Facilitating Knowledge Graph Analysis

– Acquisition and Large-Scale Analysis of Topological Graph Measures

Inaugural-Dissertation

zur

Erlangung des Doktorgrades der
Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Leschek Matthäus Zloch

aus Deutsch Piekar, Polen

Düsseldorf, November 2020

aus dem Institut für Informatik der
Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Referent: Prof. Dr. Stefan Conrad

Koreferent: Prof. Dr. Stefan Dietze

Tag der mündlichen Prüfung: 26. Januar 2021

Ich versichere an Eides Statt, dass die Dissertation von mir selbstständig und ohne unzulässige fremde Hilfe unter Beachtung der *Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine-Universität Düsseldorf* erstellt worden ist.

Die hier vorgelegte Dissertation habe ich eigenständig und ohne unerlaubte Hilfe angefertigt. Die Dissertation wurde in der vorgelegten oder in ähnlicher Form noch bei keiner anderen Institution eingereicht. Ich habe bisher keine erfolglosen Promotionsversuche unternommen.

Düsseldorf, Deutschland
26. November 2020

Leschek Matthäus Zloch

Dedicated to my family and friends

Acknowledgements

This work is the result of my research activities in the Department of Knowledge Technologies for the Social Sciences (WTS) at GESIS – Leibniz Institute for the Social Sciences in Cologne. First, I thank the managers who recognized my skills and supported my diverse research activities by giving me several contract extensions.

I want to express my sincere thanks to my two supervisors, *Stefan Conrad* and *Stefan Dietze*. I thank Stefan Conrad for his constant support of my research ideas, the freedom of choice he gave me, and his calm confidence that my research would lead to useful results. I always left our meetings encouraged and motivated. I also thank Stefan Dietze for his frank words, his genuine criticism, and his interest in my research. Stefan gives excellent and untiring suggestions for improvements as well as prompt feedback, for which I am very grateful. He pushed my research to an upper level.

Speaking of supervisors: I owe a special thanks to *Daniel Hienert* for his motivation to publish our first results. Without this mentoring and guidance, especially at the beginning of my research activities, I probably would not have followed up on this topic. Big thanks also go to *Maribel Acosta* for her assessment of the usefulness of my ideas, her openness, and her co-authorships, which motivated me a lot.

I am very grateful for the working environment I had during my Ph.D. studies, with a lot of nice people, some of whom have become good friends over the years: *Zeljko*, *Dimi* "Die Tulpe", *Johann* "Wanja", *Masoud*, *Felix*, *Kata*, *Alex*, *Tobias*, *Sascha*, and former colleagues who have left GESIS. Thank you all for the great times we have spent together. Because of you, I always looked forward to coming to the office and using our couch there, to having fun, talking, laughing, sitting, drinking, eating pizza, watching movies, or tinkering with the next doctoral hat.

I thank my other close friends, *Oliver*, *Duy*, *Mirjam*, *Christian*, and *Klaus*, for being there all those years. You gave me some nice time off once in a while with activities like camping, hiking, photography, sailing, and making Banh Baos and sandwiches, so that I could (almost) forget about work and research. The fact that I sometimes had to turn down your warm invitations in favor of doing research and writing the thesis was hard.

I want to express my deep gratitude to my parents for their love and their demanding

and stressful hard work that gave us children the opportunity to find new ways to learn and live. I thank my sister, *Anna*, for her continuous encouragement and for cheering me up with my subscription to her music recommendations. I cannot get "Hold on to the vision" out of my head.

I owe *Maren*, my partner, a very special thanks. She supported the idea of doing a Ph.D. from the first minute, giving me mental support and understanding during stressful times and a place to go when things went south, which happened more than once.

Last but not least, from the bottom of my heart, I thank my grandfather "Dziadek Władek", who was able to open my eyes to the endless fascinations in this world with his unbelievably loving, unobtrusive, and philosophical manner. This is for you.

Abstract

In today's Web, the most common model for structuring knowledge and making it machine-readable is the *knowledge graph*. In this model, vertices represent Web entities that encode real-world objects as URIs (Uniform Resource Identifiers); edges are labeled, and represent relationships between these entities, which are modeled by knowledge-domain-specific vocabularies and predefined schemas.

The topology of knowledge graphs differs fundamentally from other topologies, for example those of computer networks or social graphs. This is because, first, knowledge graphs contain hierarchical (typed) as well as transversal relationships between vertices. Second, the shape of the graph topology is significantly influenced, on the one hand, by knowledge-domain-specific vocabulary usage defined by particular schemas and, on the other hand, by the inconsistent modeling habits of researchers and modeling tools. Analyzing and understanding the distinct topology, and employing meaningful measures for the appropriate characterization of knowledge graphs is crucial, and can guide and inform the development of, for example, profiling tools, benchmarking solutions, efficient data structures and indexes, and compression techniques. Traditional measures known from network science inadequately capture the semantics that knowledge graph topologies entail. Therefore, it is of central importance to provide appropriate tools for the analysis, and proper measures for the characterization of knowledge graphs.

The present cumulative dissertation is motivated by this. It makes three scientific contributions, each of which constitutes one part of the thesis.

The first part of the thesis introduces and describes a software framework that consolidates third-party tools for the acquisition and preparation of knowledge graphs in order to enable graph-related tasks on their topology. We perform a large-scale analysis of 280 knowledge graphs from nine knowledge domains provided by the Linked Open Data (LOD) Cloud, and we calculate 54 different graph measures with this tool. The analysis results and the processed graph objects are available to the research community for further processing.

Building on this, the second part of the thesis deals with the investigation of commonly used measures from network analysis as well as measures that have been specially introduced for the characterization of RDF knowledge graphs. We examine

them in terms of their relevance and meaningfulness for generating concise descriptions of knowledge graph topologies. In particular, we seek to find measures that have the capacity to discriminate graphs from other knowledge domains in order to reveal knowledge domain specificities and derive corresponding implications for existing solutions. To this end, we employ various statistical methods and a state-of-the-art machine learning classification model.

In the third and final part of this thesis, we employ our framework introduced earlier to propose solutions in other research areas of knowledge graphs. We deal with database benchmarks for knowledge graphs and address the criticism that RDF benchmarks deliver less reliable results due to the usage of synthetic queries for runtime measurements. To this end, we propose a functionality of our framework to leverage programmatic graph representations from knowledge graphs to generate application-specific queries based on real-world data. Furthermore, we present a flexible "business use case"-driven approach, which allows to assess response times of database queries more reliably by means of building query groups.

This thesis is based on published papers submitted to high-ranked international peer-reviewed open access journals, international conferences, and workshops in the research area of Semantic Web technologies. As a commitment to open science, all code and resources have been published as open source projects under MIT license on popular code and data hosting platforms.

Zusammenfassung

Im heutigen Web ist der *Wissensgraph* (engl. *knowledge graph*) das gebräuchlichste Modell, um Wissen zu strukturieren und maschinenlesbar zu machen. In diesem Modell sind Knoten typisiert und repräsentieren Objekte der realen Welt, die als Web-Entitäten kodiert sind; Kanten sind bezeichnet und stellen Beziehungen zwischen den Knoten dar, die mit Hilfe von wissensdomänenspezifischen Vokabularen und vordefinierten Schemata modelliert werden.

Die Topologie eines Wissensgraphen, die sich grundsätzlich von anderen Topologien, wie z.B. der von Computernetzwerken oder sozialen Graphen, unterscheidet, ist durch besondere Merkmale gekennzeichnet: Wissensgraphen enthalten sowohl hierarchische (typisierte) als auch transversale Beziehungen zwischen Knoten. Weiter, ist die Topologie der Verwendung von vordefinierten wissensdomänenspezifischen Vokabularen sowie den unbeständigen Modellierungsgewohnheiten von Forschern ausgesetzt.

Die Analyse und das Verständnis der spezifischen Topologie sowie die Anwendung geeigneter Maße für die Beschreibung von Wissensgraphen kann die Entwicklung von z.B. Werkzeugen für die Profilbildung, Datenbank-Benchmarks, effizienten Datenstrukturen und Indizes, sowie Techniken zur Komprimierung von Graphdaten unterstützen und beeinflussen. Traditionelle Maße, die aus der Netzwerkanalyse bekannt sind, erfassen nur unzureichend die Semantik, die die Topologie eines Wissensgraphen mit sich bringt. Es ist daher von zentraler Bedeutung entsprechende Werkzeuge für die Analyse und geeignete Maße für die Charakterisierung von Wissensgraphen zur Verfügung zu stellen.

Davon motiviert widmet sich die vorliegende kumulative Arbeit diesem Themengebiet in drei Teilen.

Der erste Teil der Arbeit befasst sich mit der Einführung und Beschreibung eines Software Frameworks, welches der Akquisition von Wissensgraphen und deren Aufbereitung als Objektmodell dient sowie weitere graphtopologiebezogene Operationen zur Verfügung stellt. Mit diesem Werkzeug haben wir eine groß angelegte Analyse mit 280 Wissensgraphen aus neun Wissensdomänen durchgeführt und 54 verschiedene Graphmaße berechnet. Die Ergebnisse der Analyse und die aufbereiteten Graphobjekte sind Forschenden zur weiteren Verarbeitung frei zugänglich gemacht worden.

Darauf aufbauend befasst sich der zweite Teil der Arbeit mit der Untersuchung von bekannten Maßen aus der Netzwerkanalyse und Maßen die speziell für die Charakterisierung von Wissensgraphen entwickelt wurden. Unter Verwendung von statistischen Methoden und eines Machine Learning Klassifikationsverfahrens untersuchen wir ihre Aussagekraft und Relevanz hinsichtlich der Generierung prägnanter Beschreibungen von Wissensgraphen. Außerdem analysieren wir Maße, die geeignet sind, Graphen von anderen Wissensdomänen zu unterscheiden, um so wissensdomänenspezifische Besonderheiten aufzudecken und entsprechende Implikationen für bestehende Lösungen ableiten zu können.

