Evaluation protocol

The link prediction task consists of predicting a missing entity h or t when given a pair (r, t) or (h, r), respectively. During testing, for each test triple (h, r, t), we replace the subject or object entity with all entities in the knowledge graph as corruptions (Bordes, Usunier, García-Durán, et al., 2013). The evaluation then ranks the entities in descending order w.r.t. the scores calculated by a scoring function and gets the rank of the correct entity h or t. We report results based on the ranks assigned to correct entities measured using mean reciprocal rank (MRR) and Hits@n with $n \in \{1, 3, 5, 10\}$.[1] During the ranking process some positive test triples could be ranked after another true triples, which should not be considered a mistake. Therefore, the above metrics have two settings: *raw* and *filtered* (Bordes, Usunier, García-Durán, et al., 2013). In the filtered setting, metrics are computed after removing all true triples appearing in train, validation, or test sets from the ranking, whereas in the raw setting they are not removed.

We highlight the fact that in our link prediction approach predicting a fact that is true in the real world but not in the knowledge graph will be counted as an error. Because our knowledge of the real world is often imperfect, this is true for any knowledge discovery approach that assumes the Closed World Assumption (Motro, 1996).

[1] For MRR and Hits@n, the higher the better.

Table 7.2: Datasets characteristics.

Dataset	N_r	N_e	№ train	№ valid	№ test
FB13	13	81,065	350,517	5,000	5,000
WN18	18	40,943	14,1442	5,000	5,000
WN18RR	11	40,943	86,835	3,034	3,134
YAGO3-10	37	123,182	1,079,040	5,000	5,000

Note: № represents the number of triples in the train, validation and test sets.

7.4.2 Datasets

Three widely used sources for the link prediction task are WordNet (Miller, 1995), Freebase (Bollacker, R. P. Cook, and Tufts, 2007) and YAGO (Mahdisoltani, Biega, and Suchanek, 2015). In this work, we use four benchmark datasets generated from them: FB13, WN18, WN18RR and YAGO3-10. The FB13 dataset (Bordes, Weston, Collobert, et al., 2011) is a subset of Freebase containing 13 relation types and entities of type *deceased_people*, where entities appear in at least 4 relations and relation types at least 5,000 times.[2] We also use two datasets derived from WordNet, namely, WN18 and WN18RR. These datasets contain hyponym, hypernym, and other lexical relations of English concepts and words. It is known that WN18 contains ca. 72% of redundant and inverse relations (e.g., *hyponym* $\equiv$ *hypernym* $^{-1}$), which have been removed to build WN18RR dataset (Dettmers, Minervini, Stenetorp, et al., 2018). YAGO3-10 consists of entities in YAGO3 (mostly of type people) linked with at least 10 relations, such as citizenship, gender and profession. FB13, WN18RR and YAGO3-10 datasets were shown to have no redundant or trivial triples (Dettmers, Minervini, Stenetorp, et al., 2018). In Table 7.2 we describe the characteristics of each dataset.

We mine the relation cardinality constraints from the training set of each dataset, following the algorithm proposed by Muñoz and Nickles (2017) using the normalisation option but without filtering outliers. Table 7.3 gives examples of the cardinality constraints mined from each dataset. Since the source knowledge graphs of the bounds are incomplete, we manually checked the constraints and updated some of the bounds (e.g., */people/person/gender* lower bound from 0 to 1).

7.5 Results and discussion

For our experiments, we re-implemented three models using the TensorFlow framework (Abadi, Barham, Jianmin Chen, et al., 2016), namely, ER-MLP (X. Dong, Gabrilovich, Heitz, et al., 2014), DistMult (B. Yang, Yih, X. He, et al., 2015) and ComplEx (Trouillon, Welbl, Riedel, et al., 2016) (which has been proven to be equivalent to HolE (Nickel, Rosasco, and

[2] We use the corrected version by Socher, D. Chen, Manning, et al. (2013) that contains only positive samples.

Table 7.3: Cardinality constraints extracted from FB13, WN18 (WN18RR) and YAGO3-10. We also show the updated bounds after manual revision.

/people/person/place_of_birth	$(0, 2) \rightarrow (1, 1)$
/people/person/parents	$(0, 2)$
/people/person/gender	$(0, 1) \rightarrow (1, 1)$
_hyponym	$(0, 380)$
_has_part	$(0, 73)$
_hypernym	$(0, 4)$
livesIn	$(0, 12) \rightarrow (1, 12)$
hasGender	$(0, 1) \rightarrow (1, 1)$
hasChild	$(0, 19)$

Poggio, 2016)). We compare the performance over the four benchmark datasets of each model as originally stated by their authors and with the cardinality regularisation term (see Equation (7.4)). As recommended by Trouillon, Welbl, Riedel, et al. (2016), we minimise the logistic loss to train each model using SGD and AdaGrad (Duchi, Hazan, and Singer, 2011) to adaptively select the learning rate, which has been initialised as $\eta_0 = 0.1$. For each model and dataset, we selected hyperparameters maximising filtered Hits@10 on the validation set using an exhaustive grid search.

The evaluation of our approach is three-fold: (i) we measure the effects of the regulariser in the link prediction task; (ii) we measure the effects of the different sampling techniques; and (iii) we measure the violations to the cardinality constraints before and after regularisation. To reduce the search space, during the grid search in (i) we fix the sampling technique to uniform. In (ii), we use the best model identified in (i) to study the effect of different sampling techniques, whilst in (iii) we use the overall best model per dataset.

7.5.1 Link prediction evaluation

We train each model for 1,000 epochs with a mini-batches approach over the training set of each dataset, generating two negative examples per positive triple in each batch. We set $\lambda = 0$ (see Equation (7.4)) to obtain the performance results of original models (without regularisation), and use uniform sampling with sizes $\mu \in \{10, 100\}$ and $\omega \in \{10, 100, 1000\}$ for subjects and objects, respectively.[3] Tables 7.4 and 7.5 show the link prediction results, confirming that our cardinality-based regularisation term helps to improve (or at least maintain) the performance of the original ER-MLP, DistMult and ComplEx models across all datasets. The only exception we observed is ComplEx over YAGO3-10, where the model without the regularisation term reaches better Hits@10 and MRR. We believe that a reason for this is that constraining a lower bound on the sum of probabilities may not be the best technique to use when the number of entities is very large. In our experiments, we also

[3]We identified via independent experiments that larger values for μ do not yield performance improvements.

Table 7.4: Link prediction results (Hits@n and Mean Reciprocal Rank, filtered setting) on WN18 and WN18RR.

	WN18					WN18RR				
	Hits@n				MRR	Hits@n				MRR
Method	1	3	5	10		1	3	5	10	
ER-MLP	21.64	37.30	44.94	56.52	33.02	1.84	3.29	4.10	5.31	3.10
ER-MLP^C	**32.01**	**51.54**	**60.54**	**70.85**	**45.01**	**2.22**	**4.29**	**5.42**	**7.31**	**3.98**
DistMult	64.46	87.47	90.66	93.49	76.62	38.93	43.49	45.93	49.63	42.46
DistMult^C	**65.01**	**87.53**	**90.71**	**93.44**	**76.93**	**39.10**	**44.13**	**46.30**	**49.81**	**42.84**
ComplEx	88.33	93.05	94.14	95.07	90.96	40.87	**46.25**	**48.55**	**51.15**	44.52
ComplEx^C	**88.66**	**93.27**	**94.21**	**95.21**	**91.20**	**41.10**	46.06	48.13	51.09	**44.57**

Note: The evaluation metrics are MRR and Hits@n (hits at 1, 3, 5, and 10). (For all metrics, the higher the value of the metric, the better.) In bold the best results comparing both original and cardinality loss, and highlighted is the best value per evaluation metric across all models.

compare two alternative approaches, namely estimating the sum of probabilities via IS and BS.

ER-MLP and DistMult models benefit the most across all datasets with improvements of up to 36% in MRR. ComplEx shows to be the overall best performing model outperforming ER-MLP (up to 20x in WN18RR) and DistMult in every dataset and evaluation metric. Still, ComplEx benefits from the regularisation term in most of the datasets. Although, we did not perform a thorough search of the hyperparameters space to reach state-of-the-art performance, the results prove the advantages of our approach.

Table 7.5: Link prediction results (Hits@n and Mean Reciprocal Rank, filtered setting) on FB13 and YAGO3-10.

	FB13					YAGO3-10				
	Hits@n				MRR	Hits@n				MRR
Method	1	3	5	10		1	3	5	10	
ER-MLP	4.40	7.55	9.14	11.82	6.94	2.22	6.09	9.59	16.01	6.83
ER-MLP^C	**5.13**	**8.36**	**10.29**	**12.75**	**7.78**	**2.33**	**6.16**	**9.65**	**16.54**	**6.95**
DistMult	18.07	29.29	32.94	37.01	24.92	6.75	14.33	18.86	26.51	13.33
DistMult^C	**18.10**	**29.45**	**33.07**	**37.02**	**25.00**	**7.03**	**14.53**	**19.12**	**26.66**	**13.59**
ComplEx	**25.08**	31.64	34.00	36.90	**29.41**	7.12	**15.61**	**20.76**	**29.11**	14.33
ComplEx^C	24.89	**31.78**	**34.10**	**37.16**	29.36	**7.56**	15.10	20.30	29.01	**14.47**

Note: The evaluation metrics are MRR and Hits@n (hits at 1, 3, 5, and 10). (For all metrics, the higher the value of the metric, the better.) In bold the best results comparing both original and cardinality loss, and highlighted is the best value per evaluation metric across all models.

Table 7.6: Link prediction results (Hits@n and Mean Reciprocal Rank, filtered setting) for the best ComplEx model using different sampling techniques.

Dataset	Sampling	Hits@n 1	3	5	10	MRR
FB13	Uniform	25.84	31.85	34.19	37.26	29.89
	Importance	25.17	31.36	34.36	36.18	29.18
	Bernoulli	25.92	31.86	34.11	37.18	29.97
WN18	Uniform	88.98	93.66	94.84	95.98	92.12
	Importance	88.97	93.64	94.73	96.08	91.10
	Bernoulli	89.05	93.57	94.67	95.94	91.09
WN18RR	Uniform	41.27	46.57	48.58	51.51	44.87
	Importance	41.09	46.68	48.81	51.50	44.78
	Bernoulli	41.54	46.79	48.68	51.42	45.04
YAGO3-10	Uniform	8.32	15.52	20.92	29.29	15.30
	Importance	8.23	15.71	20.70	29.49	15.28
	Bernoulli	8.48	15.74	20.82	29.50	15.42

Note: The evaluation metrics are MRR and Hits@n (hits at 1, 3, 5, and 10). (For all metrics, the higher the value of the metric, the better.) Coloured values represent the best performing method across a given metric.

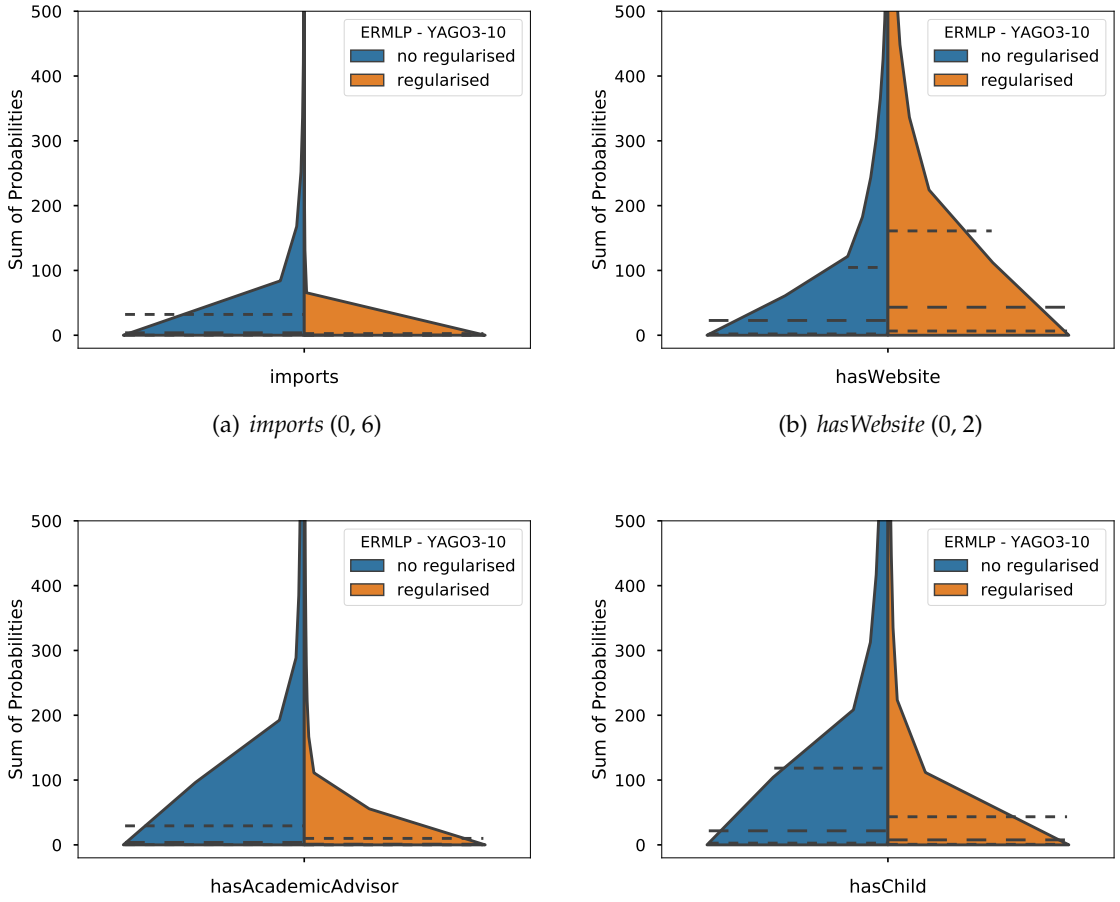
7.5.2 Sampling techniques evaluation

To approximate the sum of probabilities we test both Importance Sampling and Bernoulli Sampling, and consider hyperparameters $\mu \in \{10, 50, 100\}$ and $\omega \in \{10, 50, 100, 500, 1000\}$. Starting from the best ComplEx models learnt above, we tune the sampling technique for each of the datasets.

Results are shown in Table 7.6. In general, all sampling techniques work well and there is no *one-size-fits-all* solution: it depends on the dataset. (Information about properties of the data that benefit one of the samplings can be used, and custom sampling is also supported.) YAGO3-10 shows the biggest improvement of 6% in MRR using BS compared with the results in Table 7.5. This improvement might be correlated to the advantage of BS to handle the large number of entities in YAGO3-10. For FB13, WN18, and WN18RR we see smaller improvements in MRR and Hits@10 compared to the results in Table 7.4. Differences in results for uniform sampling compared to the results in Table 7.4 are also attributed to the expanded hyperparameters space used with more sampling sizes than previously.

7.5.3 Cardinality violations evaluation

We have shown that our regulariser is beneficial for the link prediction task, but, more importantly, the predictions that violate the cardinality constraints are significantly reduced. Figure 7.2 shows the changes on the distribution of $\mathcal{X}_{hr}[\mathcal{E}]$ in four relation cases for ER-



(a) *imports* (0, 6)

(b) *hasWebsite* (0, 2)

(c) *hasAcademicAdvisor* (0, 4)

(d) *hasChild* (0, 19)

Figure 7.2: Changes in the distribution of $\mathcal{X}_{hr}[\mathcal{E}]$ without (left, in blue) and with (right, in orange) regularisation using ER-MLP in YAGO3-10. Horizontal lines correspond to quartiles.

MLP in YAGO3-10—one of the most benefited settings. Figures 7.2(a), 7.2(c) and 7.2(d) illustrate positive impacts of the regularisation. We observed that the regulariser decreases the median and long-tail distribution above the third quartile for (almost) every relation, making predictions more accurate. For example, in relation *imports* ($\varphi = (0, 6)$) the mean of $\mathcal{X}_{hr}[\mathcal{E}]$ is reduced by 78%, meaning less violations. Conversely, the biggest negative impact was in relation *hasWebsite* ($\varphi = (0, 2)$, Figure 7.2(b)), where violations were increased by 65%. Both constraint are equally restrictive over the number of objects but they differ on their range. For the former, the objects are entities with links to other entities, while in the latter objects are literals (URLs) with no further links. The prediction of literals is a known problem for neural link predictors as there are not many links to other entities (García-Durán and Niepert, 2018).

Following the DistMult example using the constraint $\varphi_{hasParent} = (0, 2)$, Table 7.7 shows the predictions for parents of *Edgar Allan Poe*. There are less predictions with high proba-

Table 7.7: Predictions with probability > 0.8 for (*edgar_allan_poe*, *hasParent*, *?*) by DistMult when imposing the cardinality regulariser.