Im dritten und letzten Teil der Arbeit verwenden wir unsere Ergebnisse aus dem ersten Teil, um Lösungen in anderen, für Wissensgraphen relevanten, Forschungsgebieten anzubieten. Wir befassen uns mit Datenbank-Benchmarks für Wissensgraphen und der Kritik an ihnen nur unzureichende Aussagen zu liefern, sofern synthetische Anfragen für Laufzeitmessungen verwendet werden. Wir stellen daher eine weitere Funktionalität unseres zuvor entwickelten Frameworks vor. Diese ermöglicht es anwendungsspezifische Anfragen auf der Grundlage von realen Daten aus Wissensgraphen zu generieren. Ferner stellen wir einen flexiblen und „business use case“-getriebenen Ansatz vor, der erlaubt durch Gruppenbildung Antwortzeiten von Datenbankanfragen realistischer zu beurteilen.

Diese Dissertation basiert auf zuvor veröffentlichten Papieren, die in hochrangigen internationalen Open-Access-Zeitschriften, auf internationalen Konferenzen und Workshops auf dem Forschungsgebiet der Semantic Web-Technologien per peer-review Verfahren begutachtet und publiziert wurden. Als Bekenntnis zur Offenen Wissenschaft wurden alle Programme und Ressourcen als Open-Source-Projekte unter MIT-Lizenz auf populären Quellcode- und Datenhosting-Plattformen veröffentlicht.

Contents

1

Introduction

1.1 Motivation

In the past decades, the acquisition and aggregation of data and their presentation on the World Wide Web has shaped not only information technology and computer science but also our daily lives. The findability and accessibility of data on the Web presents tremendous challenges to its users and to computers serving the content.

To organize and process the vast amount of information, computers need data to be structured and in a machine-readable format. Today, the most widespread model for organizing data and making them machine-readable is the *knowledge graph* (KG), in which Web entities are represented by vertices, and relationships between entities are represented by directed and labeled edges. Knowledge graphs are widely used in contexts such as Web search (e.g., the Google Knowledge Graph,[1] the Bing Knowledge Graph[2]); open access knowledge organization (e.g., Wikidata,[3] DBpedia [Auer et al., 2007], Freebase [Bollacker et al., 2008]); domain-specific knowledge organization (e.g., Microsoft Academic Graph [Sinha et al., 2015], SemMedDB [Kilicoglu et al., 2012], Product Knowledge Graph [Dong, 2018]); and in diverse domains of research, such as natural language processing (NLP) and artificial intelligence (AI; smart assistants). Thanks to (openly accessible) knowledge graphs and to the processing power of computers today, we can relate, find, and browse human knowledge on the Web in an organized way.

[1] `https://blog.google/products/search/introducing-knowledge-graph-things-not/`. Last accessed on October 12, 2020.

[2] `https://blogs.bing.com/search-quality-insights/2018-02/bing-entity-search-api-now-generally-available`. Last accessed on October 12, 2020.

[3] `https://www.wikidata.org/`. Last accessed on October 12, 2020.

The state-of-the-art graph-based model that provides the corresponding formats to make data on the Web machine-readable is the Resource Description Framework (RDF; Manola et al., 2004). It is the de facto standard to represent and share structured data as *Linked Open Data* (LOD; Heath and Bizer, 2011). Central to this concept is the representation of entities as Uniform Resource Identifiers (URIs) and the usage of knowledge-domain-specific vocabularies to model, interlink, and dereference in order to obtain further details about entities on the Web. The popularity of RDF has led to diverse initiatives, such as the Linked Open Data Cloud[4] (LOD Cloud), which is a collection of openly accessible RDF datasets[5] that are highly interlinked to one another. It is a prominent example of and a reference for the success of interlinked and queryable data, published and pushed by the scientific community, archival institutions, and industry from diverse domains, such as government, linguistics, and the life sciences.

Knowledge graphs published in RDF consist of a set of triples. Thus, their inherent structure is graph-based: The set of triples $\{subject, predicate, object\}$ composes a directed and labeled multigraph, with the set of all *subject*s and *object*s forming the vertices – that is, the described resources – and the set of *predicate*s forming labeled edges in the graph. Such relationships form hierarchical *as well as* transversal semantic links between the vertices in the graphs.

1.2 Problem Statement & Research Goal

Compared with other graph topologies – for example, those of computer networks or social graphs – topologies of knowledge graphs modeled with RDF impose distinct characteristics (Fernández et al., 2018). Traditional measures and methods known from network analysis fail to adequately capture these characteristics and to appropriately characterize and concisely describe the particularities of state-of-the-art knowledge graphs. This is primarily due to the semantics that knowledge graphs inherently entail. Generally, knowledge graphs contain terminological statements (TBox – schema definitions) and assertional statements (ABox – the data), which complement each other in a single knowledge graph. This imposes particular semantics on the graphs, such as typed and non-typed vertices, the recurrence of topological patterns (e.g., usage of particular edge labels per vertex type), and a significant and pervasive level of redundancy, which other graphs do not have. This leads to higher connectivity, shorter paths, and a high probability of the existence of "hubs" – that is, vertices (entities) with high attractiveness from other graph vertices.

Another aspect with significant influence on the characteristics of knowledge graph

[4]Linked Open Data Cloud. `https://lod-cloud.net/`. Last accessed on August 22, 2020.

[5]In this thesis, the terms *RDF dataset*, *knowledge graph*, *knowledge base*, and *RDF data graph* are used interchangeably. Unless otherwise stated, the use of the term *dataset* refers to an RDF dataset. Similarly, the terms *RDF knowledge graph* and *RDF graph* are used interchangeably, describing knowledge graphs modeled in RDF.

topologies is related to data quality and the modeling habits of RDF users. There is no inherent need for a prior schema when using RDF. Thus, openly accessible knowledge bases tend to be diverse and heterogeneous in their graph structure due to a (partly) inconsistent practice of vocabulary usage and schema conformance within and across knowledge domains (Bobed et al., 2020). Research around vocabulary usage extends to a wide range of disciplines, including profiling tools for the creation of descriptive summarizations (Ben Ellefi et al., 2018; Zneika et al., 2019); knowledge graph completion and link prediction approaches (Rosso et al., 2018); dataset archiving mechanisms to capture the dynamics of graph data evolution (Fernández et al., 2019); benchmarking suites to model realistic synthetic datasets, queries, and test storage strategies (Wylot et al., 2018); strategies to efficiently store (Fernández et al., 2019), encrypt, and compress RDF data (Fernández et al., 2020); user-friendly tools to support data modeling processes; and studies to measure qualitative dimensions of Linked Data (Zaveri et al., 2016).

The majority of related work studies vocabulary usage in terms of two aspects: observed structural patterns and statistical distributions at the level of the RDF model. Research that addresses the first aspect studies combinations of the entities described, together with their corresponding properties, whereas research that addresses the second aspect applies descriptive statistical methods to describe patterns of vocabulary usage in the datasets. However, the shape of the knowledge graph's *topology* remains widely unexposed.

Therefore, it is fundamental to provide tools to facilitate graph-related tasks, such as large-scale analyses (cf. Zloch et al., 2019), as well as meaningful measures to characterize knowledge graphs adequately (cf. Zloch et al., 2020). Performing graph-related analyses on the topology of knowledge graphs helps to understand, for example, their shape, their evolution over time, and their differences from other types of graphs. Moreover, it helps to find solutions and to derive implications for existing and future solutions in related research areas (cf. Zloch, 2016).

In the following, we present exemplary research areas with potential use and application of graph topological features.

– **RDF Benchmarking.** RDF benchmarks are substantial for systematically evaluating novel storage solutions for RDF data and RDF query language evaluation strategies. One goal of benchmark suites is to emulate real-world datasets and queries. Therefore, RDF benchmarks are mostly RDF-schema- and vocabulary-specific.

  Besides approaches for query execution workloads, a widespread and crucial evaluation criterion for RDF benchmarks is the synthetic data generation process. Generators must *conform* to known application- or knowledge-domain-specific vocabularies, while simultaneously being able to *scale up* the dataset size. However, results of benchmark suites are not meaningful enough when datasets and queries are generated artificially (Duan et al., 2011), and they may have little relation

to real-world datasets and queries. Furthermore, vocabulary usage varies across applications and knowledge domains, even when publishers comply with one particular vocabulary (Bobed et al., 2020). For example, two publishers of bibliographic data in the *Publications* domain may have different completeness levels and focus of the published data. Whereas one publisher publishes scientific papers with metadata such as titles and authors, the other may *additionally* publish results of a linguistic extraction process of the contents of the publications, such as keyword distribution and topic distribution per author.

Vocabulary usage has a significant impact on a graph's topology because cardinality definitions in a prior schema, for example, are directly reflected in the graphs as options/restrictions to impose new vertices and edges. Beyond aspects such as the dataset size and the conformance to a particular vocabulary, considering reliable statistics about the graph topology enables synthetic dataset generators to emulate RDF datasets more appropriately.

– **Graph Sampling.** The aforementioned challenges during the upscaling of a (synthetic) dataset apply also to downscaling, that is, sampling from a large graph. Graph sampling techniques try to find a representative sample from an RDF dataset of interest. Questions that arise in this research area are (1) how to obtain a (minimal) *representative* sample, (2) what sampling method to use, and (3) how to vary and assess measurements of the sample (Leskovec and Faloutsos, 2006).

  Sampling is usually performed concerning different aspects. Apart from qualitative aspects – such as RDF classes, properties, instances, and the vocabularies and ontologies used – topological characteristics of the knowledge graphs should also be considered. To this end, primitive measures of the graphs, such as the max in-, out- and average-degree of vertices, reciprocity, and density, may be consulted to achieve more accurate results.

– **Dataset Profiling and Evolution.** RDF datasets are mostly distributed and dynamic, as the model offers simplicity, flexibility, and ease of exchange. The analysis of a broad set of RDF datasets from openly accessible knowledge bases, such as the LOD Cloud, concerning (1) the dynamics of evolution (Bobed et al., 2020); (2) aspects of quality, such as the conformance to the expected usage of a vocabulary (Rashid et al., 2019); and (3) dataset similarity for dataset search (Sousa et al., 2020), for example, has presented significant challenges in recent years. Furthermore, (4) aspects of linkage (linking into other datasets) and connectivity (linking within one dataset) have been of particular interest. From the graph perspective, all the aforementioned aspects have an immediate impact on the shape and characteristics of the corresponding graph topology.

  Profiling tools help to create *RDF dataset profiles*, which are quantitative representations – that is, descriptive statistics – of the dataset of interest (Ben Ellefi

et al., 2018). These tools extract characteristics (features) adhering to the instance- and schema-level of an RDF dataset. However, graph-based measures extracted from the topology of an RDF graph are unexposed to a great extent. As dataset profiles facilitate the challenges mentioned above in various dimensions, profiling tools should also respect the extraction of graph-based measures from RDF graphs.

Research Goals

Concerning our motivation, the problem statement, and the mentioned use cases, the main objectives of this thesis are:

- to facilitate graph-related tasks on knowledge graphs – for example, the large-scale investigation of graph topologies and their specificities – in particular within popular knowledge domains (Chapter 2);
- to assess graph measure effectiveness and importance for knowledge graphs in individual knowledge domains in order to generate concise topological profiles (Chapter 3); and
- to leverage programmatic graph representations to provide solutions in the research areas mentioned before (Chapter 4).

This dissertation is guided by these objectives. Please refer to Section 1.5 for a detailed description of the resulting structure of the thesis.

1.3 Contributions of This Thesis

The contributions of this thesis with respect to the research goals outlined in Section 1.2 are summarized as follows:

(i) ***Open source framework to support graph-related tasks on knowledge graphs.*** We introduce an open source software framework with various capabilities. The primary purpose of the framework is to facilitate graph-related tasks on RDF knowledge graphs (Chapter 2).

Primarily, the framework has the capacity (1) to acquire RDF datasets, (2) to efficiently prepare graphs, and (3) to perform graph-based analyses on the prepared graphs. One of the framework's main features is to scale up to large graphs and a large number of datasets in parallel – that is, to prepare and compute graph measure analyses efficiently over large state-of-the-art knowledge graphs of hundreds of millions of edges. For graph measure analyses, the framework supports a total of 54 graph-based measures, grouped into different categories, including measures specifically defined for characterizing RDF graphs.

In addition, the framework enables us to exploit the benefits of graph representations of knowledge graphs, such as finding graph isomorphisms to generate query instances in the benchmarking use case (Section 4.2). Prospectively, the framework will also have the capability to serve as a general-purpose framework to facilitate the above-mentioned graph-related tasks on *non-RDF* knowledge graphs, for example, social graphs or retweet networks. However, in the current release, some implementations are tailored toward the semantics of RDF graphs, such as measuring the number and ratio of typed subjects.

The framework is built on state-of-the-art third-party libraries, is extendable, well structured and documented, and is open source (published under the MIT license). The code is maintained on GitHub.[6] Latest releases are also available via Zenodo (Zloch, 2020).

(ii) ***An analysis of topological differences in popular knowledge domains.*** We conduct a systematic graph-based analysis of a large and representative sample of openly accessible knowledge bases that were part of the LOD Cloud in late 2017 (Chapter 3). The analysis covers about 11.3 billion RDF triples from nine knowledge domains provided by the LOD Cloud: *Cross-Domain*, *Geography*, *Government*, *Life Sciences*, *Linguistics*, *Publications*, *Media*, *Social Networking*, and *User-Generated*.

For each dataset in the sample, we compute all available graph measures provided by the framework mentioned in (i) above. Our analysis report covers observations

[6]The framework's source code on GitHub: `https://github.com/mazlo/lodcc`. Last accessed on September 16, 2020.

on value distributions of measures from the group of basic graph measures, degree-based measures, degree distribution statistics, and other metrics proposed for RDF graph characterization. Further, it gives insights into the structure and differences of real-world knowledge graphs within popular knowledge domains concerning graph-related measurements. This is beneficial for existing and future developments in the research areas mentioned in Section 1.2, which provide knowledge-domain-dependent solutions.

(iii) ***Generation of topology profiles for a large sample of online knowledge graphs.*** In addition to the aforementioned analysis and report over 280 knowledge graphs, we provide a collection of all programmatic graph representations used in the experiment to facilitate reproduction of the results and further reuse (Zloch, 2018). This collection's main benefit is that all graphs were already acquired, pre-processed, and instantiated as graph objects. They are provided in a binary form (efficiently compressed), ready to be processed by third-party graph analysis libraries like *graph-tool* (Peixoto, 2014). This enables advanced investigations of statistical distributions, for example, computation of advanced centrality measures (importance of vertices, i.e., URIs in a graph), clustering (group building within a graph), linkage (linking into other datasets), connectivity and density (linking within one dataset).

Compared with previous studies from related work, which were limited to a small fraction of datasets, our sample enables reuse and large-scale analyses of a representative sample of knowledge graphs from an openly accessible source of datasets.

Besides that, we make a collection of all results available to the community as topological profiles (Zloch and Acosta, 2018). To facilitate browsing of the results and inspect the varying dimensions of particular datasets and knowledge domains, we provide a website (Zloch, 2018). The website provides access to necessary resources required for further analyses, like RDF dataset metadata, binary graph representation, measure descriptions, annotation of efficient measures, and more.

(iv) ***Identification of effective measures for graph characterization.*** We study graph measure *meaningfulness* and *efficiency* to describe graphs from nine popular knowledge domains (Chapter 3). In order to accomplish this, we follow a three-stage approach and seek to answer the following research questions (RQ):

- **RQ1: What is a non-redundant set of measures to characterize graphs effectively?** To characterize graphs or sets of graphs within knowledge domains concisely, graph descriptions have to be based on meaningful and non-redundant sets of measures, with each set providing significant information gain to the graph description (cf. Zneika et al., 2019).

To this end, this question aims at finding a concise and finite set of measures to reduce redundancy and maximize information gain through correlation analysis. This step improves the effectiveness of the resulting set of graph measures and their applicability, for instance, as part of machine learning models.

- **RQ2: Which measures describe and characterize individual knowledge domains most/least efficiently?** Datasets within the LOD cloud are categorized into nine distinct knowledge domains so that each dataset is associated with precisely one specific category. To understand the representativeness and variability of graph-related measures within a knowledge domain, we apply basic statistical metrics to investigate the heterogeneity of measure values within these domains. Afterwards, we discuss and identify representative measures for these knowledge domains.

  This provides insights into the capacity of individual graph-based measures to represent the nature of particular domains, and may contribute to discriminative models and to filtering out noisy features when profiling datasets.

- **RQ3: Which measures show the best performance to discriminate individual knowledge domains?** Concerning topological dynamics partly caused by vocabulary adoption in the popular knowledge domains, we can observe distinct characteristics of graph topologies in the individual knowledge domains. Thus, the question is which graph measures are most descriptive and therefore important *within* one particular knowledge domain to discriminate dataset categories. In contrast to RQ2, where we apply statistical metrics to investigate representativeness and variability, here, we determine the most important graph measures through a state-of-the-art machine learning classification model.

Various approaches, for example synthetic dataset generators, can benefit from the findings. Benchmark suites most often target a particular domain of interest while generating and upscaling a dataset. For the *Publications* domain, for example, besides the typically used vocabularies, a generator should follow a specific set of measures and range of values, in order to be aligned with the shape of topologies of real-world knowledge graphs in this category.

(v) ***Open source framework facilitating application-specific benchmarking.*** From the list of use cases mentioned in Section 1.2, we take on the benchmarking use case and leverage graph representations to facilitate application-specific benchmarking (Chapter 4).

Our contributions in this regard are twofold. First, we propose a customizable benchmarking framework and introduce a novel *use-case-driven* approach. In

contrast to the classical approach, in which the runtime of database queries is assessed as a whole, in this approach, the runtime is assessed using groups of queries that are represented by application-specific business use cases. This allows for more differentiated statements about the behavior and implications of individual query groups in order to determine the best-performing database system. Our experiment employs data and queries from a real-world application and considers a large number of different types of database management systems (triple stores, relational and graph-based databases, column- and row-stores).

Our second contribution leverages graph representations of knowledge graphs to generate graph queries typically found in real-world query logs, for example, those studied by Saleem et al. (2015a). The queries are generated from a given set of query templates by finding appropriate matchings in the corresponding graph representations. Our software framework mentioned in (i) implements this feature together with a predefined set of query templates taken from a state-of-the-art benchmarking suite. This contribution provides a solution compared with tailored benchmarks, which prefer to consider synthetic data and query generators that cannot reproduce the observed increasing variability of real-world datasets.

1.4 Related Publications

This dissertation is based on previously published papers in the research area of Semantic Web technologies that were submitted to high-ranked international peer-reviewed open access journals, international conferences, and workshops.

A list of accepted and published papers is presented below. Publications that constitute the basis for the Ph.D. thesis, with references to the corresponding chapters and sections in this dissertation, are mentioned first.

2020:

– Matthäus Zloch, Maribel Acosta, Daniel Hienert, Stefan Conrad, and Stefan Dietze (2020). Characterizing RDF graphs through graph measures – framework and assessment. In: *Semantic Web* – Pre-press. DOI: 10.3233/SW-200409.
Contributions: Matthäus Zloch extended the framework with RDF-graph-based measures from related work and performed all experiments, which were under the supervision of Stefan Dietze. Matthäus Zloch also created all figures and tables and prepared 90% of the manuscript.
Dissertation sections: 3.1
Status: Published.
Impact factor: 3.524 (2019)

2019:

– Matthäus Zloch, Maribel Acosta, Daniel Hienert, Stefan Dietze, and Stefan Conrad (2019). A software framework and datasets for the analysis of graph measures on RDF graphs. In: *The Semantic Web – 16th Extended Semantic Web Conference (ESWC 2019)*. Vol. 11503. Lecture Notes in Computer Science. Springer, pp. 523–539. ISBN: 978-3-030-21348-0. DOI: 10.1007/978-3-030-21348-0.
Contributions: Matthäus Zloch developed the software framework, designed the experiments, and performed the graph-based analysis over all datasets. He created the website and collected the corresponding resources (datasets, metadata files, etc.). The manuscript was prepared by Matthäus Zloch (80%), Maribel Acosta, and Daniel Hienert, under the supervision of Stefan Conrad and Stefan Dietze.
Dissertation sections: 2.1, 2.2, 3.1, 4.2.2
Status: Published.
Acceptance Rate: ∼29%
Remarks: Nominated for the Best Student Paper award and Best Resource Paper award; won the Best Student Paper award.