Triple	Probability
(*edgar_allan_poe*, *hasParent*, *eliza_poe*)	0.861
(*edgar_allan_poe*, *hasParent*, *maria_poe*)	0.854
(*edgar_allan_poe*, *hasParent*, *david_poe_jr*)	0.815

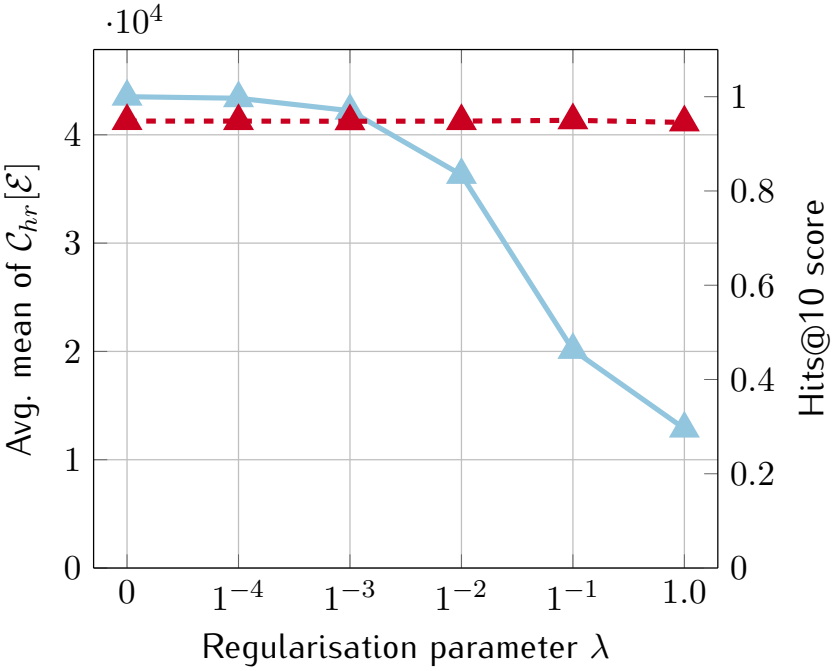


Figure 7.3: Influence of the regularisation weight over the average mean of $\mathcal{X}_{hr}[\mathcal{E}]$ (solid lines) and Hits@10 (dashed lines) in WN18 with ComplEx.

bility and the correct, but previously missing, entity *David Poe Jr.* is now scored with a high probability proving the effectiveness of our regularisation.

We did not note any major difference in results between tight and loose cardinality bounds, or between constraints for relations with few and many instances. Finally, Figure 7.3 shows the effects of using different regularisation weights $\lambda \in \{0, 0.0001, 0.001, 0.01, 0.1, 1.0\}$ over the values of average mean of $\mathcal{X}_{hr}[\mathcal{E}]$ and Hits@10 across relations in WN18RR. As λ grows, Hits@10 suffers small changes and the average mean of $\mathcal{X}_{hr}[\mathcal{E}]$ decreases. This shows that the regularisation term does not affect negatively Hits@10 (a common evaluation metric) and helps to decrease the number of violations to the cardinality constraints.

7.6 Summary

In this chapter, we presented a cardinality-based regularisation term for neural link prediction models. This regularisation term aims to solve a new problem (Section 7.1) created by the distributed representation of entities and relations in a knowledge graph: they do not follow common sense rules. The proposed regulariser (Section 7.3) incorporates background knowledge in the form of relation cardinality constraints that hitherto have been ignored by neural link predictors (Section 7.2). A limitation is that the regulariser is expensive to compute, thus, in Section 7.3.2, we proposed approximated values for it. We observed that incorporating this regularisation term in the loss function significantly reduces the number of violations produced by models at prediction time, enforcing the number of predicted triples with high probability for each relation to satisfy cardinality bounds. Experimental results (Section 7.5) show that the regulariser consistently improves the quality of knowledge graph embeddings, without affecting their efficiency or scalability.

Part IV

Conclusion

CHAPTER 8

Summary and Outlook

Contents

This thesis brings several contributions to the knowledge graph mining area. In this dissertation, we have reviewed the nascent area of knowledge graph mining (Chapter 3), we have proposed an algorithm for uncovering the structure of a knowledge graph via cardinality bounds (Chapter 4), an approximate validation algorithm for knowledge graphs (Chapter 5), a method that uses graph-based patterns to complete biomedical knowledge graphs (Chapter 6), and an approach that uses a regularisation term to enhance the learnt representations of entities and relations so they satisfy background knowledge (Chapter 7).

This chapter summarises the contributions of this thesis in Section 8.1 for each part, and in Section 8.3 we discuss the limitations and opportunities for future work in the area of knowledge graph mining.

8.1 Summary of contributions

In this section, we enumerate the contributions from the main three parts of this thesis as defined in Chapter 1: (I) *Background*, (II) *Latent shapes in knowledge graph*, and (III) *Knowledge graph mining applications*. Each part is composed by two chapters.

8.1.1 Part I: Background

The first part of this thesis established the background required to understand the contributions made to knowledge graph mining in this thesis. Knowledge graphs themselves have attracted the attention of researchers across academia and industry alike, mainly because of their versatility to model large number of entities and connections (links) between them using an easy to understand and coherent graph format. Based on the growing number of works contributed by researchers and practitioners in the last couple of years, we argue that knowledge graphs appear to be a more appealing solution for knowledge modelling than previous solutions (e.g., XML). We believe that this is mainly due to two reasons:

(i) knowledge graphs have a simplified semantics that does not require people to understand complex concepts as those underlying the Semantic Web or the logics behind knowledge bases, and

(ii) there are many (business) applications where knowledge graphs have shown to be huge factors of success (revenue), e.g., Google[1], Airbnb[2], LinkedIn[3], Amazon (X. L. Dong, 2019), among others.

Currently, for many applications, knowledge graphs (partially) bring to realisation the Semantic Web's vision and some of its main goals (Bollacker, R. P. Cook, and Tufts, 2007). This is driving the adoption of graph technologies for supporting tasks like categorisation and contextualisation in different applications. For instance, Airbnb comments[4] that:

> ... [we] will continuously invest to use our knowledge graph to enrich our understanding of the world of travel (categorization) and deliver more travel content (contextualization) to each traveller at every step of their trip planning and decision process.

Moreover, many new applications are powered by knowledge graphs for operation, for example, personal assistants (Haase, Nikolov, Trame, et al., 2017; Ram, Prasad, Khatri, et al., 2018) and chatbots (Adewale, Beatson, Buniatyan, et al., 2017; Athreya, Ngomo, and Usbeck, 2018).

By simplifying complex requirements from the Semantic Web (e.g., formal inference), practitioners and developers have been able to work seamlessly with knowledge graph and semantic technologies. An example is given by R. V. Guha, Brickley, and Macbeth (2016), who highlighted the increasing adoption and use of schema.org[5] in Web pages providing annotations about entities (e.g., products, places, events). It is easy for developers to get started with schema.org and they can quickly obtain benefits (e.g., search engine optimisation (SEO)) from sharing and representing their specific domains using annotations that can be later used to build large-scale (Web-scale) knowledge graphs.

[1] https://www.google.com/intl/bn/insidesearch/features/search/knowledge.html
[2] https://medium.com/airbnb-engineering/contextualizing-airbnb-by-building-knowledge-graph-b7077e268d5a
[3] https://engineering.linkedin.com/blog/2016/10/building-the-linkedin-knowledge-graph
[4] See Footnote 2
[5] https://schema.org/

The focus of this thesis is the analysis of structure or shapes present in knowledge graphs and its relation with two quality dimensions: consistency and completeness. Both of these dimensions have huge implications in tasks related to data mining and knowledge discovery in knowledge graphs. The first contribution of this thesis is a review and discussion around Semantic Web knowledge bases and knowledge graphs. In Chapter 2, we have reviewed the definitions of these concepts, their differences and similarities, and their interpretations. In terms of interpretation, despite the benefits that OWA and nUNA bring to knowledge bases—providing them with enormous versatility for open environments—there are many use cases like data mining, where such assumptions are not desirable at all. We believe that the UNA 2.0 proposed in SHACL and the partial closed-world assumption (PCWA) are key players in knowledge graph mining. Recently, these assumptions have been validated over noisy knowledge graphs in the link prediction task yielding state-of-the-art results (Nickel, K. Murphy, Tresp, et al., 2016; Q. Wang, Mao, B. Wang, et al., 2017).

As for the consistency of knowledge graphs, as second contribution, we have reviewed the schema languages recently proposed by the community and W3C. Knowledge graphs, and Semantic Web in general, follow a schema-last approach, where facts can be added as knowledge without checking against a schema. We provide an up-to-date overview of schema languages for knowledge graphs and paid special attention to SHACL and ShEx, which are the most popular ones. Given the schema-last approach and the lack of well-defined schemas for many knowledge graphs, we have also reviewed several approaches to inferring schema information from instance data. We believe that schema inference is a problem that will receive much more attention in the near future, given the latent need for checking and validating the content of knowledge graphs. We contribute to this problem in Chapter 5, but there are a few limitations and future works that we describe later in this chapter. This is still a niche problem and even the community has not decided yet what schema language should be the standard.

Despite all the attention towards knowledge graphs, there is no agreement about which technical approach should be adopted to knowledge graphs. Several works (including ours) have adopted an approach that uses a simplification of RDF graphs to represent and define knowledge graphs, but this is still not the standard in the community. We have identified that the two main areas of application where knowledge graphs are cited are machine learning and Semantic Web. In this thesis, it was not our goal to provide yet another definition, but we have used a definition that fits both of these areas. We have defined knowledge graphs as generic edge-labelled graphs similar to RDF graphs but leaving out some requirements such as the use of IRIs for identifying entities and relations. Moving forward, we consider that any definition for knowledge graph (different from ours) should leave out the following criteria:

(i) It should not be a requirement for knowledge graphs to cover various topical domains. They are powerful tools and can still represent knowledge in a well-bounded area of interest to support decision systems.

(ii) They should not necessarily contain ontological information as in the Semantic Web knowledge bases or RDF. But some sort of reasoning to extract new knowledge is required.

(iii) They should be seen as a system comprising facts and a reasoning engine that allows to reason and derive new knowledge (as proposed by Ehrlinger and Wöß (2016)).

(iv) They provide a common representation useful for integration, but they should not be required to follow the RDF standard for naming entities and relations.

(v) They are dynamic, their content (statements) evolve over time and their quality is not necessarily ensured when new statements are added. Thus, they require external systems for validation.

The completeness and completion problems for knowledge graphs have received significant attention in the last decade. As a dimension of knowledge graph quality, completeness is hard to measure and so far it has been defined based on an *ideal* knowledge graph. It is impossible to build such knowledge graph in real life, even for a small domain one can always find new properties for an entity. Approaches to assess how complete a knowledge graph is are much needed in areas such as Natural Language Processing and Knowledge Representation. It is particularly relevant as well, to analyse the consistency of knowledge graphs w.r.t. to a given shape graph. In Chapter 3, we have presented an extensive review of the knowledge graph completion problem (Section 3.3) and statistical relational learning models applied to the completion problem. Machine learning, and more specifically, deep learning, approaches have been used to generate distributed representations or embeddings for entities and relations in a knowledge graph that help to predict missing links and obtain state-of-the-art results in several applications. Such approaches are relevant for applications that require to move away from handcrafted features, since no feature engineering is required. In fact, features are learnt during the model training based on the optimisation of a loss function. Although they have shown to be effective in many tasks, their results lack of a direct interpretation. This causes that predictions made by machine learning models do not make sense or have any direct explanation from the observed data (Ribeiro, S. Singh, and Guestrin, 2016). That is the reason why non-latent feature approaches are still used when an interpretation of the results is required. We believe that initiatives to combine embeddings with formal logics (e.g., embeddings that satisfy logic rules) will gather even more attention in the future (S. Guo, Q. Wang, L. Wang, et al., 2016; Minervini, Costabello, Muñoz, et al., 2017; Ding, Q. Wang, B. Wang, et al., 2018; García-Durán and Niepert, 2018; S. Guo, Q. Wang, L. Wang, et al., 2018; Hamilton, Bajaj, Zitnik, et al., 2018; Liang, Z. Hu, H. Zhang, et al., 2018; Manhaeve, Dumancic, Kimmig, et al., 2018; M. Qu and Tang, 2019)

8.1.2 Part II: Latent shapes in knowledge graph

We have started part II with Chapter 4, by defining the notion of relation cardinality and introducing a principled approach for extracting cardinality bounds from instance data.

In our design, we have considered noisy knowledge graphs and proposed an algorithm able to extract accurate and robust cardinality bounds by leveraging equality axioms to deal with the UNA, and applying statistical analysis for identifying and removing outliers. Initially, we have implemented this approach using a complex query specified following the SPARQL query language (from the Semantic Web stack). This complex SPARQL query mines cardinality bounds and takes care of solving equality axioms on the fly. However, SPARQL engines suffer from scalability issues when processing complex queries over large-scale knowledge graphs. To overcome this issue, we have evaluated the Apache Spark scalable data processing framework to distribute the computation of cardinality bounds. (Note that few works recently have tried to evaluate SPARQL queries using Apache Spark (Graux, Jachiet, Genevès, et al., 2016; Agathangelos, Troullinou, Kondylakis, et al., 2018; Ayala, Koleva, Alzogbi, et al., 2019).) Spark provides distributed computing over large volumes of data and it was suitable for our cardinality mining algorithm. Our experiments showed that Apache Spark outperforms the SPARQL approach by up to 40x. Therefore, we have answered our RQ (1) by showing that it is possible to expose the structure of entity types just by looking at the cardinality of relations, and that such cardinality can be extracted efficiently from knowledge graphs. Furthermore, we have explored the use of the extracted cardinality bounds to measure consistency and completeness of a knowledge graph.

The curation and quality assurance of large knowledge graphs is an open problem for data consumers. In Chapter 5, we have introduced a new task, the approximate validation of knowledge graphs to help data managers with the task of validating knowledge graphs even when they are noisy and large. Although a solution for this task will not be exact, it is something required for an accuracy-efficiency trade-off. Current validation approaches take as input a schema and a knowledge graph, and they traverse or apply regular expressions over the whole knowledge graph looking for inconsistencies. Inspired by the notion of structure given by cardinality, we have developed a solution that extracts local patterns (subgraphs) to generate a vector representation for each node. Such subgraphs can be transformed into vectors in different ways. Here, we have vectorised the paths that compose a given subgraph. We consider two approaches for this: (a) a path exists in a subgraph or not (binary features), and (b) a path exists n times in the subgraph, where n is the cardinality of the path. This vector representation is then used to train a multi-class machine learning algorithm which is able to determine the validity of any given entity based on its neighbourhood subgraph. Our evaluation showed that if is possible to determine the right class for an entity based on the local patterns surrounding the entity. This is what we call latent shapes in this thesis. We have also demonstrated that by using a semi-supervised approach, we only need to know the label (validity) of a small portion of entities. Known labels are then extrapolate to similar entities according to the learnt patterns given by the latent shapes.

Once more, it was noted that the use of machine learning algorithms and vector representations of entities from knowledge graphs facilitate the development of more efficient solutions for knowledge graph mining tasks. The above results have provided enough information to answer our RQ (2). As future work (see Section 8.3), we would like to analyse

how to include shape information encoded as SHACL, ShEx, or other schema languages into our approach.

8.1.3 Part III: Knowledge graph mining applications

We have dedicated the whole part III to study use cases and applications of latent shape graphs. More specifically, we have focused on two problems: (a) the prediction of adverse drug reactions from Bioinformatics using a knowledge graph as data source, and (b) the enhancement of knowledge graph embeddings using a regularisation term based on cardinality constraints.

Biomedical data are heterogeneous, made available as different formats (mainly free-text publications) and stored using different technologies. Addressing this shortcoming, we build upon the Bio2RDF project (Dumontier, Callahan, Cruz-Toledo, et al., 2014), which aims to integrate multiple Biomedical data sources using the RDF format and a fixed schema. By doing so, this project built a large knowledge graph which homogenises the different data sources and becomes a rich resource for the area of Bioinformatics. Taking pieces of the Bio2RDF knowledge graph related to drugs, proteins, and side effects, we were able to present a novel approach for discovering adverse drug reactions using graph features and machine learning (Chapter 6). Similarly to what we did in Chapter 5, we have used cardinality and subgraphs to build vector representations for drugs (entities).

Given the nature of the problem, where a given drug can have multiple side effects or adverse reactions, we have treated the prediction problem as multi-label learning. We compared the performance of our approach against classical machine learning algorithms that work in the multi-learning set up using existing benchmarks and new ones that we propose in Chapter 6. The strength of our approach is demonstrated by an empirical evaluation against several machine learning models obtaining. We have showed the benefits of considering knowledge graphs for knowledge discovery in Bioinformatics and obtained new state-of-the-art results for the prediction of adverse drug reactions.

Existing applications of cardinality bounds (or constraints) in knowledge graphs are query optimisation (Papakonstantinou, Flouris, Fundulaki, et al., 2016), validation (Labra Gayo, E. Prud'hommeaux, Boneva, et al., 2018), and query re-writing (Lausen, Meier, and Schmidt, 2008), among others. However, during our initial study on knowledge graph embeddings, we have noticed that although these models are remarkably powerful in the link prediction task, they do not capture important statistical dependencies in the data. As part of this thesis, we contribute with the first attempt for using cardinalities to "guide" the learning process of knowledge graph embedding models. Our idea is to encode these constraint in the learning of relation embeddings, so that predictions satisfy the underlying cardinality as in "a person has exactly two parent". We have defined a regularisation term to achieve our goal. The cardinality regularisation term will penalise those parameters that violate the cardinality of relations during the computation of the loss function. We have

empirically analysed the benefits of our novel cardinality regulariser for several previously proposed knowledge graph embedding models and benchmark datasets, namely, Freebase, YAGO, and WordNet.

8.2 Limitations

The research we have carried out in this thesis has a number of limitations that we would like to discuss here.

Firstly, we have used several knowledge graphs to test different hypotheses across this thesis. Since many of these knowledge graphs are dynamic and evolving, we noticed that in many cases issues present in one version did not appear in a newer version or vice versa. For instance, the dumps we obtained from the Web Data Commons[6] repository are usually updated once a year, while others like PubChem are kept up to date with the latest discoveries in the Bioinformatics field. This is a very frequent problem that depends on several factors, where the most relevant is the organisations or groups behind the maintenance of the datasets. Knowledge graph construction is also limited by resources required to run expensive jobs that extract, load, and transform (ETL) data. We have seen an increasing interest from industry to support such activities with grants that cover hardware and other expenses. Examples of this are Amazon Open Data portal[7] and Google Cloud Public Datasets[8] that allow researchers and organisations to store and share large datasets publicly keeping associated metadata. Thus, helping the reproducibility of research and ensuring that data is preserved beyond the end of projects.

In terms of the limitations of our contributions, we list them per chapter.

Chapter 4. We proposed the first algorithm for mining cardinality constraints from knowledge graphs with a lower and upper bound. The extraction of lower bounds can be defined by a set of rules; however, the extraction of upper bounds is more challenging. We used statistical outlier detection for avoiding extreme upper bounds, but that is still prone to error. In our outlier detection, we assume that cardinalities follow a normal distribution that allows us to identify "extreme" values. Other types of distributions should be tested not only over DBpedia but also other more noisy knowledge graphs.

Chapter 5. Validation of knowledge graphs is hard due to their complex structure and to the infinite possible constraints that could be generated. Firstly, our proposed approximate algorithm assumes that an entity belongs to a single class at the time of validation. In practice, this is not always true and entities can belong to multiple classes—as observed in the case of drugs and side effects in Chapter 6. The problem of validating entities that belong to multiple classes is still open and we think multi-label learning approaches could provide an alternative solution to reach a more complete validation. Also, even though we tested nav-

[6] http://webdatacommons.org/
[7] https://aws.amazon.com/opendata/
[8] https://cloud.google.com/public-datasets/

igating the graph with a combination of depth-first and breath-first, and included inverse relations, we believe optimisation approaches for random walks will provide efficiency to get meaningful subgraphs faster. A filtering step could be introduced to clean our training data from: (a) spurious samples that do not have a minimum number of non-null (or non-zero) features present in their vector representations; and (b) irrelevant feature columns that do not pass a feature selection approach or criteria because of their low entropy, for example. Such filtering as a pre-step to training could help to generate better training data and more accurate models.

A future direction could go into testing knowledge graph embedding approaches (see Section 3.4) to generate the vector representations of entities. However, few considerations should be taken for considering unseen entities. We will discuss this in Section 8.3. Finally, the proposed method does not provide means to validate cycle-free (`schema:children`) or irreflexive (`schema:knows`) relations. Additional features should be extracted to teach the machine learning algorithms how to identify and score those cases, which are more frequently found in the Web.

Chapter 6. In our experiments to predict adverse drug reactions, we have considered graph-based features in order to provide interpretability. Some of these features come from what are considered functional properties, i.e., properties that can have only one (unique) value for each resource and there are no two distinct resources with the same property value (e.g. identifiers). We manually removed features with functional properties from our training dataset since they do not have predictive power. We consider that an automatic feature selection could have been done for this. It would also be nice to evaluate how knowledge graph models perform in this task.

On the other hand, although most features are considered as categorical or numerical, few free-text (given in plain English) features can also be found in Bio2RDF. We did not consider them in our experiments, but it would be possible to include them by applying algorithms like Word2Vec (Mikolov, K. Chen, Corrado, et al., 2013) or GloVe (Pennington, Socher, and Manning, 2014). Word embeddings could be added for consideration when computing similarity between drugs.

Finally, since the field of Bioinformatics and Pharmacology advances quite fast, models like the ones we proposed should be trained on the evolving data. We did not consider the effects that "new" links discovered during the predictions will have over other predictions. Such considerations are important in sensible applications dealing of any machine learning or statistical model. Specially, when dealing with life-or-death cases and generally when dealing with patients.

Chapter 7. The training of knowledge graph embeddings, as any deep learning task, is not trivial. As a limitation, we consider that even more hyperparameters could be analysed and models to evaluate our approach. We tried to mitigate the explosive number of parameters by following previously used grid searches and parameters. Also, when it comes to model training, validating the convergence of our new loss (including the regularisation term) is

something that we did not carry out and leave as future work. Finally, finding new ways to encode not only cardinality but other logical constructs is a relevant research area that requires more attention—we say more about this in our future directions (Section 8.3).

Therefore, certainly, there is plenty of room for improvement regarding approaches to knowledge graph mining in general and the specific challenges we have addressed here. In the next section, we summarise possible future directions for research in the area.

8.3 Future directions

In this section, we give an outlook into the future of knowledge graph mining research. This is divided into three topics that we believe are crucial for the advancement of the area.

8.3.1 Dynamic knowledge graphs and definition of completeness

In Chapter 3, we have described how completeness of knowledge graphs can be computed based on an ideal (complete) knowledge graph that is impractical to obtain. Here, we focus on a different but related problem, the dynamically evolving knowledge graphs, where new facts become known in time. For example, consider a knowledge graph of presidents who are elected every couple of years, where new entities will appear and replace old ones as president of a country. The new presidents will also bring associated neighbourhoods such as their families, *alma mater*, etc. For these scenarios, there is currently no link prediction (or machine learning) model that can handle unseen entities without the need of retraining every time a new entity (e.g., president) is added. Q. Wang, P. Huang, H. Wang, et al. (2019) takes into account that entities and relations appear in different contexts and should not be assigned a fixed embedding representation. Instead, they propose an approach that learns dynamic, flexible and contextualised embeddings for entities and relations in a knowledge graph. Note that dynamicity is a characteristic not frequently considered when defining knowledge graphs as they are most of the time seen as static for analysis.

Completion approaches reviewed in this thesis do assume that the only missing elements are links—the link prediction problem. However, a few approaches have considered the problem of missing entities or out-of-vocabulary entities (Z. Wang, Jianwen Zhang, J. Feng, et al., 2014a; Hamaguchi, Oiwa, Shimbo, et al., 2017), which are entities not present during training. Such approaches usually make use of external text describing the entities or clustering (neighbourhood comparisons) to assign an embedding representation to previously unseen entities. To achieve their goal, they usually optimise more complex loss functions and add extra parameters to the model. In the future, we would like to consider practical applications of latent shape graphs to the problem of dynamic knowledge graphs, as a mean to provide an idea of boundaries for what is missing akin to works such as González and Hogan (2018).

8.3.2 Veracity of statements

Along with dynamicity of knowledge graphs, we consider that data pollution and malicious attacks are latent problems that will become highly relevant. Especially, knowledge graphs with commercial purpose, e.g., Google's Knowledge Graph could be polluted to increase the ranking of a website. One must consider that most knowledge graph mining approaches assume that statements in the knowledge graph are trustworthy. Clearly, such an assumption is weak and it should be assumed that knowledge graphs contain noise and erroneous statements, and that one of their goals is to be 100% correct (even if it is not 100% complete). We believe that existing tools developed for fact checking could help with the identification and protection against such malicious content. This problem has been previously highlighted in the Semantic Web by Isele, Jentzsch, and Bizer (2010), Hasnain, Al-Bakri, Costabello, et al. (2012), and Muñoz, Aldarra, and Costabello (2017) Moreover, new articles have appeared proposing methods to identify the trustiness of statements in a knowledge graph (S. Liu, d'Aquin, and Motta, 2017; Jia, Xiang, X. Chen, et al., 2019; Zhao, H. Feng, and Gallinari, 2019). As the multi-linguality of knowledge graphs increase (M. Chen, Tian, M. Yang, et al., 2017), it will also pose challenges that will require more attention for fact validation.

The validation of new facts is also highly relevant when knowledge discovery approaches, such as the link prediction ones that we have focused in this thesis, predict triples that are true in the real world but not in the knowledge graph. In those cases, such predictions are counted as errors since they are *unknown*, thus false under the closed-world assumption. In Chapter 6, we addressed this issue by training and testing our models using a knowledge graph with valid facts at a time t, and then validating our "novel" prediction on a knowledge graph including known facts from time $t + 1$ and thereafter.

8.3.3 Embeddings, logics, and scalability

First, we consider that more research is required for the problem of automatic extraction of constraints from knowledge graphs at scale. We are convinced that latent schemas will play a huge role in the consumption and creation of knowledge graphs, since fixed schemas have limited expressibility. Moving forward, the requirements for embeddings to satisfy logical constraints will probably become a default. Several authors (including our work in this thesis) have already proposed ways to inject logical constraints in knowledge graph embeddings (Q. Wang, B. Wang, and L. Guo, 2015; Manhaeve, Dumancic, Kimmig, et al., 2018; Rocktäschel, 2018; H. Gao, X. Zheng, W. Li, et al., 2019). But they suffer from the increasing number of parameters and complexity, which is not desirable due to the demanding computing power and time required to train these models.

However, other research areas in machine learning could provide a solution to the previously mentioned scalability issues. Many scoring functions have been proposed to account for nuances in knowledge graphs, and no idea seems to have been left uncovered. We believe that treating knowledge graph embedding models as neural networks and testing

different architectures will help. Thus, the completion problem can be transformed in a Neural Architecture Search (NAS) problem (Zoph and Le, 2017). NAS is considered a subfield of Automatic Machine Learning (AutoML) (Feurer, Klein, Eggensperger, et al., 2015). NAS can help to find the best embedding model for a given knowledge graph and task.

8.3.4 Automatic benchmark generation

Rigorously evaluating a knowledge graph embedding model's performance requires access to gold standard evaluation benchmarks. As knowledge graphs become increasingly popular, there is a need for such benchmarks to test different hypotheses. Benchmark datasets should clearly come with attached metadata and annotations necessary for testing different models. This is also highlighted by the reproducibility issues found in literature, when authors test their approaches in closed access datasets it is not possible to reproduce and validate their results. See McDermott, S. Wang, Marinsek, et al. (2019) and Mittal, Pandey, and Kant (2019) for recent discussions about reproducibility in machine learning. (A similar situation applies for hardware requirements used in experiments.) We consider necessary more work towards the systematic generation of benchmark knowledge graphs, which posses some desired properties like the ones studied by Fernández, Martínez-Prieto, Fuente Redondo, et al. (2018). Clearly, such activities will depend primarily on the resources available during the construction of knowledge graphs. And both the construction algorithm and the output should be accessible and trustworthy. Only in this way, we see that academic environments will be able to increasingly adopt large scale knowledge graphs. For example, the biggest knowledge graph reported is found in industry[9], Google's Knowledge Graph (ca. 70 billion triples), while on the open side closer competitors are Wikidata (57 million triples) and DBpedia (9.5 billion triples). Furthermore, research will benefit from more clarity on the characteristics of available datasets. For example, WN18 and FB15k—some of the most used datasets for knowledge graph completion—were long used before Dettmers, Minervini, Stenetorp, et al. (2018) found out that they suffer from test set leakage, given that inverse relations from the train set are present in the test set.

Finally, we believe that there is a lack of evaluation metrics that best represent the completion requirements for knowledge graphs. Standard evaluation metrics such as MR, MRR, and Hits@k are still not enough to show the efficiency of models in real-world scenarios.

Hopefully, in this thesis we have contributed to the nascent area of knowledge graph mining with valuable insights and algorithms for exploiting their latent shape graphs. The use of latent shape graphs has boosted the exploitation of knowledge graphs and opened the way for new applications in different areas (e.g., Bioinformatics). In this thesis, we have just touched on some of the many research directions and (humbly) hope to inspire many more approaches in the area.

[9] We did not find any public description for other knowledge graphs such as Bing Satori, LinkedIn, Facebook, Amazon, among others.

Bibliography

Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, et al. (2016). 'TensorFlow: A System for Large-Scale Machine Learning'. In: *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016.* Ed. by Kimberly Keeton and Timothy Roscoe. USENIX Association, pp. 265–283.

Abedjan, Ziawasch, Toni Grütze, Anja Jentzsch, and Felix Naumann (2014). 'Profiling and mining RDF data with ProLOD++'. In: *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*. Ed. by Isabel F. Cruz, Elena Ferrari, Yufei Tao, Elisa Bertino, and Goce Trajcevski. IEEE Computer Society, pp. 1198–1201.

Abiteboul, Serge, Richard Hull, and Victor Vianu (1995). *Foundations of Databases*. Addison-Wesley. ISBN: 0-201-53771-0.

Adewale, Oluwatosin, Alex Beatson, Davit Buniatyan, Jason Ge, Mikhail Khodak, Holden Lee, Niranjani Prasad, Nikunj Saunshi, Ari Seff, Karan Singh, Daniel Suo, Cyril Zhang, et al. (2017). 'Pixie: A Social Chatbot'. In: *Alexa Prize Proceedings*.

Agathangelos, Giannis, Georgia Troullinou, Haridimos Kondylakis, Kostas Stefanidis, and Dimitris Plexousakis (2018). 'RDF Query Answering Using Apache Spark: Review and Assessment'. In: *34th IEEE International Conference on Data Engineering Workshops, ICDE Workshops 2018, Paris, France, April 16-20, 2018*. IEEE Computer Society, pp. 54–59.

Aldarra, Suad and Emir Muñoz (2015). 'A Linked Data-Based Decision Tree Classifier to Review Movies'. In: *Proceedings of the 4th Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data co-located with 12th Extended Semantic Web Conference (ESWC 2015), Portoroz, Slovenia, May 31, 2015.* Ed. by Johanna Völker, Heiko Paulheim, Jens Lehmann, and Vojtech Svátek. Vol. 1365. CEUR Workshop Proceedings. CEUR-WS.org.

Aldarra, Suad, Emir Muñoz, Pierre-Yves Vandenbussche, and Vít Nováček (2014). 'SemanTex: Semantic Text Exploration Using Document Links Implied by Conceptual Networks Extracted from the Texts'. In: *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.* Ed. by Matthew Horridge, Marco Rospocher, and Jacco van Ossenbruggen. Vol. 1272. CEUR Workshop Proceedings. CEUR-WS.org, pp. 345–348.

Arenas, Marcelo, Sebastián Conca, and Jorge Pérez (2012). 'Counting beyond a Yottabyte, or how SPARQL 1.1 property paths will prevent adoption of the standard'. In: *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*. Ed. by Alain Mille, Fabien L. Gandon, Jacques Misselis, Michael Rabinovich, and Steffen Staab. ACM, pp. 629–638.

Arenas, Marcelo, Claudio Gutiérrez, and Jorge Pérez (2009). 'Foundations of RDF Databases'. In: *Reasoning Web. Semantic Technologies for Information Systems, 5th International Summer School 2009, Brixen-Bressanone, Italy, August 30 - September 4, 2009, Tutorial Lectures*. Ed. by Sergio Tessaris, En-

rico Franconi, Thomas Eiter, Claudio Gutiérrez, Siegfried Handschuh, Marie-Christine Rousset, and Renate A. Schmidt. Vol. 5689. Lecture Notes in Computer Science. Springer, pp. 158–204.

Athreya, Ram G., Axel-Cyrille Ngonga Ngomo, and Ricardo Usbeck (2018). 'Enhancing Community Interactions with Data-Driven Chatbots-The DBpedia Chatbot'. In: *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*. Ed. by Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis. ACM, pp. 143–146.

Atias, Nir and Roded Sharan (2011). 'An algorithmic framework for predicting side effects of drugs'. In: *Journal of Computational Biology* 18.3, pp. 207–218.

Auer, Sören, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives (2007). 'DBpedia: A Nucleus for a Web of Open Data'. In: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.* Ed. by Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux. Vol. 4825. Lecture Notes in Computer Science. Springer, pp. 722–735.

Ayala, Victor Anthony Arrascue, Polina Koleva, Anas Alzogbi, Matteo Cossu, Michael Färber, Patrick Philipp, Guilherme Schievelbein, Io Taxidou, and Georg Lausen (2019). 'Relational schemata for distributed SPARQL query processing'. In: *Proceedings of the International Workshop on Semantic Big Data, SBD@SIGMOD 2019, Amsterdam, The Netherlands, July 5, 2019*. Ed. by Sven Groppe and Le Gruenwald. ACM, 3:1–3:6.

Baazizi, Mohamed Amine, Dario Colazzo, Giorgio Ghelli, and Carlo Sartiani (2019). 'Parametric schema inference for massive JSON datasets'. In: *VLDB J.* 28.4, pp. 497–521.

Baeza, Pablo Barceló (2013). 'Querying graph databases'. In: *Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2013, New York, NY, USA - June 22 - 27, 2013*. Ed. by Richard Hull and Wenfei Fan. ACM, pp. 175–188.

Bake, Thomas and Eric Prud'hommeaux (2017). *Shape Expressions (ShEx) Primer*. http://shex.io/shex-primer/.

Banda, Juan M., Lee Evans, Rami S. Vanguri, Nicholas P. Tatonetti, Patrick B. Ryan, and Nigam H. Shah (May 2016). 'A curated and standardized adverse drug event resource to accelerate drug safety research'. In: *Scientific Data* 3.