2017:

– Matthäus Zloch, Daniel Hienert, and Stefan Conrad (2017). Towards a use case driven evaluation of database systems for RDF data storage – a case study for statistical data. In: *Joint Proceedings of BLINK 2017: Benchmarking Linked Data and NLIWoD3: Natural Language Interfaces for the Web of Data (BLINK 2017-NLIWoD3)*. CEUR Workshop Proceedings 1932. Aachen: CEUR-WS.org. URL: http://ceur-ws.org/Vol-1932/#paper-08.
Contributions: Matthäus Zloch developed the framework, created all figures and tables, and performed all experiments, which were under the supervision of Daniel Hienert. The manuscript was prepared by Matthäus Zloch (90%) and Daniel Hienert, under the supervision of Stefan Conrad.
Dissertation sections: 4.3
Status: Published.

A number of other publications (co-)authored by Matthäus Zloch during the Ph.D. study period do not play a major role in this dissertation but are strongly related to Semantic Web technologies. They include, among others:

2020:

– Dimitar Dimitrov, Erdal Baran, Pavlos Fafalios, Ran Yu, Xiaofei Zhu, Matthäus Zloch, and Stefan Dietze (2020). TweetsCOV19 – a knowledge base of semantically annotated tweets about the COVID-19 pandemic. In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. CIKM '20. New York, NY, USA: ACM, pp. 2991–2998. ISBN:

978-1-450-36859-9. DOI: 10.1145/3340531.3412765.
Contributions: Matthäus Zloch set up and administered the infrastructure, imported all RDF data into the data store, and enabled SPARQL query evaluation. He also created the project website.[7] Moreover, he contributed to formulating SPARQL-query examples in early versions of the manuscript and proofread the manuscript.
Status: Published.

2019:

– Andon Tchechmedjiev, Pavlos Fafalios, Katarina Boland, Malo Gasquet, Matthäus Zloch, Benjamin Zapilko, Stefan Dietze, and Konstantin Todorov (2019). ClaimsKG: a knowledge graph of fact-checked claims. In: *The Semantic Web – 18th International Semantic Web Conference (ISWC 2019), Proceedings Part II*. vol. 11779. Lecture Notes in Computer Science. Springer, pp. 309–324. ISBN: 978-3-030-30795-0. DOI: 10.1007/978-3-030-30796-7_20.
Contributions: Matthäus Zloch set up and administered the infrastructure, imported all RDF data into the data store, and enabled SPARQL query evaluation. He also created the project website.[8] Furthermore, he made minor contributions to the manuscript and to proofreading.
Status: Published.

2016:

– Matthäus Zloch (2016). Methods for automatic selection of database systems for optimized query performance. In: *46. Jahrestagung der Gesellschaft für Informatik (INFORMATIK 2016)*. Vol. P-259. Lecture Notes in Informatics (LNI) – Proceedings. Bonn: Gesellschaft für Informatik e.V., pp. 2019–2024. ISBN: 978-3-88579-653-4. URL: https://dl.gi.de/20.500.12116/1097.
Contributions: The research of the concept and the preparation of the manuscript was carried out entirely by Matthäus Zloch.
Status: Published.

[7] TweetsCOV19 website. https://data.gesis.org/tweetscov19. Last accessed on November 2, 2020.

[8] ClaimsKG website. https://data.gesis.org/claimskg. Last accessed on November 2, 2020.

1.5 Structure of the Thesis

This is a cumulative dissertation. It consists of three building blocks, each of which is treated in a separate chapter. Central to each chapter is one publication, addressing one of the objectives formulated in Section 1.2. In addition to a summary, the author's contributions, and the impact of each of the corresponding publications, we will present further details, which could not be published with the manuscripts.

The following gives a brief overview of each chapter's content.

Chapter 2: Acquisition of Knowledge Graphs and Graph Measure Computation
Chapter 2 addresses our first objective, that is, to facilitate graph-related tasks on RDF knowledge graphs, such as large-scale analyses of graph topologies. We present a software framework capable of acquiring knowledge graphs from one popular source of datasets. Further, it can prepare and perform graph-based analyses over the corresponding graph topology. In the related publication, we introduce the framework, provide a collection of prepared datasets, and report on graph-based measures computed with the framework. In addition, as the framework is central to follow-up research activities, we describe the package structure and other related functionalities of the framework.

Chapter 3: Assessing Graph Measures for Knowledge Graph Characterization
Chapter 3 continues with addressing our second objective, that is, assessing graph measures for knowledge graph characterization. The corresponding publication presents our large-scale study of measure meaningfulness and effectiveness. We question a graph measure's informative value and predictive power to discriminate knowledge graphs by popular knowledge domains. In addition to the results presented in the publication, we provide further details about the experimental setup. Please note that the related publication also contains a comprehensive related work section that aligns this whole work with existing tools and studies for graph analysis and measure computation in the research field of Semantic Web technologies.

Chapter 4: Towards an Application-Specific RDF Benchmarking Suite
The third objective of the thesis is to leverage graph representations from RDF knowledge graphs to provide customized solutions for a related research field in Semantic Web technologies. Thus, in this chapter, we take on the benchmarking use case mentioned in Section 1.2 and present two contributions towards an application-specific benchmarking suite. We demonstrate how graph representations can facilitate the generation of customized application-specific benchmark queries, based on data from real-world knowledge graphs.

The second contribution is represented by a published work. It deals with an approach

to evaluate queries executed against different types of RDF data stores to evaluate query runtime more reliably.

Chapter 5: Conclusion and Future Work
The thesis concludes with a summary of the achievements made concerning our objectives and the stated contributions. Further, we mention implications and derive future work plans for each work.

2

Acquisition of Knowledge Graphs and Graph Measure Computation

Our first objective is to facilitate graph-related tasks on state-of-the-art knowledge graphs, such as large-scale graph analysis and measure computation. To achieve this, we need to acquire a large number of RDF datasets and efficiently prepare them in such a way that we can operate on their graph structure. To provide a solution to the mentioned challenges, we introduce a software framework (Zloch et al., 2019).

The framework can deal with typical issues one has to face when working with publicly available RDF data, for example, format and media type issues. To publish data in RDF, one can choose from several different formats, of which the most prominent are RDF/XML, Notation 3, N-Triples, and Turtle. Not all formats support the ad hoc creation of a graph structure. Existing libraries that support these formats do not support the out-of-the-box computation of graph measures over the obtained graph structures. Furthermore, to do large-scale analyses, there is no unique and standardized way to acquire RDF datasets from particular knowledge domains. The Linked Open Data Cloud (LOD Cloud) has been a prominent example of the availability and linking of openly accessible RDF datasets. However, submitting a new dataset does not come with any validation of the supported formats or media types and the aimed availability and sustainability of the datasets. This is problematic for automated acquisition and analysis techniques.

Section 2.1 gives a summary of the related publication. Section 2.2 describes the structure of the framework introduced in Zloch et al. (2019) and shows additional functionalities. A brief description of the supported graph measures of the framework is given in Section 2.3.

2.1 A Software Framework and Datasets for the Analysis of Graph Measures on RDF Graphs

Matthäus Zloch, Maribel Acosta, Daniel Hienert,
Stefan Dietze, and Stefan Conrad.

In: *The Semantic Web – 16th Extended Semantic Web Conference (ESWC 2019), Proceedings*, Vol. 11503. Lecture Notes in Computer Science. Springer, pp. 523–539.
DOI: 10.1007/978-3-030-21348-0.

Summary

In the paper, we propose a software framework that has two major goals. The primary goal is to provide an integrated code base for graph-related tasks on RDF knowledge graphs. These tasks include the acquisition and preparation of RDF knowledge graphs concerning a graph-based analysis of their topology. Other tasks include the generation of topological profiles, by employing a number of graph-based measures, as well as query generation (see Section 4.2 below). To this end, the second goal of the framework is to provide this functionality at a large scale and for a high number of state-of-the-art knowledge graphs (hundreds of millions of edges) in parallel. The paper describes all stages of the processing pipeline, that is, acquisition, preparation, graph instantiation, and measure computation, and gives an overview of five groups of graph measures supported by the framework at the time of publication (Zloch, 2020).

To evaluate and stress our framework, we described the acquisition of about 280 RDF datasets from a popular dataset provider, the LOD Cloud.[4] In a subsequent step, we calculated 28 graph-based measures[9] for all graphs with the framework (see Section 2.3 below). We reported on preliminary analysis results of measure value distributions from three different groups of measures (*basic graph measures*, *degree-based measures*, and *degree distribution statistics*; see Section 2.3 below). The report includes all datasets acquired from the nine knowledge domains provided by the LOD Cloud (Zloch and Acosta, 2018).

Concerning our preliminary analysis results presented in the paper, we observed general and knowledge-domain-driven particularities, which are induced by the shape of the individual graph topology. For example, knowledge graphs in *Cross-Domain* category have a particularly low average degree; the average degree over all graphs is approximately eight; the *Publications* domain has the highest number of graphs reporting a scale-free behavior (Zloch et al., 2019). For solutions respecting domain-driven specificities, such as synthetic dataset generation, this may have strong implications.

[9]The framework supported 28 graph-based measures at the time of manuscript publication. For the full set of graph measures and recent releases, see Section 2.3 or Zloch (2020), respectively.