Beek, Wouter, Laurens Rietveld, Hamid R. Bazoobandi, Jan Wielemaker, and Stefan Schlobach (2014). 'LOD Laundromat: A Uniform Way of Publishing Other People's Dirty Data'. In: *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*. Ed. by Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandecic, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz, and Carole A. Goble. Vol. 8796. Lecture Notes in Computer Science. Springer, pp. 213–228.

Belleau, François, Marc-Alexandre Nolin, Nicole Tourigny, Philippe Rigault, and Jean Morissette (Oct. 2008). 'Bio2RDF: Towards a Mashup to Build Bioinformatics Knowledge Systems'. In: *J. of Biomedical Informatics* 41.5, pp. 706–716. ISSN: 1532-0464.

Bengio, Yoshua, Régis Cardin, Renato de Mori, and Piero Cosi (1988). 'Use of Multi-Layered Networks for Coding Speech with Phonetic Features'. In: *Advances in Neural Information Processing Systems 1, [NIPS Conference, Denver, Colorado, USA, 1988]*. Ed. by David S. Touretzky. Morgan Kaufmann, pp. 224–231.

Bengio, Yoshua, Aaron C. Courville, and Pascal Vincent (2012). 'Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives'. In: *CoRR* abs/1206.5538.

Berant, Jonathan, Andrew Chou, Roy Frostig, and Percy Liang (2013). 'Semantic Parsing on Freebase from Question-Answer Pairs'. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, pp. 1533–1544.

Berners-Lee, Tim, James Hendler, and Ora Lassila (May 2001). 'The Semantic Web'. In: *Scientific American*.

Bex, Geert Jan, Wouter Gelade, Frank Neven, and Stijn Vansummeren (2010). 'Learning Deterministic Regular Expressions for the Inference of Schemas from XML Data'. In: *TWEB* 4.4, 14:1–14:32.

Bischoff, Kerstin (2012). 'We love rock 'n' roll: analyzing and predicting friendship links in Last.fm'. In: *Web Science 2012, WebSci '12, Evanston, IL, USA - June 22 - 24, 2012*. Ed. by Noshir S. Contractor, Brian Uzzi, Michael W. Macy, and Wolfgang Nejdl. ACM, pp. 47–56.

Bollacker, Kurt D., Robert P. Cook, and Patrick Tufts (2007). 'Freebase: A Shared Database of Structured General Human Knowledge'. In: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*. AAAI Press, pp. 1962–1963.

Boneva, Iovka, José Emilio Labra Gayo, Samuel Hym, Eric G. Prud'hommeaux, Harold R. Solbrig, and Slawek Staworko (2014). 'Validating RDF with Shape Expressions'. In: *CoRR* abs/1404.1270.

Boneva, Iovka, José Emilio Labra Gayo, and Eric G. Prud'hommeaux (2017). 'Semantics and Validation of Shapes Schemas for RDF'. In: *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*. Ed. by Claudia d'Amato, Miriam Fernández, Valentina A. M. Tamma, Freddy Lécué, Philippe Cudré-Mauroux, Juan F. Sequeda, Christoph Lange, and Jeff Heflin. Vol. 10587. Lecture Notes in Computer Science. Springer, pp. 104–120.

Bordes, Antoine, Sumit Chopra, and Jason Weston (2014). 'Question Answering with Subgraph Embeddings'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. ACL, pp. 615–620.

Bordes, Antoine, Xavier Glorot, Jason Weston, and Yoshua Bengio (2012). 'Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing'. In: *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, Spain, April 21-23, 2012*. Ed. by Neil D. Lawrence and Mark A. Girolami. Vol. 22. JMLR Proceedings. JMLR.org, pp. 127–135.

Bordes, Antoine, Xavier Glorot, Jason Weston, and Yoshua Bengio (2014). 'A semantic matching energy function for learning with multi-relational data - Application to word-sense disambiguation'. In: *Machine Learning* 94.2, pp. 233–259.

Bordes, Antoine, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko (2013). 'Translating Embeddings for Modeling Multi-relational Data'. In: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.* Ed. by Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, pp. 2787–2795.

Bordes, Antoine, Jason Weston, Ronan Collobert, and Yoshua Bengio (2011). 'Learning Structured Embeddings of Knowledge Bases'. In: *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*. Ed. by Wolfram Burgard and Dan Roth. AAAI Press.

Bordes, Antoine, Jason Weston, and Nicolas Usunier (2014). 'Open Question Answering with Weakly Supervised Embedding Models'. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I*. Ed. by Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo. Vol. 8724. Lecture Notes in Computer Science. Springer, pp. 165–180.

Bosch, Thomas, Erman Acar, Andreas Nolle, and Kai Eckert (2015). 'The role of reasoning for RDF validation'. In: *Proceedings of the 11th International Conference on Semantic Systems, SEMANTICS 2015, Vienna, Austria, September 15-17, 2015*. Ed. by Axel Polleres, Tassilo Pellegrini, Sebastian Hellmann, and Josiane Xavier Parreira. ACM, pp. 33–40.

Bosch, Thomas and Kai Eckert (2015). 'Guidance, Please! Towards a Framework for RDF-Based Constraint Languages'. In: *Proc. of the International Conference on Dublin Core and Metadata Applications*.

Botev, Aleksandar, Bowen Zheng, and David Barber (2017). 'Complementary Sum Sampling for Likelihood Approximation in Large Scale Classification'. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*. Ed. by Aarti Singh and Xiaojin (Jerry) Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, pp. 1030–1038.

Bowes, Joanne, Andrew J Brown, Jacques Hamon, Wolfgang Jarolimek, Arun Sridhar, Gareth Waldron, and Steven Whitebread (2012). 'Reducing safety-related drug attrition: the use of in vitro pharmacological profiling'. In: *Nature reviews Drug discovery* 11.12, pp. 909–922.

Bresso, Emmanuel, Renaud Grisoni, Gino Marchetti, Arnaud Sinan Karaboga, Michel Souchet, Marie-Dominique Devignes, and Malika Smaïl-Tabbone (2013). 'Integrative relational machine-learning for understanding drug side-effect profiles'. In: *BMC bioinformatics* 14.1, p. 1.

Brickley, Dan, R.V. Guha, and Brian McBride (2014). *RDF Schema 1.1*. https://www.w3.org/TR/rdf-schema/.

Calvanese, Diego, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi (2000). 'Containment of Conjunctive Regular Path Queries with Inverse'. In: *KR 2000, Principles of Knowledge Representation and Reasoning Proceedings of the Seventh International Conference, Breckenridge, Colorado, USA, April 11-15, 2000*. Ed. by Anthony G. Cohn, Fausto Giunchiglia, and Bart Selman. Morgan Kaufmann, pp. 176–185.

Carlson, Andrew, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell (2010). 'Toward an Architecture for Never-Ending Language Learning'. In: *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. Ed. by Maria Fox and David Poole. AAAI Press.

Carroll, J. Douglas and Jih-Jie Chang (Sept. 1970). 'Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition'. In: *Psychometrika* 35.3, pp. 283–319. ISSN: 1860-0980.

Čebirić, Šejla, François Goasdoué, Haridimos Kondylakis, Dimitris Kotzinos, Ioana Manolescu, Georgia Troullinou, and Mussab Zneika (2019). 'Summarizing semantic graphs: a survey'. In: *VLDB J.* 28.3, pp. 295–327.

Čebirić, Šejla, François Goasdoué, and Ioana Manolescu (2015). 'Query-Oriented Summarization of RDF Graphs'. In: *Data Science - 30th British International Conference on Databases, BICOD 2015, Edinburgh, UK, July 6-8, 2015, Proceedings*. Ed. by Sebastian Maneth. Vol. 9147. Lecture Notes in Computer Science. Springer, pp. 87–91.

Chapelle, Olivier, Bernhard Schölkopf, and Alexander Zien, eds. (2006). *Semi-Supervised Learning*. The MIT Press. ISBN: 9780262033589.

Chen, Muhao, Yingtao Tian, Mohan Yang, and Carlo Zaniolo (2017). 'Multilingual Knowledge Graph Embeddings for Cross-lingual Knowledge Alignment'. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. Ed. by Carles Sierra. ijcai.org, pp. 1511–1517.

Chen, Peter P. (1976). 'The Entity-Relationship Model - Toward a Unified View of Data'. In: *ACM Trans. Database Syst.* 1.1, pp. 9–36.

Chen, Wenhu, Wenhan Xiong, Xifeng Yan, and William Yang Wang (2018). 'Variational Knowledge Graph Reasoning'. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Ed. by Marilyn A. Walker, Heng Ji, and Amanda Stent. Association for Computational Linguistics, pp. 1823–1832.

Choi, Seung-seok, Sung-hyuk Cha, and Charles C Tappert (2010). 'A Survey of Binary Similarity and Distance Measures'. In: *Journal on Systemics, Cybernetics and Informatics* 0.1, pp. 43–48.

Christodoulou, Klitos, Norman W. Paton, and Alvaro A. A. Fernandes (2015). 'Structure Inference for Linked Data Sources Using Clustering'. In: *Trans. Large-Scale Data- and Knowledge-Centered Systems* 19, pp. 1–25.

Cimiano, Philipp, Andreas Hotho, and Steffen Staab (2004). 'Comparing Conceptual, Divise and Agglomerative Clustering for Learning Taxonomies from Text'. In: *Proceedings of the 16th European Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*. Ed. by Ramón López de Mántaras and Lorenza Saitta. IOS Press, pp. 435–439.

Cochez, Michael, Petar Ristoski, Simone Paolo Ponzetto, and Heiko Paulheim (2017). 'Global RDF Vector Space Embeddings'. In: *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*. Ed. by Claudia d'Amato, Miriam Fernández, Valentina A. M. Tamma, Freddy Lécué, Philippe Cudré-Mauroux, Juan F. Sequeda, Christoph Lange, and Jeff Heflin. Vol. 10587. Lecture Notes in Computer Science. Springer, pp. 190–207.

Codd, E. F. (1970). 'A Relational Model of Data for Large Shared Data Banks'. In: *Commun. ACM* 13.6, pp. 377–387.

Corman, Julien, Juan L. Reutter, and Ognjen Savkovic (2018a). 'Semantics and Validation of Recursive SHACL'. In: *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I*. Ed. by Denny Vrandecic, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee, and Elena Simperl. Vol. 11136. Lecture Notes in Computer Science. Springer, pp. 318–336.

Corman, Julien, Juan L. Reutter, and Ognjen Savkovic (2018b). 'Towards a Robust Semantics for SHACL: Preliminary Discussion'. In: *Proceedings of the 12th Alberto Mendelzon International Workshop on Foundations of Data Management, Cali, Colombia, May 21-25, 2018*. Ed. by Dan Olteanu and Barbara Poblete. Vol. 2100. CEUR Workshop Proceedings. CEUR-WS.org.

Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein (2009). *Introduction to Algorithms, 3rd Edition*. MIT Press. ISBN: 978-0-262-03384-8.

Coyle, Karen and Tom Baker (2013). 'Dublin Core Application Profiles'. In: *W3C RDF Validation discussion*.

Cyganiak, Richard, David Wood, and Markus Lanthaler (Feb. 2014). *RDF 1.1 Concepts and Abstract Syntax*. https://www.w3.org/TR/rdf11-concepts/.

d'Amato, Claudia, Nicola Fanizzi, and Floriana Esposito (2010). 'Inductive learning for the Semantic Web: What does it buy?' In: *Semantic Web* 1.1-2, pp. 53–59.

Davis, Jesse and Mark Goadrich (2006). 'The relationship between Precision-Recall and ROC curves'. In: *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*. Ed. by William W. Cohen and Andrew W. Moore. Vol. 148. ACM International Conference Proceeding Series. ACM, pp. 233–240.

Dettmers, Tim, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel (2018). 'Convolutional 2D Knowledge Graph Embeddings'. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*. Ed. by Sheila A. McIlraith and Kilian Q. Weinberger. AAAI Press.

Dignum, Frank and Reind P. van de Riet (1991). 'Knowledge base modelling based on linguistics and founded in logic'. In: *Data Knowl. Eng.* 7, pp. 1–34.

Ding, Boyang, Quan Wang, Bin Wang, and Li Guo (2018). 'Improving Knowledge Graph Embedding Using Simple Constraints'. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Ed. by Iryna Gurevych and Yusuke Miyao. Association for Computational Linguistics, pp. 110–121.

Dong, Xin Luna (2019). 'Building a Broad Knowledge Graph for Products'. In: *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*. IEEE, p. 25.

Dong, Xin, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang (2014). 'Knowledge vault: a web-scale approach to probabilistic knowledge fusion'. In: *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. Ed. by Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani. ACM, pp. 601–610.

Drumond, Lucas, Steffen Rendle, and Lars Schmidt-Thieme (2012). 'Predicting RDF triples in incomplete knowledge bases with tensor factorization'. In: *Proceedings of the ACM Symposium on Applied Computing, SAC 2012, Riva, Trento, Italy, March 26-30, 2012*. Ed. by Sascha Ossowski and Paola Lecca. ACM, pp. 326–331.

Duchi, John C., Elad Hazan, and Yoram Singer (2011). 'Adaptive Subgradient Methods for Online Learning and Stochastic Optimization'. In: *Journal of Machine Learning Research* 12, pp. 2121–2159.

Dumontier, Michel, Alison Callahan, Jose Cruz-Toledo, Peter Ansell, Vincent Emonet, François Belleau, and Arnaud Droit (2014). 'Bio2RDF Release 3: A larger, more connected network of Linked Data for the Life Sciences'. In: *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014*. Ed. by Matthew Horridge, Marco Rospocher, and Jacco van Ossenbruggen. Vol. 1272. CEUR Workshop Proceedings. CEUR-WS.org, pp. 401–404.

Ehrlinger, Lisa and Wolfram Wöß (2016). 'Towards a Definition of Knowledge Graphs'. In: *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) co-located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016*. Ed. by Michael Martin, Martí Cuquet, and Erwin Folmer. Vol. 1695. CEUR Workshop Proceedings. CEUR-WS.org.

Ellefi, Mohamed Ben, Zohra Bellahsene, John G. Breslin, Elena Demidova, Stefan Dietze, Julian Szymanski, and Konstantin Todorov (2018). 'RDF dataset profiling - a survey of features, methods, vocabularies and applications'. In: *Semantic Web* 9.5, pp. 677–705.

Elsken, Thomas, Jan Hendrik Metzen, and Frank Hutter (2019). 'Neural Architecture Search: A Survey'. In: *J. Mach. Learn. Res.* 20, 55:1–55:21.

Ermilov, Ivan, Jens Lehmann, Michael Martin, and Sören Auer (2016). 'LODStats: The Data Web Census Dataset'. In: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe,*

Japan, October 17-21, 2016, Proceedings, Part II. Ed. by Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil. Vol. 9982. Lecture Notes in Computer Science, pp. 38–46.

Erxleben, Fredo, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandecic (2014). 'Introducing Wikidata to the Linked Data Web'. In: *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*. Ed. by Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandecic, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz, and Carole A. Goble. Vol. 8796. Lecture Notes in Computer Science. Springer, pp. 50–65.

Fernández-Álvarez, Daniel, Herminio García-González, Johannes Frey, Sebastian Hellmann, and José Emilio Labra Gayo (2018). 'Inference of Latent Shape Expressions Associated to DBpedia Ontology'. In: *Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th - to - 12th, 2018*. Ed. by Marieke van Erp, Medha Atre, Vanessa López, Kavitha Srinivas, and Carolina Fortuna. Vol. 2180. CEUR Workshop Proceedings. CEUR-WS.org.

Fernández, Javier D., Miguel A. Martínez-Prieto, Pablo de la Fuente Redondo, and Claudio Gutiérrez (2018). 'Characterising RDF data sets'. In: *J. Inf. Sci.* 44.2, pp. 203–229.

Ferrarotti, Flavio, Sven Hartmann, and Sebastian Link (2013). 'Efficiency frontiers of XML cardinality constraints'. In: *Data Knowl. Eng.* 87, pp. 297–319.

Ferrarotti, Flavio, Sven Hartmann, Sebastian Link, Mauricio Marín, and Emir Muñoz (2013). 'Soft Cardinality Constraints on XML Data - How Exceptions Prove the Business Rule'. In: *Web Information Systems Engineering - WISE 2013 - 14th International Conference, Nanjing, China, October 13-15, 2013, Proceedings, Part I*. Ed. by Xuemin Lin, Yannis Manolopoulos, Divesh Srivastava, and Guangyan Huang. Vol. 8180. Lecture Notes in Computer Science. Springer, pp. 382–395.

Feurer, Matthias, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter (2015). 'Efficient and Robust Automated Machine Learning'. In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Ed. by Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, pp. 2962–2970.

Firebaugh, Morris W. (1988). *Artificial Intelligence: A Knowledge-based Approach*. Danvers, MA, USA: Boyd & Fraser Publishing Co. ISBN: 0-87835-325-9.

Fisher, Ronald A (1936). 'The use of multiple measurements in taxonomic problems'. In: *Annals of human genetics* 7.2, pp. 179–188.

Fokoue, Achille and Arthur Ryman (2013). 'OSLC Resource Shape: A Linked Data Constraint Language'. In: *W3C RDF Validation discussion*.

Fürber, Christian and Martin Hepp (2010). 'Using SPARQL and SPIN for Data Quality Management on the Semantic Web'. In: *Business Information Systems, 13th International Conference, BIS 2010, Berlin, Germany, May 3-5, 2010. Proceedings*. Ed. by Witold Abramowicz and Robert Tolksdorf. Vol. 47. Lecture Notes in Business Information Processing. Springer, pp. 35–46.

Galárraga, Luis Antonio, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek (2013). 'AMIE: association rule mining under incomplete evidence in ontological knowledge bases'. In: *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*. Ed. by Daniel Schwabe, Virgílio A. F. Almeida, Hartmut Glaser, Ricardo A. Baeza-Yates, and Sue B. Moon. International World Wide Web Conferences Steering Committee / ACM, pp. 413–422.

Galárraga, Luis, Simon Razniewski, Antoine Amarilli, and Fabian M. Suchanek (2017). 'Predicting Completeness in Knowledge Bases'. In: *Proceedings of the Tenth ACM International Conference on*

Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017. Ed. by Maarten de Rijke, Milad Shokouhi, Andrew Tomkins, and Min Zhang. ACM, pp. 375–383.

Galárraga, Luis, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek (2015). 'Fast rule mining in ontological knowledge bases with AMIE+'. In: *VLDB J.* 24.6, pp. 707–730.

Gao, Huan, Xianda Zheng, Weizhuo Li, Guilin Qi, and Meng Wang (2019). 'Cosine-Based Embedding for Completing Schematic Knowledge'. In: *Natural Language Processing and Chinese Computing - 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9-14, 2019, Proceedings, Part I*. Ed. by Jie Tang, Min-Yen Kan, Dongyan Zhao, Sujian Li, and Hongying Zan. Vol. 11838. Lecture Notes in Computer Science. Springer, pp. 249–261.

García-Durán, Alberto, Antoine Bordes, and Nicolas Usunier (2014). 'Effective Blending of Two and Three-way Interactions for Modeling Multi-relational Data'. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I*. Ed. by Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo. Vol. 8724. Lecture Notes in Computer Science. Springer, pp. 434–449.

García-Durán, Alberto and Mathias Niepert (2018). 'KBLRN: End-to-End Learning of Knowledge Base Representations with Latent, Relational, and Numerical Features'. In: *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*. Ed. by Amir Globerson and Ricardo Silva. AUAI Press, pp. 372–381.

Gardner, Matt and Tom M. Mitchell (2015). 'Efficient and Expressive Knowledge Base Completion Using Subgraph Feature Extraction'. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. Ed. by Lluís Márquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton. The Association for Computational Linguistics, pp. 1488–1498.

Getoor, Lise and Ben Taskar (2007). *Introduction to statistical relational learning*. MIT press.

Glimm, Birte, Aidan Hogan, Markus Krötzsch, and Axel Polleres (2012). 'OWL: Yet to arrive on the Web of Data?' In: *WWW2012 Workshop on Linked Data on the Web, Lyon, France, 16 April, 2012*. Ed. by Christian Bizer, Tom Heath, Tim Berners-Lee, and Michael Hausenblas. Vol. 937. CEUR Workshop Proceedings. CEUR-WS.org.

Glorot, Xavier and Yoshua Bengio (2010). 'Understanding the difficulty of training deep feedforward neural networks'. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*. Ed. by Yee Whye Teh and D. Mike Titterington. Vol. 9. JMLR Proceedings. JMLR.org, pp. 249–256.

González, Larry and Aidan Hogan (2018). 'Modelling Dynamics in Semantic Web Knowledge Graphs with Formal Concept Analysis'. In: *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. Ed. by Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis. ACM, pp. 1175–1184.

Goodfellow, Ian J., Yoshua Bengio, and Aaron C. Courville (2016). *Deep Learning*. Adaptive computation and machine learning. MIT Press. ISBN: 978-0-262-03561-3.

Grant, John and Anthony Hunter (2006). 'Measuring inconsistency in knowledgebases'. In: *J. Intell. Inf. Syst.* 27.2, pp. 159–184.

Graux, Damien, Louis Jachiet, Pierre Genevès, and Nabil Layaïda (2016). 'SPARQLGX: Efficient Distributed Evaluation of SPARQL with Apache Spark'. In: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II*. Ed. by Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil. Vol. 9982. Lecture Notes in Computer Science, pp. 80–87.

Grimm, Stephan and Boris Motik (2005). 'Closed World Reasoning in the Semantic Web through Epistemic Operators'. In: *Proceedings of the OWLED*05 Workshop on OWL: Experiences and Directions, Galway, Ireland, November 11-12, 2005*. Ed. by Bernardo Cuenca Grau, Ian Horrocks, Bijan Parsia, and Peter F. Patel-Schneider. Vol. 188. CEUR Workshop Proceedings. CEUR-WS.org.

Grimnes, Gunnar Aastrand, Peter Edwards, and Alun D. Preece (2004). 'Learning Meta-descriptions of the FOAF Network'. In: *The Semantic Web - ISWC 2004: Third International Semantic Web Conference,Hiroshima, Japan, November 7-11, 2004. Proceedings*. Ed. by Sheila A. McIlraith, Dimitris Plexousakis, and Frank van Harmelen. Vol. 3298. Lecture Notes in Computer Science. Springer, pp. 152–165.

Grover, Aditya and Jure Leskovec (2016). 'node2vec: Scalable Feature Learning for Networks'. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. Ed. by Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi. ACM, pp. 855–864.

Gruber, Thomas R. (1995). 'Toward principles for the design of ontologies used for knowledge sharing?' In: *Int. J. Hum.-Comput. Stud.* 43.5-6, pp. 907–928.

Guarino, Nicola, Daniel Oberle, and Steffen Staab (2009). 'What Is an Ontology?' In: *Handbook on Ontologies*. Ed. by Steffen Staab and Rudi Studer. International Handbooks on Information Systems. Springer, pp. 1–17.

Guha, Ramanathan V., Dan Brickley, and Steve Macbeth (2016). 'Schema.org: evolution of structured data on the web'. In: *Commun. ACM* 59.2, pp. 44–51.

Guo, Shu, Quan Wang, Lihong Wang, Bin Wang, and Li Guo (2016). 'Jointly Embedding Knowledge Graphs and Logical Rules'. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. Ed. by Jian Su, Xavier Carreras, and Kevin Duh. The Association for Computational Linguistics, pp. 192–202.

Guo, Shu, Quan Wang, Lihong Wang, Bin Wang, and Li Guo (2018). 'Knowledge Graph Embedding With Iterative Guidance From Soft Rules'. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. Ed. by Sheila A. McIlraith and Kilian Q. Weinberger. AAAI Press, pp. 4816–4823.

Gutiérrez, Claudio, Carlos A. Hurtado, Alberto O. Mendelzon, and Jorge Pérez (2011). 'Foundations of Semantic Web databases'. In: *J. Comput. Syst. Sci.* 77.3, pp. 520–541.

Haase, Peter, Andriy Nikolov, Johannes Trame, Artem Kozlov, and Daniel M. Herzig (2017). 'Alexa, Ask Wikidata! Voice Interaction with Knowledge Graphs using Amazon Alexa'. In: *Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017*. Ed. by Nadeschda Nikitina, Dezhao Song, Achille Fokoue, and Peter Haase. Vol. 1963. CEUR Workshop Proceedings. CEUR-WS.org.

Hakimov, Sherzod, Salih Atilay Oto, and Erdogan Dogdu (2012). 'Named entity recognition and disambiguation using linked data and graph-based centrality scoring'. In: *Proceedings of the 4th International Workshop on Semantic Web Information Management, SWIM 2012, Scottsdale, AZ, USA, May 20, 2012*. Ed. by Roberto De Virgilio, Fausto Giunchiglia, and Letizia Tanca. ACM, p. 4.

Hamaguchi, Takuo, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto (2017). 'Knowledge Transfer for Out-of-Knowledge-Base Entities : A Graph Neural Network Approach'. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. Ed. by Carles Sierra. ijcai.org, pp. 1802–1808.

Hamilton, William L., Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec (2018). 'Embedding Logical Queries on Knowledge Graphs'. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicoló Cesa-Bianchi, and Roman Garnett, pp. 2030–2041.

Harpaz, Rave, Santiago Vilar, William DuMouchel, Hojjat Salmasian, Krystl Haerian, Nigam H Shah, Herbert S Chase, and Carol Friedman (2013). 'Combing signals from spontaneous reports and electronic health records for detection of adverse drug reactions'. In: *Journal of the American Medical Informatics Association* 20.3, pp. 413–419.

Harris, Steve, Andy Seaborne, and Eric Prud'hommeaux (2013). *SPARQL 1.1 Query Language*. https://www.w3.org/TR/sparql11-query/.

Harris, Zellig S. (1954). 'Distributional Structure'. In: *WORD* 10.2-3, pp. 146–162.

Hartmann, Thomas (2016). 'Validation Framework for RDF-based Constraint Languages'. PhD thesis. Karlsruhe Institute of Technology, Germany.

Hasnain, Ali, Mustafa Al-Bakri, Luca Costabello, Zijie Cong, Ian Davis, and Tom Heat (2012). 'Spamming in Linked Data'. In: *Proceedings of the Third International Workshop on Consuming Linked Data, COLD 2012, Boston, MA, USA, November 12, 2012*. Ed. by Juan F. Sequeda, Andreas Harth, and Olaf Hartig. Vol. 905. CEUR Workshop Proceedings. CEUR-WS.org.

Hasnain, Ali, Qaiser Mehmood, Syeda Sana e Zainab, and Aidan Hogan (2016). 'SPORTAL: Profiling the Content of Public SPARQL Endpoints'. In: *Int. J. Semantic Web Inf. Syst.* 12.3, pp. 134–163.

Hayashi, Katsuhiko and Masashi Shimbo (2017). 'On the Equivalence of Holographic and Complex Embeddings for Link Prediction'. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*. Ed. by Regina Barzilay and Min-Yen Kan. Association for Computational Linguistics, pp. 554–559.

Hayes-Roth, Frederick (1983). *Building expert systems*. Vol. 1. Advanced book program. Addison-Wesley. ISBN: 0201106868.

Hayes, Jonathan and Claudio Gutiérrez (2004). 'Bipartite Graphs as Intermediate Model for RDF'. In: *The Semantic Web - ISWC 2004: Third International Semantic Web Conference,Hiroshima, Japan, November 7-11, 2004. Proceedings*. Ed. by Sheila A. McIlraith, Dimitris Plexousakis, and Frank van Harmelen. Vol. 3298. Lecture Notes in Computer Science. Springer, pp. 47–61.

Hellmann, Sebastian, Jens Lehmann, and Sören Auer (2009). 'Learning of OWL Class Descriptions on Very Large Knowledge Bases'. In: *Int. J. Semantic Web Inf. Syst.* 5.2, pp. 25–48.

Hogan, Aidan (2018). 'Profiling Graphs: Order from Chaos'. In: *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*. Ed. by Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis. ACM, pp. 1481–1482.

Hogan, Aidan, Andreas Harth, Alexandre Passant, Stefan Decker, and Axel Polleres (2010). 'Weaving the Pedantic Web'. In: *Proceedings of the WWW2010 Workshop on Linked Data on the Web, LDOW 2010, Raleigh, USA, April 27, 2010*. Ed. by Christian Bizer, Tom Heath, Tim Berners-Lee, and Michael Hausenblas. Vol. 628. CEUR Workshop Proceedings. CEUR-WS.org.

Hogan, Aidan, Jürgen Umbrich, Andreas Harth, Richard Cyganiak, Axel Polleres, and Stefan Decker (2012). 'An empirical survey of Linked Data conformance'. In: *J. Web Semant.* 14, pp. 14–44.

Horrocks, Ian, Bijan Parsia, Peter F. Patel-Schneider, and James A. Hendler (2005). 'Semantic Web Architecture: Stack or Two Towers?' In: *Principles and Practice of Semantic Web Reasoning, Third International Workshop, PPSWR 2005, Dagstuhl Castle, Germany, September 11-16, 2005, Proceed-*

ings. Ed. by François Fages and Sylvain Soliman. Vol. 3703. Lecture Notes in Computer Science. Springer, pp. 37–41.

Horrocks, Ian and Sergio Tessaris (2002). 'Querying the Semantic Web: A Formal Approach'. In: *The Semantic Web - ISWC 2002, First International Semantic Web Conference, Sardinia, Italy, June 9-12, 2002, Proceedings*. Ed. by Ian Horrocks and James A. Hendler. Vol. 2342. Lecture Notes in Computer Science. Springer, pp. 177–191.

Huang, Liang-Chin, Xiaogang Wu, and Jake Y Chen (2013). 'Predicting adverse drug reaction profiles by integrating protein interaction networks with drug structures'. In: *Proteomics* 13.2, pp. 313–324.

Isele, Robert, Anja Jentzsch, and Christian Bizer (2010). 'Silk Server - Adding missing Links while consuming Linked Data'. In: *Proceedings of the First International Workshop on Consuming Linked Data, Shanghai, China, November 8, 2010*. Ed. by Olaf Hartig, Andreas Harth, and Juan F. Sequeda. Vol. 665. CEUR Workshop Proceedings. CEUR-WS.org.

Ivakhnenko, A. G. (1971). 'Polynomial Theory of Complex Systems'. In: *IEEE Trans. Systems, Man, and Cybernetics* 1.4, pp. 364–378.

Jahid, Md Jamiul and Jianhua Ruan (2013). 'An ensemble approach for drug side effect prediction'. In: *Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on*. IEEE, pp. 440–445.

Jenatton, Rodolphe, Nicolas Le Roux, Antoine Bordes, and Guillaume Obozinski (2012). 'A latent factor model for highly multi-relational data'. In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. Ed. by Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, pp. 3176–3184.

Jensen, David D. and Jennifer Neville (2002). 'Linkage and Autocorrelation Cause Feature Selection Bias in Relational Learning'. In: *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002*. Ed. by Claude Sammut and Achim G. Hoffmann. Morgan Kaufmann, pp. 259–266.

Jia, Shengbin, Yang Xiang, Xiaojun Chen, Kun Wang, and Shijia E (2019). 'Triple Trustworthiness Measurement for Knowledge Graph'. In: *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. Ed. by Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia. ACM, pp. 2865–2871.

Johnson, David S. (1974). 'Approximation Algorithms for Combinatorial Problems'. In: *J. Comput. Syst. Sci.* 9.3, pp. 256–278.

Käfer, Tobias, Jürgen Umbrich, Aidan Hogan, and Axel Polleres (2012). 'DyLDO: Towards a Dynamic Linked Data Observatory'. In: *WWW2012 Workshop on Linked Data on the Web, Lyon, France, 16 April, 2012*. Ed. by Christian Bizer, Tom Heath, Tim Berners-Lee, and Michael Hausenblas. Vol. 937. CEUR Workshop Proceedings. CEUR-WS.org.

Kanehisa, Minoru, Miho Furumichi, Mao Tanabe, Yoko Sato, and Kanae Morishima (2017). 'KEGG: new perspectives on genomes, pathways, diseases and drugs'. In: *Nucleic Acids Research* 45.D1, pp. D353–D361.

Kellou-Menouer, Kenza and Zoubida Kedad (2015). 'Evaluating the Gap Between an RDF Dataset and Its Schema'. In: *Advances in Conceptual Modeling - ER 2015 Workshops, AHA, CMS, EMoV, MoBiD, MORE-BI, MReBA, QMMQ, and SCME Stockholm, Sweden, October 19-22, 2015, Proceedings*. Ed. by Manfred A. Jeusfeld and Kamalakar Karlapalem. Vol. 9382. Lecture Notes in Computer Science. Springer, pp. 283–292.

Khatchadourian, Shahan and Mariano P. Consens (2010). 'ExpLOD: Summary-Based Exploration of Interlinking and RDF Usage in the Linked Open Data Cloud'. In: *The Semantic Web: Research and*

Applications, 7th Extended Semantic Web Conference, ESWC 2010, Heraklion, Crete, Greece, May 30 - June 3, 2010, Proceedings, Part II. Ed. by Lora Aroyo, Grigoris Antoniou, Eero Hyvönen, Annette ten Teije, Heiner Stuckenschmidt, Liliana Cabral, and Tania Tudorache. Vol. 6089. Lecture Notes in Computer Science. Springer, pp. 272–287.

Kim, Sunghwan, Paul A Thiessen, Evan E Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, Jane He, Siqian He, Benjamin A Shoemaker, et al. (2015). 'PubChem substance and compound databases'. In: *Nucleic acids research*.

Knublauch, Holger, James A. Hendler, and Kingsley Idehen (2011). *SPIN - Overview and Motivation*. https://www.w3.org/Submission/spin-overview/.

Knublauch, Holger and Dimitris Kontokostas (2017). *Shapes Constraint Language (SHACL)*. https://www.w3.org/TR/shacl/.

Kontokostas, Dimitris, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri (2014). 'Test-driven evaluation of linked data quality'. In: *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*. Ed. by Chin-Wan Chung, Andrei Z. Broder, Kyuseok Shim, and Torsten Suel. ACM, pp. 747–758.