Finally, we studied measure correlation and identified a set of nine measures, which do not correlate with each other, and are therefore candidates for a minimal set of measures to characterize knowledge graphs.

Latest releases of the framework are available under MIT license via Github and Zenodo (Zloch, 2020).

Importance and Impact on This Thesis

The paper was nominated for the Best Student Paper award and Best Resource Paper award at the Extended Semantic Web Conference (ESWC) 2019, and received the Best Student Paper award for its presentation and impact to the research community. Thus, the two resources introduced in the paper – the software framework and the pre-processed knowledge graphs – have significant importance for the thesis and our follow-up research activities.

Follow-up research ideas built on the investigation of measure effectiveness and importance, in particular, to distinguish knowledge domains by means of graph-topological specificities. Chapter 3 addresses the assessment of graph measures and introduces our publication about effective measures for RDF graph characterization in the individual knowledge domains.

To facilitate reuse of the software framework, and to demonstrate its connectivity to related research areas in the Semantic Web community, we included a functionality to leverage a graph representation for generating queries in application-specific settings (see Section 4.2).

Author Contributions

The first author, Matthäus Zloch, developed the software framework, designed the processing pipeline, and chose the measures to compute over the graphs. He also created the overall structure and wrote the majority of the content of the paper. And finally, he created the website in order to have a browsable version for all datasets and measures (Zloch, 2018).

Maribel Acosta contributed to Section 2 by preparing related work. Later on, she created Figure 2 and Figure 3, in order to provide an overview of the corresponding measure values. Daniel Hienert and Matthäus Zloch made first efforts to analyze the results from the graph measure analysis. Daniel Hienert then had the idea to analyze the correlation coefficients of all values. He created first figures that showed the dependencies. Later on, these figures were replaced by Figure 5, also created by Maribel Acosta.

The paper underwent proofreading by all authors. The whole work was under the general supervision of Stefan Dietze and Stefan Conrad.

2.2 Framework Functionalities and Structure

In Zloch et al. (2019), we introduced a software framework supporting graph-related tasks, such as dataset acquisition, preparation, and graph measure computation, on knowledge graph topologies. In addition to the publication, this section provides an overview of the core functionalities of the framework (see Figure 2.1) and a technical overview of the package structure (see Figure 2.2).

Our software framework has a modular package structure. It is designed with the focus to be easily maintainable and extendable. For example, implementation of core functionalities, such as data acquisition and graph measure computation, are separated from executable code.

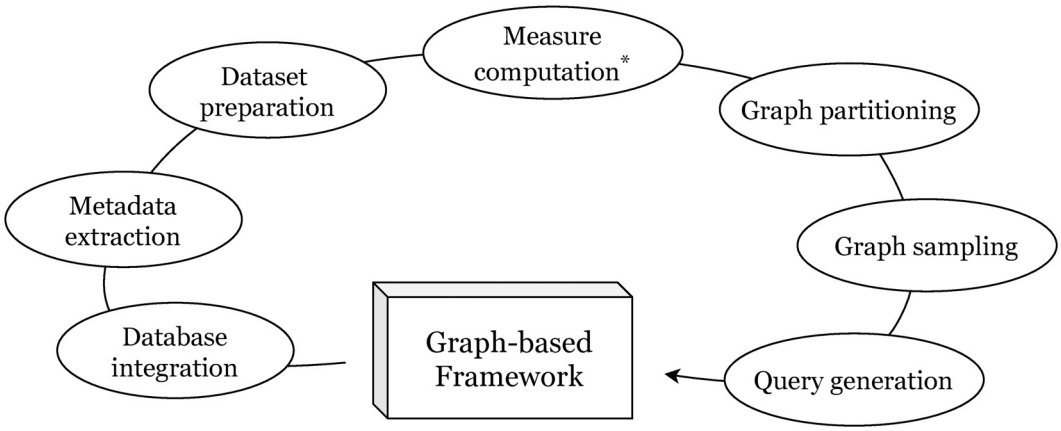


Figure 2.1: **Overview of core functionalities** provided by our framework.
* partly provided by the third-party library *graph-tool* (Peixoto, 2014).

Most of the code is written in the language Python. Python has a rich community and allows smooth integration and usage of third-party libraries, such as the integrated graph-analysis library *graph-tool* (Peixoto, 2014).

In the following, we summarize the functionalities and describe the individual packages. Figure 2.2 shows an overview of the package structure and the most essential packages of the framework.

`bin`. Some tasks – for example, the logic behind RDF data transformation into N-Triples, the merging of edgelists created by multiple RDF data files, or the dereferencing of hashed vertex labels – require the employment of third-party Unix-tools. These tasks are handled via shell scripts, which are then invoked by the framework. The shell-scripts can be found in the `bin` folder.

`constants`. Files in the package `constants` are imported and read throughout the whole framework. Some values, such as property files for a specific database connection, may be adjusted by users. Others values – for example, the path and the mapping to RDF transformation scripts or the supported (compressed) file and media types – should preferably be adjusted by developers.

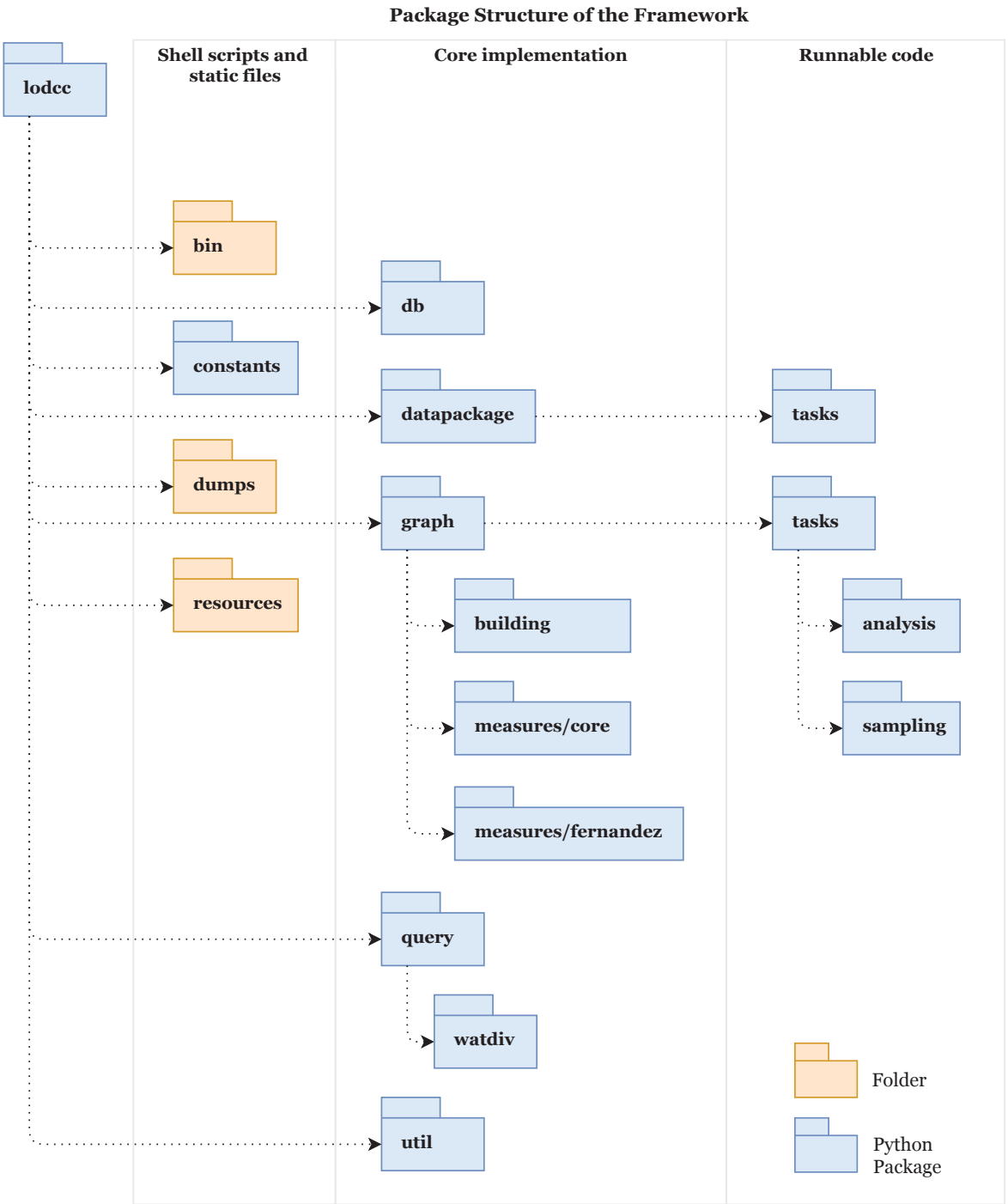


Figure 2.2: **Package structure overview** of the framework's most important packages.

`db`. The framework has support for the relational database *SQLite*[10] to store metadata about the corresponding knowledge base – for example, category, media types, download URLs, and the obtained graph measure values after graph measure computation. The package `db` contains related code – for example, the loading of datasets from the corresponding tables – which is used throughout the framework. The support for Postgresql and MySQL database has been discontinued since version v0.5.

`datapackage`. This package contains code for the optional preliminary initialization of datasets from datahub.io,[11] an online platform that provides access to open (RDF) data and metadata, such as the datasets from the Linked Open Data Cloud.[4] The framework offers the functionality to extract metadata from the so-called *datapackage.json* file provided by datahub.io. It contains the available media types and URLs to access and download RDF dataset dumps.

`graph`. The `graph` package is split up into further subpackages: `graph.building` contains code for graph preparation from RDF datasets and graph instantiation; `graph.measures.core` contains the set of traditional network-based measures, such as the fill, h-index, and diameter, which were described in the paper beforehand; the package `graph.measures.fernandez` contains RDF-related measures developed by Fernandez et al. in Fernández et al. (2018), which were integrated into the framework.