Kostylev, Egor V., Juan L. Reutter, Miguel Romero, and Domagoj Vrgoc (2015). 'SPARQL with Property Paths'. In: *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*. Ed. by Marcelo Arenas, Óscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul T. Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab. Vol. 9366. Lecture Notes in Computer Science. Springer, pp. 3–18.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2012). 'ImageNet Classification with Deep Convolutional Neural Networks'. In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. Ed. by Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, pp. 1106–1114.

Krompaß, Denis, Maximilian Nickel, and Volker Tresp (2014). 'Querying Factorized Probabilistic Triple Databases'. In: *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II*. Ed. by Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandecic, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz, and Carole A. Goble. Vol. 8797. Lecture Notes in Computer Science. Springer, pp. 114–129.

Kuhn, Michael, Ivica Letunic, Lars Juhl Jensen, and Peer Bork (2015). 'The SIDER database of drugs and side effects'. In: *Nucleic acids research*.

Kyrola, Aapo (2013). 'DrunkardMob: billions of random walks on just a PC'. In: *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*. Ed. by Qiang Yang, Irwin King, Qing Li, Pearl Pu, and George Karypis. ACM, pp. 257–264.

Labra Gayo, José Emilio, Herminio García-González, Daniel Fernández-Alvarez, and Eric Prud'hommeaux (2019). 'Challenges in RDF Validation'. In: *Current Trends in Semantic Web Technologies: Theory and Practice*. Ed. by Giner Alor-Hernández, José Luis Sánchez-Cervantes, Alejandro Rodríguez-González, and Rafael Valencia-García. Cham: Springer International Publishing, pp. 121–151. ISBN: 978-3-030-06149-4.

Labra Gayo, José Emilio, Eric Prud'hommeaux, Iovka Boneva, and Dimitris Kontokostas (2018). *Validating RDF Data*. Vol. 7. Synthesis Lectures on the Semantic Web: Theory and Technology 1. Morgan & Claypool, p. 328.

Lajus, Jonathan and Fabian M. Suchanek (2018). 'Are All People Married?: Determining Obligatory Attributes in Knowledge Bases'. In: *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. Ed. by Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis. ACM, pp. 1115–1124.

Lao, Ni and William W. Cohen (2010). 'Relational retrieval using a combination of path-constrained random walks'. In: *Machine Learning* 81.1, pp. 53–67.

Lao, Ni, Tom M. Mitchell, and William W. Cohen (2011). 'Random Walk Inference and Learning in A Large Scale Knowledge Base'. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, pp. 529–539.

Lausen, Georg, Michael Meier, and Michael Schmidt (2008). 'SPARQLing constraints for RDF'. In: *EDBT 2008, 11th International Conference on Extending Database Technology, Nantes, France, March 25-29, 2008, Proceedings*. Ed. by Alfons Kemper, Patrick Valduriez, Noureddine Mouaddib, Jens Teubner, Mokrane Bouzeghoub, Volker Markl, Laurent Amsaleg, and Ioana Manolescu. Vol. 261. ACM International Conference Proceeding Series. ACM, pp. 499–509.

Law, Vivian, Craig Knox, Yannick Djoumbou, Tim Jewison, An Chi Guo, Yifeng Liu, Adam Maciejewski, David Arndt, Michael Wilson, Vanessa Neveu, et al. (2014). 'DrugBank 4.0: shedding new light on drug metabolism'. In: *Nucleic acids research* 42.D1, pp. D1091–D1097.

LeCun, Yann, Yoshua Bengio, and Geoffrey E. Hinton (2015). 'Deep learning'. In: *Nature* 521.7553, pp. 436–444.

LeCun, Yann, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel (1989). 'Backpropagation Applied to Handwritten Zip Code Recognition'. In: *Neural Computation* 1.4, pp. 541–551.

Lehmann, Jens (2009). 'DL-Learner: Learning Concepts in Description Logics'. In: *J. Mach. Learn. Res.* 10, pp. 2639–2642.

Lenat, Douglas B. (1995). 'CYC: A Large-Scale Investment in Knowledge Infrastructure'. In: *Commun. ACM* 38.11, pp. 32–38.

Levy, Alon Y. (1996). 'Obtaining Complete Answers from Incomplete Databases'. In: *VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases, September 3-6, 1996, Mumbai (Bombay), India*. Ed. by T. M. Vijayaraman, Alejandro P. Buchmann, C. Mohan, and Nandlal L. Sarda. Morgan Kaufmann, pp. 402–412.

Liang, Xiaodan, Zhiting Hu, Hao Zhang, Liang Lin, and Eric P. Xing (2018). 'Symbolic Graph Reasoning Meets Convolutions'. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicoló Cesa-Bianchi, and Roman Garnett, pp. 1858–1868.

Liddle, Stephen W., David W. Embley, and Scott N. Woodfield (1993). 'Cardinality Constraints in Semantic Data Models'. In: *Data Knowl. Eng.* 11.3, pp. 235–270.

Lin, Yankai, Zhiyuan Liu, Huan-Bo Luan, Maosong Sun, Siwei Rao, and Song Liu (2015). 'Modeling Relation Paths for Representation Learning of Knowledge Bases'. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. Ed. by Lluís Márquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton. The Association for Computational Linguistics, pp. 705–714.

Lin, Yankai, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu (2015). 'Learning Entity and Relation Embeddings for Knowledge Graph Completion'. In: *Proceedings of the Twenty-Ninth AAAI*

Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA. Ed. by Blai Bonet and Sven Koenig. AAAI Press, pp. 2181–2187.

Lisi, Francesca A. (2010). 'Inductive Logic Programming in Databases: From Datalog to DL+log'. In: *TPLP* 10.3, pp. 331–359.

Liu, Bo, Sebastian Link, and Emir Muñoz (2015). 'Validation of Expressive XML Keys with XML Schema and XQuery'. In: *11th Asia-Pacific Conference on Conceptual Modelling, APCCM 2015, Sydney, Australia, January 2015*. Ed. by Motoshi Saeki and Henning Köhler. Vol. 165. CRPIT. Australian Computer Society, pp. 81–90.

Liu, Mei, Yonghui Wu, Yukun Chen, Jingchun Sun, Zhongming Zhao, Xue-wen Chen, Michael Edwin Matheny, and Hua Xu (2012). 'Large-scale prediction of adverse drug reactions using chemical, biological, and phenotypic properties of drugs'. In: *Journal of the American Medical Informatics Association* 19.e1, e28–e35.

Liu, Quan, Hui Jiang, Zhen-Hua Ling, Si Wei, and Yu Hu (2016). 'Probabilistic Reasoning via Deep Learning: Neural Association Models'. In: *CoRR* abs/1603.07704.

Liu, Shuangyan, Mathieu d'Aquin, and Enrico Motta (2017). 'Measuring Accuracy of Triples in Knowledge Graphs'. In: *Language, Data, and Knowledge - First International Conference, LDK 2017, Galway, Ireland, June 19-20, 2017, Proceedings*. Ed. by Jorge Gracia, Francis Bond, John P. McCrae, Paul Buitelaar, Christian Chiarcos, and Sebastian Hellmann. Vol. 10318. Lecture Notes in Computer Science. Springer, pp. 343–357.

Liu, Tie-Yan (2011). *Learning to Rank for Information Retrieval*. Springer. ISBN: 978-3-642-14266-6.

Lösch, Uta, Stephan Bloehdorn, and Achim Rettinger (2012). 'Graph Kernels for RDF Data'. In: *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*. Ed. by Elena Simperl, Philipp Cimiano, Axel Polleres, Óscar Corcho, and Valentina Presutti. Vol. 7295. Lecture Notes in Computer Science. Springer, pp. 134–148.

Maaten, Laurens van der and Geoffrey Hinton (2008). 'Visualizing data using t-SNE'. In: *Journal of machine learning research* 9.Nov, pp. 2579–2605.

Maedche, Alexander and Valentin Zacharias (2002). 'Clustering Ontology-Based Metadata in the Semantic Web'. In: *Principles of Data Mining and Knowledge Discovery, 6th European Conference, PKDD 2002, Helsinki, Finland, August 19-23, 2002, Proceedings*. Ed. by Tapio Elomaa, Heikki Mannila, and Hannu Toivonen. Vol. 2431. Lecture Notes in Computer Science. Springer, pp. 348–360.

Mahdisoltani, Farzaneh, Joanna Biega, and Fabian M. Suchanek (2015). 'YAGO3: A Knowledge Base from Multilingual Wikipedias'. In: *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org.

Manhaeve, Robin, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt (2018). 'DeepProbLog: Neural Probabilistic Logic Programming'. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicoló Cesa-Bianchi, and Roman Garnett, pp. 3753–3763.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze (2008). *Introduction to information retrieval*. Cambridge University Press. ISBN: 978-0-521-86571-5.

McDermott, Matthew B. A., Shirly Wang, Nikki Marinsek, Rajesh Ranganath, Marzyeh Ghassemi, and Luca Foschini (2019). 'Reproducibility in Machine Learning for Health'. In: *Reproducibility in Machine Learning, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net.

McInnes, Leland, John Healy, Nathaniel Saul, and Lukas Großberger (2018). 'UMAP: Uniform Manifold Approximation and Projection'. In: *J. Open Source Software* 3.29, p. 861.

McPherson, Miller, Lynn Smith-Lovin, and James M Cook (2001). 'Birds of a feather: Homophily in social networks'. In: *Annual review of sociology* 27.1, pp. 415–444.

Meester, Ben De, Pieter Heyvaert, Dörthe Arndt, Anastasia Dimou, and Ruben Verborgh (2019). 'RDF Graph Validation Using Rule-Based Reasoning'. In: *Journal of Web Semantics*.

Menday, Roger, Luca Costabello, Jürgen Umbrich, Pierre-Yves Vandenbussche, Emir Muñoz, and Vít Nováček (Oct. 2016). *Query mediator, a method of querying a polyglot data tier and a computer program executable to carry out a method of querying a polyglot data tier*. US Patent App. 15/084,293.

Meusel, Robert, Petar Petrovski, and Christian Bizer (2014). 'The WebDataCommons Microdata, RDFa and Microformat Dataset Series'. In: *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*. Ed. by Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandecic, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz, and Carole A. Goble. Vol. 8796. Lecture Notes in Computer Science. Springer, pp. 277–292.

Mihindukulasooriya, Nandana, María Poveda-Villalón, Raúl García-Castro, and Asunción Gómez-Pérez (2015). 'Loupe - An Online Tool for Inspecting Datasets in the Linked Data Cloud'. In: *Proceedings of the ISWC 2015 Posters & Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC-2015), Bethlehem, PA, USA, October 11, 2015*. Ed. by Serena Villata, Jeff Z. Pan, and Mauro Dragoni. Vol. 1486. CEUR Workshop Proceedings. CEUR-WS.org.

Mihindukulasooriya, Nandana, Mohammad Rifat Ahmmad Rashid, Giuseppe Rizzo, Raúl García-Castro, Óscar Corcho, and Marco Torchiano (2018). 'RDF shape induction using knowledge base profiling'. In: *Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC 2018, Pau, France, April 09-13, 2018*. Ed. by Hisham M. Haddad, Roger L. Wainwright, and Richard Chbeir. ACM, pp. 1952–1959.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013). 'Efficient Estimation of Word Representations in Vector Space'. In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun.

Miller, George A. (1995). 'WordNet: A Lexical Database for English'. In: *Commun. ACM* 38.11, pp. 39–41.

Min, Bonan, Ralph Grishman, Li Wan, Chang Wang, and David Gondek (2013). 'Distant Supervision for Relation Extraction with an Incomplete Knowledge Base'. In: *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*. Ed. by Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff. The Association for Computational Linguistics, pp. 777–782.

Minervini, Pasquale, Luca Costabello, Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (2017). 'Regularizing Knowledge Graph Embeddings via Equivalence and Inversion Axioms'. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part I*. Ed. by Michelangelo Ceci, Jaakko Hollmén, Ljupco Todorovski, Celine Vens, and Saso Dzeroski. Vol. 10534. Lecture Notes in Computer Science. Springer, pp. 668–683.

Minervini, Pasquale, Luca Costabello, Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (May 2018). *Method and apparatus for completing a knowledge graph*. US Patent App. 15/821,088.

Minervini, Pasquale, Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel (2017). 'Adversarial Sets for Regularising Neural Link Predictors'. In: *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. Ed. by Gal Elidan, Kristian Kersting, and Alexander T. Ihler. AUAI Press.

Minker, Jack (1982). 'On Indefinite Databases and the Closed World Assumption'. In: *6th Conference on Automated Deduction, New York, USA, June 7-9, 1982, Proceedings*. Ed. by Donald W. Loveland. Vol. 138. Lecture Notes in Computer Science. Springer, pp. 292–308.

Mirza, Paramita, Simon Razniewski, Fariz Darari, and Gerhard Weikum (2017). 'Cardinal Virtues: Extracting Relation Cardinalities from Text'. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*. Ed. by Regina Barzilay and Min-Yen Kan. Association for Computational Linguistics, pp. 347–351.

Mitchell, Tom M. (1997). *Machine learning*. McGraw Hill series in computer science. McGraw-Hill. ISBN: 978-0-07-042807-2.

Mitchell, Tom M., William W. Cohen, Estevam R. Hruschka Jr., Partha P. Talukdar, Bo Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matt Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, et al. (2018). 'Never-ending learning'. In: *Commun. ACM* 61.5, pp. 103–115.

Mittal, Harshal, Kartikey Pandey, and Yash Kant (2019). 'ICLR Reproducibility Challenge Report (Padam : Closing The Generalization Gap Of Adaptive Gradient Methods in Training Deep Neural Networks)'. In: *CoRR* abs/1901.09517.

Mizutani, Sayaka, Edouard Pauwels, Véronique Stoven, Susumu Goto, and Yoshihiro Yamanishi (2012). 'Relating drug–protein interaction network with drug side effects'. In: *Bioinformatics* 28.18, pp. i522–i528.

Mohamed, Sameh K., Emir Muñoz, Vít Nováček, and Pierre-Yves Vandenbussche (2017). 'Identifying Equivalent Relation Paths in Knowledge Graphs'. In: *Language, Data, and Knowledge - First International Conference, LDK 2017, Galway, Ireland, June 19-20, 2017, Proceedings*. Ed. by Jorge Gracia, Francis Bond, John P. McCrae, Paul Buitelaar, Christian Chiarcos, and Sebastian Hellmann. Vol. 10318. Lecture Notes in Computer Science. Springer, pp. 299–314.

Mohamed, Sameh K., Vít Nováček, and Pierre-Yves Vandenbussche (2018). 'Knowledge base completion using distinct subgraph paths'. In: *Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC 2018, Pau, France, April 09-13, 2018*. Ed. by Hisham M. Haddad, Roger L. Wainwright, and Richard Chbeir. ACM, pp. 1992–1999.

Mohamed, Sameh K., Vít Nováček, Pierre-Yves Vandenbussche, and Emir Muñoz (2019). 'Loss Functions in Knowledge Graph Embedding Models'. In: *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG2019) Co-located with the 16th Extended Semantic Web Conference 2019 (ESWC 2019), Portoroz, Slovenia, June 2, 2019*. Ed. by Mehwish Alam, Davide Buscaldi, Michael Cochez, Francesco Osborne, Diego Reforgiato Recupero, and Harald Sack. Vol. 2377. CEUR Workshop Proceedings. CEUR-WS.org, pp. 1–10.

Motik, Boris, Ian Horrocks, and Ulrike Sattler (2009). 'Bridging the gap between OWL and relational databases'. In: *J. Web Sem.* 7.2, pp. 74–89.

Motik, Boris, Yavor Nenov, Robert Edgar Felix Piro, and Ian Horrocks (2015). 'Handling Owl: sameAs via Rewriting'. In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.* Ed. by Blai Bonet and Sven Koenig. AAAI Press, pp. 231–237.

Motik, Boris, Peter F. Patel-Schneider, and Bijan Parsia (2012). *OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition)*. https://www.w3.org/TR/owl2-syntax/.

Motik, Boris, Peter F Patel-Schneider, and Bijan Parsia (2012). *OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition)*. http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/.

Motro, Amihai (1989). 'Integrity = Validity + Completeness'. In: *ACM Trans. Database Syst.* 14.4, pp. 480–502.

Motro, Amihai (1996). 'Sources of Uncertainty, Imprecision, and Inconsistency in Information Systems'. In: *Uncertainty Management in Information Systems: From Needs to Solution*. Ed. by Amihai Motro and Philippe Smets. Kluwer Academic Publishers, Boston, pp. 9–34.

Muñoz, Emir (2014). 'Learning Content Patterns from Linked Data'. In: *Proceedings of the Second International Workshop on Linked Data for Information Extraction (LD4IE 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, October 20, 2014.* Ed. by Anna Lisa Gentile, Ziqi Zhang, Claudia d'Amato, and Heiko Paulheim. Vol. 1267. CEUR Workshop Proceedings. CEUR-WS.org, pp. 21–32.

Muñoz, Emir (2016). 'On Learnability of Constraints from RDF Data'. In: *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings*. Ed. by Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Simone Paolo Ponzetto, and Christoph Lange. Vol. 9678. Lecture Notes in Computer Science. Springer, pp. 834–844.

Muñoz, Emir, Ahmed Abdelrahman, Vít Nováček, and Pierre-Yves Vandenbussche (Nov. 2016). *Apparatus and a system for calculating similarities between drugs and using the similarities to extrapolate side effects*. US Patent App. 15/087,902.