Measures in the package `graph.measures.fernandez` are much more computationally intensive than the set of network-based measures in the `core`-package. For example, many measures require creating lists of tuples of vertices and outgoing (or incoming) edges. One must then group these lists and count the number of occurrences of the outgoing edges. This does not scale well for large graphs with hundreds of millions of edges.

To improve this, this package includes an implementation of a graph-partitioning feature that may be configured before execution. It enables to dispatch chunks of work to subsets of vertices/edges, and improves overall computation time while requiring fewer resources of the host system. Please note that at the time of writing, this feature is not yet production-ready.

`query`. The package `query` contains an implementation for SPARQL query instantiation from SPARQL query templates. It works by finding subgraph isomorphisms of the query graphs in the corresponding graph. An example set of state-of-the-art real-world SPARQL query templates from the *Waterloo Diversity Benchmark*

[10] SQLite, an embedded SQL database engine. `https://sqlite.org/`. Last accessed on September 10, 2020.

[11] Data sharing provider. `https://datahub.io`. Last accessed on October 26, 2020.

Suite (WatDiv) can be found in the `query.watdiv` subpackage. Please find details of that implementation in Section 4.2.

`util`. The package `util` provides additional helper methods. For example, it provides methods to de-reference hash values of vertex or edge labels, which employs a brute-force search mechanism in the original data. This functionality is required for query generation, for example (see Section 4.2.2). The package also contains legacy code of former versions of the framework. For instance, vertices used to be coded as unique integer values in order to save memory and hard disk space consumption (Kunegis, 2013).

`tasks.*`. Some of the packages contain a `tasks` subpackage (see the far right-hand column in Figure 2.2), for example, `datapackage.tasks`, `graph.tasks`. They contain parameterizable executable code. For instance, to prepare a bunch of RDF datasets for graph instantiation and graph analysis, one may use the following command:

```
$ python3 -m graph.tasks.prepare \
             --from-db nobelprizes museums-in-italy \
                       oecd-linked-data transport-data-gov-uk \
             --threads 3
```

Detailed descriptions can be found in the README-file of the corresponding subpackage or via the `--help` parameter.

2.3 Supported Measures for Knowledge Graph Characterization

Supplementary to Zloch et al. (2019), we give a descriptive summary of the two groups of measures supported by our software framework to perform knowledge graph characterization. The first group contains measures from classical network analysis, which are typically used to characterize non-RDF graphs. The second group contains novel measures introduced by related work primarily for RDF knowledge graphs. We also address the applicability of the measures to non-RDF graphs.

2.3.1 Graph Measures

In Zloch et al. (2019), we introduced measures from classical network analysis that can also be applied to graphs imposed by the RDF data model. As there are many measures to characterize graphs in classical network analysis, we considered a subset only. We based our choice on the following criteria:

Popularity of the measure in related literature, for example, the size (the number of vertices) and volume (the number of edges) of the graph. Some of these measures – such as like the average degree and degree centrality, as well as statistical measures on the degree distribution – represent basic and rather primitive characteristics of graphs and are computationally not expensive.

Relevance of the measure, in particular to RDF graph characterization. These represent measures such as the number of parallel edges and reciprocity that respect the distinct graph topology of RDF graphs compared with social graphs and computer networks, for instance.

Impact of the measure with regard to RDF dataset discrimination concerning popular knowledge domains. Such measures – for example, the diameter (the longest shortest path between two vertices in the graph) - can be computationally intensive.

We categorized the chosen measures into five groups: (1) basic graph measures, (2) degree-based measures, (3) centrality measures, (4) edge-based measures, and (5) descriptive statistical measures. Zloch et al. (2020) provided a detailed description of the measure groups and the formalization of the chosen measures concerning RDF graphs.

2.3.2 RDF-Graph-Based Measures

In addition to the measures mentioned in Section 2.3.1, recent releases of the framework implement a wider set of measures, that is, RDF-graph-specific measures.

Knowledge graphs modeled in RDF have a topology that is distinct from that of other graphs, such as social graphs or computer networks, due to the pervasive existence of hierarchical relations. Relations within the ABox (assertional statements – the data) are complemented by relations within the TBox (terminological statements – schema definitions, e.g., `rdfs:subClassOf`) and between the ABox and the TBox. The most well-known example, which adheres to almost every description of a resource in an RDF knowledge graph, is probably `rdf:type`. These particularities are directly reflected in a graph's topology, and lead, for example, to higher overall connectivity and existence of redundant structural patterns in the graphs, and as such, they cannot be captured with ordinary measures. In addition to primitive measures, such as the number of vertices/edges and the distribution of vertex degrees, there have been some efforts to introduce measures that capture particularities of RDF graphs.

Fernández et al. (2018) introduced a comprehensive list of measures for capturing low-level metrics tailored for RDF graphs. In particular, they addressed the development of efficient data storage techniques, index structures, and compression algorithms for RDF data. The measures introduced are grouped into six groups: *subject*

and object degrees; *predicate degrees*; *common ratios*; *subject-object degrees*; *predicate lists*; and *typed subjects and classes*.

The main difference between these measures and the measures introduced in Section 2.3.1 is that the former additionally consider fine-grained combinations of (multivalued) pairs of vertices of specific types, such as subject-predicate and subject-object. Moreover, the distribution of predicates, repeated predicate lists in use, and the distribution of subjects and objects per predicate receive special attention. Such measures allow frequent patterns in RDF data and their graph topology to be exposed when subjects are described. By introducing a number of different ratios to capture repetitions of vertex usage, they also help to expose particular local constructs, such as vertices acting as a "star" (multiple incoming edges) and the presence of paths between vertices.

The authors consider that these measures characterize RDF graphs in particular, due to the required classification of vertices into types – that is, a subject or an object. On closer inspection, some of the measures can be computed on non-RDF graphs, too. To this end, we can treat a subject vertex as one having outgoing edges, and an object vertex as one having incoming edges. However, the existence of constant labeling of edges is still necessary for most of the measures introduced. See our future work plans on this in Section 5.2.

3

Assessing Graph Measures for Knowledge Graph Characterization

The software framework introduced in the last chapter enables us to perform graph-related tasks, such as knowledge graph acquisition and large-scale graph-based analyses on their topologies, by means of the provided set of graph measures. However, some measures involve a fair degree of complexity and are computationally expensive. Moreover, after computation, a measure may be of no additional informative value when included in a topological profile, as it may be redundant due to the fact that it was unable to capture knowledge-domain-dependent specificities.

Thus, our second objective in this dissertation is to detail our investigation of graph topologies and their specificities, particularly within distinct knowledge domains. To this end, we investigate the *quality* of our measures concerning the generation of *concise* topological profiles for knowledge graphs in the given knowledge domains.

Central to this chapter is the publication described in Section 3.1, which deals with the assessment of the efficiency of graph measures and their informative value. In addition to the description of the experimental setup and results in the publication, Section 3.2 addresses the challenges of determining measures that are particularly important for the characterization of knowledge graphs in the individual knowledge domains.

3.1 Characterizing RDF Graphs Through Graph-Based Measures – Framework and Assessment

Matthäus Zloch, Maribel Acosta, Daniel Hienert,
Stefan Conrad and Stefan Dietze.

In: *Semantic Web – Pre-press*. DOI: 10.3233/SW-200409.

This paper addresses our second objective, which is to assess graph measure effectiveness.

Summary

In this paper, we follow up on our previous investigations to identify graph measures that are non-redundant, meaningful, and efficient. To achieve this, we follow a three-stage approach and let the following research questions (RQ) guide our experiments:

RQ1 What is an efficient and non-redundant set of features for characterizing RDF graphs?

RQ2 Which measures and values describe and characterize knowledge domains most/least efficiently?

RQ3 Which measures show the best performance to discriminate knowledge domains?

To answer these questions we employ various methods commonly known in statistics and machine learning (feature selection), such as analyzing the Spearman correlation coefficients (Spearman, 1904) and performing low variance and univariate statistical tests, for example, chi-square, mutual information (MI), and maximum information-based nonparametric estimation (MINE) (Pedregosa et al., 2011). To determine a measure's capacity to discriminate knowledge graphs from other categories, we employ a state-of-the-art machine learning classification model (see Section 3.2 below). Measures performing well in the learning process can be considered useful and important for particular categories, that is, one or more of the nine knowledge domains provided by the LOD Cloud.

To make the study more comprehensive, and to respect recent findings on RDF graph characterization from related work, we implemented an additional set of 29 RDF graph measures into our framework (see Section 2.3.2, below) and repeated the graph-based analysis on all 280 knowledge graphs acquired before (see Section 3.2.2 below).

As a result of the investigations, we identified a distinct set of 29 measures as being meaningful, and 13 measures as having the capacity to discriminate graphs from other knowledge domains particularly well. While respecting the measures' individual

semantics, the paper gives a comprehensive report about the particularities of the inspected knowledge domains. For instance, in contrast to the *Cross-Domain* category, knowledge graphs in the *Publications* category report on a regular use of vocabularies, and thus, contain recurrent patterns in the topologies; a distinctive factor of knowledge graphs in the *Linguistics* category is their unusually large diameter.

Most of the measures considered important for knowledge graphs in the given knowledge domains were measures particularly designed to capture topological specificities of RDF graphs. This confirms our claims stated in Section 1.2 and previous findings of related work that knowledge graph topologies are distinct from those of other types of graphs. Therefore, it is fundamental to provide appropriate tools and meaningful measures to characterize knowledge graphs adequately.

Importance and Impact on This Thesis

As some measures are computationally expensive, the across-the-board computation of all graph measures over a graph topology is computationally expensive, not always reasonable, and thus discouraged. Therefore, we identified an efficient set of measures for the sake of creating topological profiles that contain meaningful and essential measures only. Hence, the paper constitutes a major contribution to the objectives of the thesis formulated in Section 1.2.

We are confident that various research fields related to RDF knowledge graphs can derive implications from our findings. Section 7 in the paper details implications in some domains of RDF research, such as RDF benchmarking (synthetic dataset generation), graph sampling, and RDF profiling (e.g., frameworks for quality evaluation).

In fact, to extend and deepen our knowledge about the particularities of knowledge graph topologies modeled in RDF, we consider, as future prospect, including non-RDF graphs and investigating the differences (see Section 5.2.2), which can also be done with the methods provided by our framework.

Author Contributions

The first author, Matthäus Zloch, extended the software framework with RDF graph-based measures. He created the experimental environment, performed all experiments, and presented the results to the other authors. Matthäus Zloch also designed and created all figures and tables. He created the overall structure and wrote the majority of the content of the paper.

Maribel Acosta suggested to include measures from Fernández et al. (2018). She also supervised Matthäus Zloch during the implementation of the graph-partitioning feature for this set of measures (see Section 2.2). Maribel Acosta reviewed all mathematical notations in Section 3 of the paper.

The design of the research questions and experimental setup was under the supervision of Stefan Dietze. He also contributed to parts of Section 1 and Section 6.

The paper underwent proofreading by all authors. The whole work was under the general supervision of Stefan Conrad and Stefan Dietze.

3.2 Determining Graph Measure Importance

In addition to the results of our investigation on overall and category-wise measure importance presented in Zloch et al. (2020), we describe in this section the setup of our learning pipeline, and provide additional results on the models' predictive power.

First, Section 3.2.1 gives a summary of the general functionality of the employed model. Section 3.2.2 describes standard but necessary data preparation tasks applied to the input datasets. Section 3.2.3 details the two classification tasks and the machine learning pipeline we set up. Section 3.2.4 presents and discusses the additional results and mentions some caveats and limitations.

3.2.1 Introduction

In Zloch et al. (2020) we employed random forest (Breiman, 2004; Louppe, 2014), a popular and state-of-the-art machine learning multiclass classification model, to determine overall and category-wise measure importance. To this end, we set up two classification tasks and treat our graph measures as features in the learning process (see Section 3.2.3). The following gives a brief summary of the functionality of the employed classifier.

Central to random forest is the concept of a decision tree. A decision tree is a binary tree, where each node in the tree represents a feature (i.e., a graph measure) and the leaf nodes represent the target variables (i.e., dataset categories) to be predicted. Besides the ability to predict, decision trees can be employed to obtain feature importance scores at each node in the tree after building it up from the available training data. Random forest is an *ensemble method*, which means that it consists of a predefined number of single decision trees that are built up and evaluated from different subsamples of the original dataset. Final prediction performance is then averaged over the given number of single trees. By this means, random forest tackles the tendency of single Decision Trees to overfit the data (Louppe, 2014).

For a random forest model, it is crucial to find the tree with the best choice of nodes (i.e., graph measures) where at each node the value of a predefined cost function is minimized. The cost-function evaluates the splitting of the input data at each node concerning the target variable, and is the final indicator for the efficiency of the represented feature at that node. The cost is minimized when the feature is able to perfectly split its values according to the given target variable. Popular cost functions for features of decision trees are *Gini impurity* and *entropy*.

A random forest model reaches acceptable prediction accuracy in many situations and is available for the Python programming language (Pedregosa et al., 2011). However, predicting categories for RDF datasets imposes some challenges, which we depict in the following. Please note that, in this context, we use the terms *knowledge domain*, *category*, and *class* interchangeably.

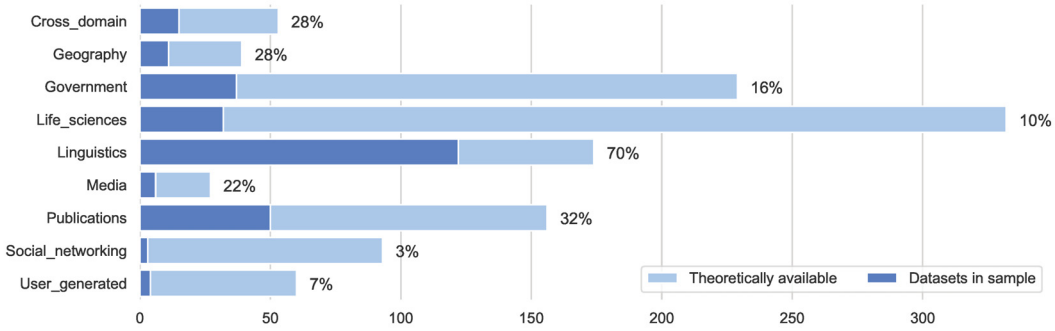


Figure 3.1: **Distribution of RDF datasets** in the LOD Cloud in the corresponding knowledge domains (light blue). The percentage value on each bar gives the ratio between the theoretically available number of datasets and the number of datasets we acquired and prepared successfully.

3.2.2 Data Preparation

To obtain reasonable results from a prediction model, data preparation is an essential part of a learning process. Typically, input data to machine learning models is prepared according to the aspects of (i) class imbalance and (ii) value distribution of the individual features.

Class imbalance

Class imbalance has a strong impact on the accuracy of prediction models. In our context, class imbalance means the unequal distribution of RDF datasets across the known different classes (i.e., categories). If a model was trained on a set with overrepresented datasets from the majority class (the class with the largest number of datasets), the model will likely have high overall accuracy, but poor performance on datasets of the minority class, that is, the underrepresented category (the one with the smallest number of datasets). Figure 3.1 shows the distribution of RDF datasets in the LOD Cloud, and the corresponding percentage of acquired datasets (see Table 2 in Zloch et al. (2020)).

Class imbalance is addressed by making classes (approximately) equal in size. This can be achieved, for example, via (a) *downsampling* or (b) *upsampling*. Option (a) makes all classes except the minority class smaller, and downsamples datasets of each class to the size of the minority class, via random sampling of datasets, for instance. However, especially in situations where the set of classes contains one class with a very low number of datasets, this approach would throw away potential and probably valuable training data from the other classes.

The naive approach to option (b), upsampling, is duplication of datasets of each class up to the size of the majority class. However, duplication, has negative consequences for the predictive power of the model. It leads to good accuracy of the

predictive model that can be over-interpreted and which does not show how good the model is on datasets from the underrepresented category. Instead of duplication, a more robust approach is to create synthetic datasets in the category with underrepresented samples. This leads to a more distinctive set of datasets as training data in the corresponding categories. We employed the *synthetic minority over-sampling technique* (SMOTE) (Chawla et al., 2002; Pedregosa et al., 2011), which is the most prominent approach to tackling class imbalance in statistical learning.

Feature value distribution

We investigated the heterogeneity of our graph measure values for knowledge graphs in various knowledge domains (Zloch et al., 2020). We found that most of the values have different scales. Their values range from natural numbers of hundreds to hundreds of millions, for example, in the case of number of edges; similarly decimal numbers with high precision ranging from one to six decimal places (10^{-6}), for example, in the case of fill and reciprocity. Apart from that, we found that many features have single to many isolated outliers. This can be problematic for machine learning prediction models, as most of the implementations expect feature values to be standard normal distributed (Pedregosa et al., 2011). This is due to the fact that normal distribution of feature values is very common when measuring physical and economical phenomena (Wonnacott and Wonnacott, 1990). However, we are the first to do graph-based analyses of this extent on such a large sample of knowledge graph topologies (Zloch et al., 2020). Thus, we do not have any evidence that our graph measure values are inherently normally distributed too. In addition, samples, such as our samples from the LOD Cloud may not necessarily reflect the same type of distribution concerning a particular measure as the original set (Stumpf et al., 2005).

To return robust prediction results, and to perform well in terms of computation time, our input data needed to be standardized accordingly. Concerning our findings on the presence of outliers, we decided to use the robust scaling method proposed by Cao et al. (2016), primarily because it can deal with outliers very well.

3.2.3 Classification Tasks

In Zloch et al. (2020), we sought to investigate measure importance (1) with respect to all available categories, and (2) with respect to one category in binary classification. Supplementary to the description of the experimental setup in the publication, Figure 3.2 illustrates the complete process pipeline to accomplish our goals in the two classification tasks described in Zloch et al. (2020). To recap:

Task 1. In the first classification task, we set up, trained, and tuned one classification model to predict one out of all available categories (i.e., knowledge domains). Thus, we investigated measure importance in terms of the classifier's capacity

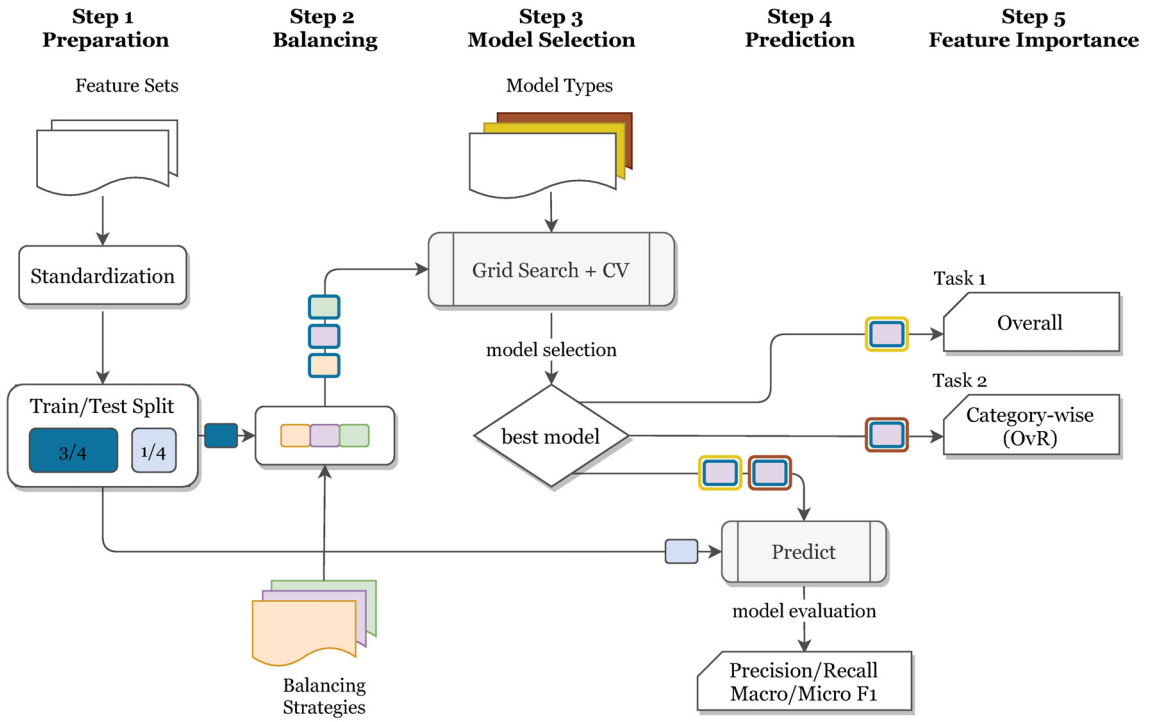


Figure 3.2: **Experimental setup** for determining (a) overall and (b) category-wise graph measure importance.

to discriminate all knowledge domains from each other by using graph measures as features, that is, structural measures of the RDF knowledge graph topology.