Muñoz, Emir, Suad Aldarra, and Luca Costabello (Jan. 2017). *A method, computer program and filter for protecting a computer device against malicious RDF content in Linked Data*. EPO Patent App. EP20170153795.

Muñoz, Emir, Luca Costabello, and Pierre-Yves Vandenbussche (2014). 'μRaptor: A DOM-based System with Appetite for hCard Elements'. In: *Proceedings of the Second International Workshop on Linked Data for Information Extraction (LD4IE 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, October 20, 2014.* Ed. by Anna Lisa Gentile, Ziqi Zhang, Claudia d'Amato, and Heiko Paulheim. Vol. 1267. CEUR Workshop Proceedings. CEUR-WS.org, pp. 67–71.

Muñoz, Emir, Aidan Hogan, and Alessandra Mileo (2014). 'Using linked data to mine RDF from wikipedia's tables'. In: *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*. Ed. by Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler. ACM, pp. 533–542.

Muñoz, Emir, Pasquale Minervini, and Matthias Nickles (2019). 'Embedding cardinality constraints in neural link predictors'. In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, Limassol, Cyprus, April 8-12, 2019*. Ed. by Chih-Cheng Hung and George A. Papadopoulos. ACM, pp. 2243–2250.

Muñoz, Emir and Matthias Nickles (2017). 'Mining Cardinalities from Knowledge Bases'. In: *Database and Expert Systems Applications - 28th International Conference, DEXA 2017, Lyon, France, August 28-31, 2017, Proceedings, Part I*. Ed. by Djamal Benslimane, Ernesto Damiani, William I. Grosky, Abdelkader Hameurlain, Amit P. Sheth, and Roland R. Wagner. Vol. 10438. Lecture Notes in Computer Science. Springer, pp. 447–462.

Muñoz, Emir and Matthias Nickles (2018). 'Statistical Relation Cardinality Bounds in Knowledge Bases'. In: *T. Large-Scale Data- and Knowledge-Centered Systems* 39, pp. 67–97.

Muñoz, Emir, Vít Nováček, and Pierre-Yves Vandenbussche (2016). 'Using Drug Similarities for Discovery of Possible Adverse Reactions'. In: *AMIA 2016, American Medical Informatics Association Annual Symposium, Chicago, IL, USA, November 12-16, 2016*. 2500122[PII]. AMIA, pp. 924–933.

Muñoz, Emir, Vít Nováček, and Pierre-Yves Vandenbussche (2019). 'Facilitating prediction of adverse drug reactions by using knowledge graphs and multi-label learning models'. In: *Briefings in Bioinformatics* 20.1, pp. 190–202.

Murphy, Kevin P. (2012). *Machine learning - a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press. ISBN: 0262018020.

Nair, Vinod and Geoffrey E. Hinton (2010). 'Rectified Linear Units Improve Restricted Boltzmann Machines'. In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*. Ed. by Johannes Fürnkranz and Thorsten Joachims. Omnipress, pp. 807–814.

Neumann, Thomas and Guido Moerkotte (2011). 'Characteristic sets: Accurate cardinality estimation for RDF queries with multiple joins'. In: *Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany*. Ed. by Serge Abiteboul, Klemens Böhm, Christoph Koch, and Kian-Lee Tan. IEEE Computer Society, pp. 984–994.

Nguyen, Dat Quoc, Kairit Sirts, Lizhen Qu, and Mark Johnson (2016). 'Neighborhood Mixture Model for Knowledge Base Completion'. In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*. Ed. by Yoav Goldberg and Stefan Riezler. ACL, pp. 40–50.

Nguyen, Tin A., Walton A. Perkins, Thomas J. Laffey, and Deanne Pecora (1985). 'Checking an Expert Systems Knowledge Base for Consistency and Completeness'. In: *Proceedings of the 9th International Joint Conference on Artificial Intelligence. Los Angeles, CA, USA, August 1985*. Ed. by Aravind K. Joshi. Morgan Kaufmann, pp. 375–378.

Nickel, Maximilian and Douwe Kiela (2017). 'Poincaré Embeddings for Learning Hierarchical Representations'. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, pp. 6341–6350.

Nickel, Maximilian, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich (2016). 'A Review of Relational Machine Learning for Knowledge Graphs'. In: *Proceedings of the IEEE* 104.1, pp. 11–33.

Nickel, Maximilian, Lorenzo Rosasco, and Tomaso A. Poggio (2016). 'Holographic Embeddings of Knowledge Graphs'. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.* Ed. by Dale Schuurmans and Michael P. Wellman. AAAI Press, pp. 1955–1961.

Nickel, Maximilian, Volker Tresp, and Hans-Peter Kriegel (2011). 'A Three-Way Model for Collective Learning on Multi-Relational Data'. In: *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*. Ed. by Lise Getoor and Tobias Scheffer. Omnipress, pp. 809–816.

Nickel, Maximilian, Volker Tresp, and Hans-Peter Kriegel (2012). 'Factorizing YAGO: scalable machine learning for linked data'. In: *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*. Ed. by Alain Mille, Fabien L. Gandon, Jacques Misselis, Michael Rabinovich, and Steffen Staab. ACM, pp. 271–280.

Nilsson, Mikael (2008). *Description Set Profiles: A constraint language for Dublin Core Application Profiles*. https://www.dublincore.org/specifications/dublin-core/dc-dsp/.

Noy, Natalya Fridman, Michael Sintek, Stefan Decker, Monica Crubézy, Ray W. Fergerson, and Mark A. Musen (2001). 'Creating Semantic Web Contents with Protégé-2000'. In: *IEEE Intelligent Systems* 16.2, pp. 60–71.

Olivé, Antoni (2007). *Conceptual modeling of information systems*. Springer.

Papakonstantinou, Vassilis, Giorgos Flouris, Irini Fundulaki, and Andrey Gubichev (2016). 'Some Thoughts on OWL-Empowered SPARQL Query Optimization'. In: *The Semantic Web - ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*. Ed. by Harald Sack, Giuseppe Rizzo, Nadine Steinmetz, Dunja Mladenic, Sören Auer, and Christoph Lange. Vol. 9989. Lecture Notes in Computer Science, pp. 12–16.

Patel-Schneider, Peter F. (2015). 'Using Description Logics for RDF Constraint Checking and Closed-World Recognition'. In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. Ed. by Blai Bonet and Sven Koenig. AAAI Press, pp. 247–253.

Paulheim, Heiko (2017). 'Knowledge graph refinement: A survey of approaches and evaluation methods'. In: *Semantic Web* 8.3, pp. 489–508.

Paulheim, Heiko and Christian Bizer (2014). 'Improving the Quality of Linked Data Using Statistical Distributions'. In: *Int. J. Semantic Web Inf. Syst.* 10.2, pp. 63–86.

Pauwels, Edouard, Véronique Stoven, and Yoshihiro Yamanishi (2011). 'Predicting drug side-effect profiles: a chemical fragment-based approach'. In: *BMC bioinformatics* 12.1, p. 169.

Pearson, Ronald K. (2005). *Mining imperfect data - dealing with contamination and incomplete records*. SIAM. ISBN: 978-0-89871-582-8.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, et al. (Nov. 2011). 'Scikit-learn: Machine Learning in Python'. In: *J. Mach. Learn. Res.* 12, pp. 2825–2830. ISSN: 1532-4435.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). 'Glove: Global Vectors for Word Representation'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. ACL, pp. 1532–1543.

Pham, Minh-Duc and Peter A. Boncz (2016). 'Exploiting Emergent Schemas to Make RDF Systems More Efficient'. In: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*. Ed. by Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil. Vol. 9981. Lecture Notes in Computer Science, pp. 463–479.

Pietriga, Emmanuel, Hande Gözükan, Caroline Appert, Marie Destandau, Šejla Čebirić, François Goasdoué, and Ioana Manolescu (2018). 'Browsing Linked Data Catalogs with LODAtlas'. In: *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part II*. Ed. by Denny Vrandecic, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee, and Elena Simperl. Vol. 11137. Lecture Notes in Computer Science. Springer, pp. 137–153.

Polikar, R. (Mar. 2006). 'Ensemble based systems in decision making'. In: *IEEE Circuits and Systems Magazine* 6.3, pp. 21–45. ISSN: 1531-636X.

Polleres, Axel, Juan L. Reutter, and Egor V. Kostylev (2016). 'Nested Constructs vs. Sub-Selects in SPARQL'. In: *Proceedings of the 10th Alberto Mendelzon International Workshop on Foundations of Data Management, Panama City, Panama, May 8-10, 2016*. Ed. by Reinhard Pichler and Altigran Soares da Silva. Vol. 1644. CEUR Workshop Proceedings. CEUR-WS.org.

Polleres, Axel, François Scharffe, and Roman Schindlauer (2007). 'SPARQL++ for Mapping Between RDF Vocabularies'. In: *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS, OTM Confederated International Conferences CoopIS, DOA, ODBASE, GADA, and IS 2007, Vilamoura, Portugal, November 25-30, 2007, Proceedings, Part I*. Ed. by Robert Meersman and Zahir Tari. Vol. 4803. Lecture Notes in Computer Science. Springer, pp. 878–896.

Prud'hommeaux, Eric and Thomas Baker (2019). *ShapeMap Structure and Language*. https://shexspec.github.io/shape-map/.

Prud'hommeaux, Eric, Iovka Boneva, José Emilio Labra Gayo, and Gregg Kellogg (2018). *Shape Expressions Language 2.1*. http://shex.io/shex-semantics/.

Prud'hommeaux, Eric, José Emilio Labra Gayo, and Harold R. Solbrig (2014). 'Shape expressions: an RDF validation and transformation language'. In: *Proceedings of the 10th International Conference on Semantic Systems, SEMANTICS 2014, Leipzig, Germany, September 4-5, 2014*. Ed. by Harald Sack, Agata Filipowska, Jens Lehmann, and Sebastian Hellmann. ACM, pp. 32–40.

Prud'hommeaux, Eric and Andy Seaborne (2008). *SPARQL Query Language for RDF*. https://www.w3.org/TR/rdf-sparql-query/.

Qu, Meng and Jian Tang (2019). 'Probabilistic Logic Neural Networks for Reasoning'. In: *CoRR* abs/1906.08495.

Quinlan, J. Ross (1990). 'Learning Logical Definitions from Relations'. In: *Machine Learning* 5, pp. 239–266.

Rahmani, Hossein, Gerhard Weiss, Oscar Méndez-Lucio, and Andreas Bender (2016). 'ARWAR: A network approach for predicting Adverse Drug Reactions'. In: *Computers in biology and medicine* 68, pp. 101–108.

Ram, Ashwin, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, Eric King, Kate Bland, et al. (2018). 'Conversational AI: The Science Behind the Alexa Prize'. In: *CoRR* abs/1801.03604.

Razniewski, Simon, Flip Korn, Werner Nutt, and Divesh Srivastava (2015). 'Identifying the Extent of Completeness of Query Answers over Partially Complete Databases'. In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*. Ed. by Timos K. Sellis, Susan B. Davidson, and Zachary G. Ives. ACM, pp. 561–576.

Razniewski, Simon and Werner Nutt (2014). 'Databases under the Partial Closed-world Assumption: A Survey'. In: *Proceedings of the 26th GI-Workshop Grundlagen von Datenbanken, Bozen-Bolzano, Italy, October 21st to 24th, 2014*. Ed. by Friederike Klan, Günther Specht, and Hans Gamper. Vol. 1313. CEUR Workshop Proceedings. CEUR-WS.org, pp. 59–64.

Razniewski, Simon, Ognjen Savkovic, and Werner Nutt (2016). 'Turning The Partial-Closed World Assumption Upside Down'. In: *Proceedings of the 10th Alberto Mendelzon International Workshop on Foundations of Data Management, Panama City, Panama, May 8-10, 2016*. Ed. by Reinhard Pichler and Altigran Soares da Silva. Vol. 1644. CEUR Workshop Proceedings. CEUR-WS.org.

Razniewski, Simon, Fabian M. Suchanek, and Werner Nutt (2016). 'But What Do We Actually Know?' In: *Proceedings of the 5th Workshop on Automated Knowledge Base Construction, AKBC@NAACL-HLT 2016, San Diego, CA, USA, June 17, 2016*. Ed. by Jay Pujara, Tim Rocktäschel, Danqi Chen, and Sameer Singh. The Association for Computer Linguistics, pp. 40–44.

Reiter, Raymond (1977). 'On Closed World Data Bases'. In: *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, 1977*. Ed. by Hervé Gallaire and Jack Minker. Advances in Data Base Theory. New York: Plemum Press, pp. 55–76.

Reiter, Raymond (1986). 'Foundations for Knowledge-Based Systems (Invited Paper)'. In: *Information Processing 86, Proceedings of the IFIP 10th World Computer Congress, Dublin, Ireland, September 1-5, 1986*. Ed. by Hans-Jürgen Kugler. North-Holland/IFIP, pp. 663–668.

Rendle, Steffen (2013). 'Scaling Factorization Machines to Relational Data'. In: *PVLDB* 6.5, pp. 337–348.

Rendle, Steffen, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme (2009). 'BPR: Bayesian Personalized Ranking from Implicit Feedback'. In: *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*. Ed. by Jeff A. Bilmes and Andrew Y. Ng. AUAI Press, pp. 452–461.

Rettinger, Achim, Uta Lösch, Volker Tresp, Claudia d'Amato, and Nicola Fanizzi (2012). 'Mining the Semantic Web - Statistical learning for next generation knowledge bases'. In: *Data Min. Knowl. Discov.* 24.3, pp. 613–662.

Ribeiro, Marco Túlio, Sameer Singh, and Carlos Guestrin (2016). '"Why Should I Trust You?": Explaining the Predictions of Any Classifier'. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. Ed. by Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi. ACM, pp. 1135–1144.

Richardson, Matthew and Pedro M. Domingos (2006). 'Markov logic networks'. In: *Machine Learning* 62.1-2, pp. 107–136.

Rietveld, Laurens, Wouter Beek, Rinke Hoekstra, and Stefan Schlobach (2017). 'Meta-data for a lot of LOD'. In: *Semantic Web* 8.6, pp. 1067–1080.

Ristoski, Petar and Heiko Paulheim (2014). 'A Comparison of Propositionalization Strategies for Creating Features from Linked Open Data'. In: *Proceedings of the 1st Workshop on Linked Data for Knowledge Discovery co-located with European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2014), Nancy, France, September 19th, 2014*. Ed. by Ilaria Tiddi, Mathieu d'Aquin, and Nicolas Jay. Vol. 1232. CEUR Workshop Proceedings. CEUR-WS.org.

Ristoski, Petar and Heiko Paulheim (2016a). 'RDF2Vec: RDF Graph Embeddings for Data Mining'. In: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*. Ed. by Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil. Vol. 9981. Lecture Notes in Computer Science, pp. 498–514.

Ristoski, Petar and Heiko Paulheim (2016b). 'Semantic Web in data mining and knowledge discovery: A comprehensive survey'. In: *J. Web Semant.* 36, pp. 1–22.

Rivero, Carlos R., Inma Hernández, David Ruiz, and Rafael Corchuelo (2012). 'Towards Discovering Ontological Models from Big RDF Data'. In: *Advances in Conceptual Modeling - ER 2012 Workshops CMS, ECDM-NoCoDA, MoDIC, MORE-BI, RIGiM, SeCoGIS, WISM, Florence, Italy, October 15-18, 2012. Proceedings*. Ed. by Silvana Castano, Panos Vassiliadis, Laks V. S. Lakshmanan, and Mong-Li Lee. Vol. 7518. Lecture Notes in Computer Science. Springer, pp. 131–140.

Robbins, Herbert and Sutton Monro (Sept. 1951). 'A Stochastic Approximation Method'. In: *Ann. Math. Statist.* 22.3, pp. 400–407.

Rocktäschel, Tim (2018). 'Combining representation learning with logic for language processing'. PhD thesis. University College London, UK.

Rocktäschel, Tim, Sameer Singh, and Sebastian Riedel (2015). 'Injecting Logical Background Knowledge into Embeddings for Relation Extraction'. In: *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*. Ed. by Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar. The Association for Computational Linguistics, pp. 1119–1129.

Rosasco, Lorenzo, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri (2004). 'Are Loss Functions All the Same?' In: *Neural Computation* 16.5, pp. 1063–107.

Rosner, Bernard (1983). 'Percentage Points for a Generalized ESD Many-Outlier Procedure'. In: *Technometrics* 25.2, pp. 165–172.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986). 'Learning representations by back-propagating errors'. In: *Nature* 323.6088, pp. 533–536. ISSN: 1476-4687.

Russell, Stuart J. and Peter Norvig (2010). *Artificial Intelligence - A Modern Approach (3. internat. ed.)* Pearson Education. ISBN: 978-0-13-207148-2.

Ryman, Arthur G., Arnaud Le Hors, and Steve Speicher (2013). 'OSLC Resource Shape: A language for defining constraints on Linked Data'. In: *Proceedings of the WWW2013 Workshop on Linked Data on the Web, Rio de Janeiro, Brazil, 14 May, 2013*. Ed. by Christian Bizer, Tom Heath, Tim Berners-Lee, Michael Hausenblas, and Sören Auer. Vol. 996. CEUR Workshop Proceedings. CEUR-WS.org.

Sarjant, Samuel, Catherine Legg, Michael Robinson, and Olena Medelyan (2009). '"All You Can Eat" Ontology-Building: Feeding Wikipedia to Cyc'. In: *2009 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2009, Milan, Italy, 15-18 September 2009, Main Conference Proceedings*. IEEE Computer Society, pp. 341–348.

Schenner, Gottfried, Stefan Bischof, Axel Polleres, and Simon Steyskal (2014). 'Integrating Distributed Configurations With RDFS and SPARQL'. In: *Proceedings of the 16th International Configuration Workshop, Novi Sad, Serbia, September 25-26, 2014.* Ed. by Alexander Felfernig, Cipriano Forza, and Albert Haag. Vol. 1220. CEUR Workshop Proceedings. CEUR-WS.org, pp. 9–15.

Schmidhuber, Jürgen (2015). 'Deep learning in neural networks: An overview'. In: *Neural Networks* 61, pp. 85–117.

Schmidt, Michael and Georg Lausen (2013). 'Pleasantly Consuming Linked Data with RDF Data Descriptions'. In: *Proceedings of the Fourth International Workshop on Consuming Linked Data, COLD 2013, Sydney, Australia, October 22, 2013*. Ed. by Olaf Hartig, Juan F. Sequeda, Aidan Hogan, and Takahide Matsutsuka. Vol. 1034. CEUR Workshop Proceedings. CEUR-WS.org.

Schmidt, Michael, Michael Meier, and Georg Lausen (2010). 'Foundations of SPARQL query optimization'. In: *Database Theory - ICDT 2010, 13th International Conference, Lausanne, Switzerland, March 23-25, 2010, Proceedings*. Ed. by Luc Segoufin. ACM International Conference Proceeding Series. ACM, pp. 4–33.

Sengupta, Kunal, Adila Alfa Krisnadhi, and Pascal Hitzler (2011). 'Local Closed World Semantics: Grounded Circumscription for OWL'. In: *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*. Ed. by Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Fridman Noy, and Eva Blomqvist. Vol. 7031. Lecture Notes in Computer Science. Springer, pp. 617–632.

Shervashidze, Nino, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt (2011). 'Weisfeiler-Lehman Graph Kernels'. In: *J. Mach. Learn. Res.* 12, pp. 2539–2561.

Shi, Chuan, Yitong Li, Jiawei Zhang, Yizhou Sun, and Philip S. Yu (2017). 'A Survey of Heterogeneous Information Network Analysis'. In: *IEEE Trans. Knowl. Data Eng.* 29.1, pp. 17–37.

Singh, Ajit Paul and Geoffrey J. Gordon (2008). 'Relational learning via collective matrix factorization'. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*. Ed. by Ying Li, Bing Liu, and Sunita Sarawagi. ACM, pp. 650–658.

Socher, Richard, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng (2013). 'Reasoning With Neural Tensor Networks for Knowledge Base Completion'. In: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.* Ed. by Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, pp. 926–934.

Song, Qi, Yinghui Wu, Peng Lin, Xin Dong, and Hui Sun (2018). 'Mining Summaries for Knowledge Graph Search'. In: *IEEE Trans. Knowl. Data Eng.* 30.10, pp. 1887–1900.

Soulet, Arnaud and Fabian M. Suchanek (2019). 'Anytime Large-Scale Analytics of Linked Open Data'. In: *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I*. Ed. by Chiara Ghidini, Olaf Hartig, Maria Maleshkova, Vojtech Svátek, Isabel F. Cruz, Aidan Hogan, Jie Song, Maxime Lefrançois, and Fabien Gandon. Vol. 11778. Lecture Notes in Computer Science. Springer, pp. 576–592.

Spahiu, Blerina, Andrea Maurino, and Matteo Palmonari (2018). 'Towards Improving the Quality of Knowledge Graphs with Data-driven Ontology Patterns and SHACL'. In: *Proceedings of the 9th Workshop on Ontology Design and Patterns (WOP 2018) co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 9th, 2018*. Ed. by Martin G. Skjæveland, Yingjie Hu, Karl Hammar, Vojtech Svátek, and Agnieszka Lawrynowicz. Vol. 2195. CEUR Workshop Proceedings. CEUR-WS.org, pp. 52–66.

Spahiu, Blerina, Riccardo Porrini, Matteo Palmonari, Anisa Rula, and Andrea Maurino (2016). 'ABSTAT: Ontology-driven Linked Data Summaries with Pattern Minimalization'. In: *Proceedings of the 2nd International Workshop on Summarizing and Presenting Entities and Ontologies (SumPre 2016) co-located with the 13th Extended Semantic Web Conference (ESWC 2016), Anissaras, Greece, May 30, 2016*. Ed. by Andreas Thalhammer, Gong Cheng, and Kalpa Gunaratna. Vol. 1605. CEUR Workshop Proceedings. CEUR-WS.org.

Studer, Rudi, V. Richard Benjamins, and Dieter Fensel (1998). 'Knowledge Engineering: Principles and Methods'. In: *Data Knowl. Eng.* 25.1-2, pp. 161–197.

Suchanek, Fabian M., David Gross-Amblard, and Serge Abiteboul (2011). 'Watermarking for Ontologies'. In: *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*. Ed. by Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Fridman Noy, and Eva Blomqvist. Vol. 7031. Lecture Notes in Computer Science. Springer, pp. 697–713.

Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum (2007). 'Yago: a core of semantic knowledge'. In: *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. Ed. by Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy. ACM, pp. 697–706.

Sun, Yizhou and Jiawei Han (2012). 'Mining heterogeneous information networks: a structural analysis approach'. In: *SIGKDD Explorations* 14.2, pp. 20–28.

Szarfman, Ana, Stella G Machado, and Robert T O'neill (2002). 'Use of screening algorithms and computer systems to efficiently signal higher-than-expected combinations of drugs and events in the US FDA's spontaneous reports database'. In: *Drug Safety* 25.6, pp. 381–392.

Tan, Yuxiang, Yong Hu, Xiaoxiao Liu, Zhinan Yin, Xue-wen Chen, and Mei Liu (2016). 'Improving drug safety: From adverse drug reaction knowledge discovery to clinical implementation'. In: *Methods*.

Tandon, Niket, Aparna S. Varde, and Gerard de Melo (2017). 'Commonsense Knowledge in Machine Intelligence'. In: *SIGMOD Record* 46.4, pp. 49–52.

Tanon, Thomas Pellissier, Denny Vrandecic, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher (2016). 'From Freebase to Wikidata: The Great Migration'. In: *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. Ed. by Jacqueline Bourdeau, Jim Hendler, Roger Nkambou, Ian Horrocks, and Ben Y. Zhao. ACM, pp. 1419–1428.

Tao, Jiao, Evren Sirin, Jie Bao, and Deborah L. McGuinness (2010). 'Integrity Constraints in OWL'. In: *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. Ed. by Maria Fox and David Poole. AAAI Press.

Tekli, Joe, Richard Chbeir, Agma J. M. Traina, Caetano Traina Jr., and Renato Fileto (2015). 'Approximate XML structure validation based on document-grammar tree similarity'. In: *Inf. Sci.* 295, pp. 258–302.

Thalheim, Bernhard (1992). 'Fundamentals of Cardinality Constraints'. In: *Entity-Relationship Approach - ER 92, 11th International Conference on the Entity-Relationship Approach, Karlsruhe, Germany, October 7-9, 1992, Proceedings*. Ed. by Günther Pernul and A Min Tjoa. Vol. 645. Lecture Notes in Computer Science. Springer, pp. 7–23.

Toutanova, Kristina and Danqi Chen (July 2015). 'Observed versus latent features for knowledge base and text inference'. In: *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*. Beijing, China: Association for Computational Linguistics, pp. 57–66.

Trouillon, Théo, Christopher R. Dance, Éric Gaussier, Johannes Welbl, Sebastian Riedel, and Guillaume Bouchard (2017). 'Knowledge Graph Completion via Complex Tensor Factorization'. In: *Journal of Machine Learning Research* 18, 130:1–130:38.

Trouillon, Théo and Maximilian Nickel (2017). 'Complex and Holographic Embeddings of Knowledge Graphs: A Comparison'. In: *CoRR* abs/1707.01475.

Trouillon, Théo, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard (2016). 'Complex Embeddings for Simple Link Prediction'. In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. Ed. by Maria-Florina Balcan and Kilian Q. Weinberger. Vol. 48. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 2071–2080.

Tsoumakas, Grigorios and Ioannis Katakis (2006). 'Multi-label classification: An overview'. In: *International Journal of Data Warehousing and Mining* 3.3.

Tsoumakas, Grigorios, Ioannis Katakis, and Ioannis P. Vlahavas (2010). 'Mining Multi-label Data'. In: *Data Mining and Knowledge Discovery Handbook*. Springer, pp. 667–685.

Tucker, Ledyard R. (Sept. 1966). 'Some mathematical notes on three-mode factor analysis'. In: *Psychometrika* 31.3, pp. 279–311. ISSN: 1860-0980.

Vandenbussche, Pierre-Yves, Ghislain Atemezing, María Poveda-Villalón, and Bernard Vatant (2017). 'Linked Open Vocabularies (LOV): A gateway to reusable semantic vocabularies on the Web'. In: *Semantic Web* 8.3, pp. 437–452.

Völker, Johanna and Mathias Niepert (2011). 'Statistical Schema Induction'. In: *The Semantic Web: Research and Applications - 8th Extended Semantic Web Conference, ESWC 2011, Heraklion, Crete, Greece, May 29-June 2, 2011, Proceedings, Part I*. Ed. by Grigoris Antoniou, Marko Grobelnik, Elena Paslaru Bontas Simperl, Bijan Parsia, Dimitris Plexousakis, Pieter De Leenheer, and Jeff Z. Pan. Vol. 6643. Lecture Notes in Computer Science. Springer, pp. 124–138.

Vrandecic, Denny (2012). 'Wikidata: a new platform for collaborative data collection'. In: *Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume)*. Ed. by Alain Mille, Fabien L. Gandon, Jacques Misselis, Michael Rabinovich, and Steffen Staab. ACM, pp. 1063–1064.

Vrandecic, Denny and Markus Krötzsch (2014). 'Wikidata: a free collaborative knowledgebase'. In: *Commun. ACM* 57.10, pp. 78–85.

Vries, Gerben Klaas Dirk de and Steven de Rooij (2015). 'Substructure counting graph kernels for machine learning from RDF data'. In: *J. Web Semant.* 35, pp. 71–84.

Wang, Quan, Pingping Huang, Haifeng Wang, Songtai Dai, Wenbin Jiang, Jing Liu, Yajuan Lyu, Yong Zhu, and Hua Wu (2019). 'CoKE: Contextualized Knowledge Graph Embedding'. In: *CoRR* abs/1911.02168.

Wang, Quan, Jing Liu, Yuanfei Luo, Bin Wang, and Chin-Yew Lin (2016). 'Knowledge Base Completion via Coupled Path Ranking'. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Wang, Quan, Zhendong Mao, Bin Wang, and Li Guo (2017). 'Knowledge Graph Embedding: A Survey of Approaches and Applications'. In: *IEEE Trans. Knowl. Data Eng.* 29.12, pp. 2724–2743.

Wang, Quan, Bin Wang, and Li Guo (2015). 'Knowledge Base Completion Using Embeddings and Rules'. In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, pp. 1859–1866.

Wang, Richard Y. and Diane M. Strong (1996). 'Beyond Accuracy: What Data Quality Means to Data Consumers'. In: *J. of Management Information Systems* 12.4, pp. 5–33.

Wang, Zhen, Jianwen Zhang, Jianlin Feng, and Zheng Chen (2014a). 'Knowledge Graph and Text Jointly Embedding'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. ACL, pp. 1591–1601.

Wang, Zhen, Jianwen Zhang, Jianlin Feng, and Zheng Chen (2014b). 'Knowledge Graph Embedding by Translating on Hyperplanes'. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*. Ed. by Carla E. Brodley and Peter Stone. AAAI Press, pp. 1112–1119.

Williamson, David P. and David B. Shmoys (2011). *The Design of Approximation Algorithms*. Cambridge University Press. ISBN: 978-0-521-19527-0.

Wolpert, David H. (1996). 'The Lack of A Priori Distinctions Between Learning Algorithms'. In: *Neural Computation* 8.7, pp. 1341–1390.

Wolpert, David H. and William G. Macready (1997). 'No free lunch theorems for optimization'. In: *IEEE Trans. Evolutionary Computation* 1.1, pp. 67–82.

Wynn, Karen (Aug. 1990). 'Childrens understanding of counting'. In: *Cognition* 36.2, pp. 155–193. ISSN: 0010-0277.

Yamanishi, Yoshihiro, Edouard Pauwels, and Masaaki Kotera (2012). 'Drug side-effect prediction based on the integration of chemical and biological spaces'. In: *Journal of chemical information and modeling* 52.12, pp. 3284–3292.

Yang, Bishan, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng (2015). 'Embedding Entities and Relations for Learning and Inference in Knowledge Bases'. In: *International Conference on Learning Representations*.

Yang, Runtao, Chengjin Zhang, Rui Gao, and Lina Zhang (Feb. 2015). 'An Ensemble Method with Hybrid Features to Identify Extracellular Matrix Proteins'. In: *PLOS ONE* 10.2, pp. 1–21.

Yu, Xiao, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han (2014). 'Personalized entity recommendation: a heterogeneous information network approach'. In: *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*. Ed. by Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler. ACM, pp. 283–292.

Yumusak, Semih, Emir Muñoz, Pasquale Minervini, Erdogan Dogdu, and Halife Kodaz (2016). 'A Hybrid Method for Rating Prediction Using Linked Data Features and Text Reviews'. In: *Joint Proceedings of the 5th Workshop on Data Mining and Knowledge Discovery meets Linked Open Data and the 1st International Workshop on Completing and Debugging the Semantic Web (Know@LOD-2016, CoDeS-2016) co-located with 13th ESWC 2016, Heraklion, Greece, May 30th, 2016.* Ed. by Heiko Paulheim, Jens Lehmann, Vojtech Svátek, Craig A. Knoblock, Matthew Horridge, Patrick Lambrix, and Bijan Parsia. Vol. 1586. CEUR Workshop Proceedings. CEUR-WS.org.

Zha, Zheng-Jun, Tao Mei, Jingdong Wang, Zengfu Wang, and Xian-Sheng Hua (2009). 'Graph-based semi-supervised learning with multiple labels'. In: *Journal of Visual Communication and Image Representation* 20.2. Special issue on Emerging Techniques for Multimedia Content Sharing, Search and Understanding, pp. 97–103. ISSN: 1047-3203.

Zhang, Fuzheng, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma (2016). 'Collaborative Knowledge Base Embedding for Recommender Systems'. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. Ed. by Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi. ACM, pp. 353–362.

Zhang, Jiawei, Jianhui Chen, Junxing Zhu, Yi Chang, and Philip S. Yu (2017). 'Link Prediction with Cardinality Constraint'. In: *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. Ed. by Maarten de Rijke, Milad Shokouhi, Andrew Tomkins, and Min Zhang. ACM, pp. 121–130.

Zhang, Min-Ling and Zhi-Hua Zhou (Oct. 2006). 'Multilabel Neural Networks with Applications to Functional Genomics and Text Categorization'. In: *IEEE Transactions on Knowledge and Data Engineering* 18.10, pp. 1338–1351. ISSN: 1041-4347.

Zhang, Min-Ling and Zhi-Hua Zhou (2014). 'A Review on Multi-Label Learning Algorithms'. In: *IEEE Trans. Knowl. Data Eng.* 26.8, pp. 1819–1837.

Zhang, Min-Ling and Zhi-Hua Zhou (Aug. 2014). 'A Review on Multi-Label Learning Algorithms'. In: *IEEE Transactions on Knowledge and Data Engineering* 26.8, pp. 1819–1837. ISSN: 1041-4347.

Zhang, Wen, Yanlin Chen, Shikui Tu, Feng Liu, and Qianlong Qu (Dec. 2016). 'Drug side effect prediction through linear neighborhoods and multiple data source integration'. In: *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 427–434.

Zhang, Wen, Feng Liu, Longqiang Luo, and Jingxia Zhang (2015). 'Predicting drug side effects by multi-label learning and ensemble learning'. In: *BMC bioinformatics* 16.1, p. 1.

Zhang, Wen, Hua Zou, Longqiang Luo, Qianchao Liu, Weijian Wu, and Wenyi Xiao (2016). 'Predicting potential side effects of drugs by recommender methods and ensemble learning'. In: *Neurocomputing* 173, pp. 979–987.

Zhao, Yu, Huali Feng, and Patrick Gallinari (Nov. 2019). 'Embedding Learning with Triple Trustiness on Noisy Knowledge Graph'. In: *Entropy* 21.11, p. 1083. ISSN: 1099-4300.

Zilke, Jan Ruben, Eneldo Loza Mencía, and Frederik Janssen (2016). 'DeepRED - Rule Extraction from Deep Neural Networks'. In: *Discovery Science - 19th International Conference, DS 2016, Bari, Italy,*

October 19-21, 2016, Proceedings. Ed. by Toon Calders, Michelangelo Ceci, and Donato Malerba. Vol. 9956. Lecture Notes in Computer Science, pp. 457–473.

Zoph, Barret and Quoc V. Le (2017). 'Neural Architecture Search with Reinforcement Learning'. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.