Task 2. In the second classification task we sought to find the most important measures to describe one particular knowledge domain. Graph measures with the best performance have the ability to characterize datasets within one particular category most effectively. For each individual knowledge domain, we investigated this by training one independent classifier per knowledge domain. This is done by employing the binary relevance method, also known as the *one-vs-rest* (OvR) version of the first classification task (Pedregosa et al., 2011).

The experimental setup can be split into five individual steps. In the following, we describe each step in detail.

1. Preparation. First, we performed feature engineering tasks like correlation coefficient analysis and feature selection, as described in Zloch et al. (2020). By this means, we obtained two different sets of features from this step, that is, the *full* set of features and the *meaningful* ones only. We then standardized (and scaled) the input data according to previous investigations on the distribution of values in the data, as described in Section 3.2.2. We experimented with different scaling implementations and decided to use the robust scaling method proposed by Cao et al. (2016). This

Table 3.1: **Parameters for grid search** that we used to tune each of the instantiated classifiers (see Step 3, *Model Selection*).

Parameter	Short description	Value range
`criterion`	Measure by which the splitting of nodes is based in the trees	entropy, Gini
`max_depth`	Maximum depth of the trees	$2, 4, 6, 8, 10, 12$
`max_features`	Maximum number of features to consider for split criteria	[4..25]
`min_samples_leaf`	Number of samples per tree leaf	$1, 3, 5, 7, 9$
`min_samples_split`	Number of samples per tree node	$2, 4, 6, 8, 10, 12$
`n_estimators`	Number of decision trees in the forest	300

method reduces the negative effects of outliers (if present). After that, we split the data into 75% for training and 25% for testing.

2. Balancing. As mentioned in Section 3.2.2, our sample of knowledge domains is not balanced. In order to avoid overfitting the model concerning datasets from the majority class, that is, Linguistics (see Figure 3.1), in this step, the training data was subjected to three balancing strategies: we experimented with undersampling, oversampling techniques and with leaving the training data unbalanced by not applying any balancing strategy. For undersampling, we randomly sampled from all classes except the minority class. For oversampling, we used the SMOTE algorithm to systematically generate synthetic datasets for all categories except the majority class. As a result of this step, we obtained three balanced training datasets for each of the two feature sets.

3. Model Selection. Note that our main aim was to understand overall and category-wise graph measure importance, rather than finding the best prediction model type for predicting category labels of knowledge graphs. However, we were obliged to find meaningful results. Thus, for each balancing strategy obtained from Step 2, we instantiated the classifiers in different variations: (i) a basic random forest, (ii) a random forest with a stratified sampling of categories, and (iii) a one-vs-rest (OvR) binary classifier with an instantiated basic random forest. The difference between (i) and (ii) is that the latter respects a balanced subsample concerning the available categories to be predicted for each decision tree that gets evaluated.

All classifier instances were subjected to hyperparameter tuning via grid search with five-fold cross-validation (Pedregosa et al., 2011). Grid search finds the best constellation of hyperparameter values for a given model instance and training dataset to ensure that the model that best fits the data will be found. Table 3.1 lists the parameters that we tuned and their corresponding range of values. Each classifier is instantiated with a fixed number of 300 decision trees in the "forest".

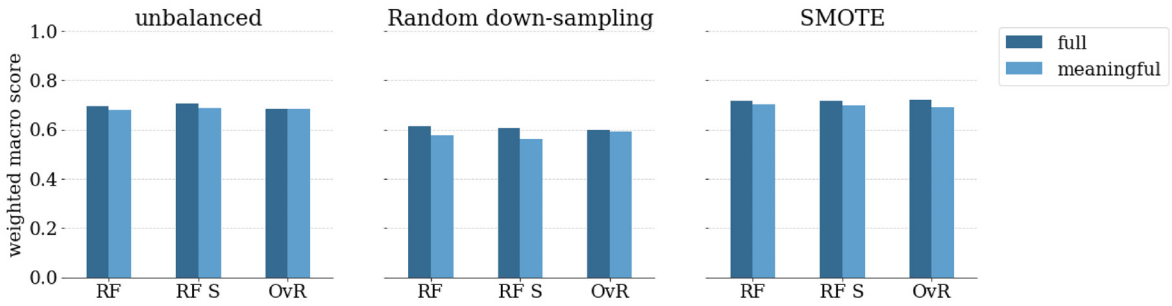


Figure 3.3: **Prediction model performance** represented by the *weighted macro F1-score*. The figure shows values averaged over 10 prediction attempts for the different models instantiated with different balancing strategies and feature sets. SMOTE = synthetic minority over-sampling technique; RF = Random Forest; RF S = Random Forest Stratified; OvR = one-vs-rest.

As a result of this step, we obtained one tuned model for each of the tasks. We could now use these models to extract feature importance scores and to validate the prediction performance.

4. Prediction and 5. Feature Importance In Step 1, we split the test dataset from the original dataset, so that it did participate in the learning process. The test dataset was now used to predict the categories of the unseen datasets via the selected models obtained from Step 3. Finally, we evaluated the model by computing validation measures, such as precision, recall, and micro/macro F1.

In order to obtain more robust results, we repeated the described learning process ten times and averaged the validation scores accordingly. Please find the validation scores in Zloch et al. (2020).

3.2.4 Additional Results & Discussion

In Zloch et al. (2020) we report on the classification performance of the classifier one-vs-rest (OvR) with random forest, as we were primarily interested in category-wise classification performance. In addition to that, Figure 3.3 shows the prediction performance for all the instantiated models for 10 prediction attempts. The figure shows the *weighted macro F1-score*, averaged over 10 prediction attempts following the setup described in Figure 3.2. To clarify: the macro F1-score is obtained by averaging the classifier's F1-scores per category of the test dataset. The *weighted* macro F1-score gives each category an additional weight while building the average value by respecting the number of seen samples in that category of the test dataset.

We would like to address two observations about the results shown in Figure 3.3. The first is that the scores for the full set of features (dark-blue columns) are always above those of the (limited) meaningful set of features (light-blue columns), except for the unbalanced OvR case, where it is almost equal (far left-hand sub-figure, two

Table 3.2: **Evaluation metrics** for the best prediction model, i.e., the tuned OvR model with random forest, instantiated with the *meaningful* feature set and *SMOTE* as balancing strategy. *Note.* For the experiments, we removed categories with too few samples (see Zloch et al. (2020)). Prec = precision; rec = recall.

Knowledge Domain	**Prec.**	**Rec.**	**F1-Score**	**Support**
Cross-Domain	0.250	0.250	0.247	4
Geography	0.000	0.000	0.000	3
Government	0.649	0.611	0.628	9
Life Sciences	0.758	0.975	0.852	8
Linguistics	0.892	0.806	0.847	31
Publications	0.571	0.666	0.615	12

columns). As we applied standard feature selection methods to obtain a meaningful feature set before training the models (see Zloch et al., 2020), we expected the models employing this set to score (considerably) higher than the other models employing the whole set of features. This was not the case.

The second observation is that models employing datasets balanced using the upsampling strategy (far right-hand sub-figure in Figure 3.3) performed similarly to models that employed unbalanced datasets, although the former had slightly higher scores. Models employing datasets that were downsampled performed worst. This was apparently due to the fact that there were too few samples per category during training.

Both observations are consequences of the model type chosen for the tasks (i.e., random forest), which we used for classification and feature importance extraction. Random forest averages prediction performance over the number of decision trees to be created, each fitting *different subsamples* of the training dataset, including a random feature set for each of the created samples. Thus, concerning both observations, the model was robust against the actual set of features employed. It seems that the outliers that we observed in single features in our dataset (see Section 3.2.2) did not have significant impact.

In Zloch et al. (2020), we used the best model obtained from Step 4 to extract category-wise feature importances, which employs the *meaningful* feature set with the *SMOTE* balancing strategy. To give an impression of the actual precision, recall, and F1-measure values for this model, Table 3.2 shows averaged evaluation metric values over 10 predictions attempts. The far right-hand column, *Support*, gives the number of samples in the corresponding categories given in the test set. Please note that samples in the test set are not balanced over the categories, as the data were split before balancing (see Step 1 in Figure 3.2). The test dataset with which the model was tested thus contains only real-world samples and no artificial samples.

The model reaches the best prediction performance for test samples of knowledge

graphs in the *Life Sciences* and *Linguistics* domains, with F1-scores of 0.852 and 0.847, respectively. The exceptionally high recall value of 0.975 for test samples in the *Life Sciences* category is striking. It means that approximately 97% of all test samples in this category were correctly classified. The precision value is 0.758, which means that approximately 76% of the predicted samples were correctly classified. The *Linguistics* category had the most samples (31) to predict. Here, the classifier achieved a precision value 0.892 and a recall value of 0.806. The model performed less well on knowledge graph samples from the *Government* and *Publications* domains, with F1-scores of 0.628 and 0.615, respectively. In both categories, less than two-thirds of the samples were correctly classified.

The model performed very poorly on samples predicted for the categories *Cross-Domain* and *Geography*. For the 10 prediction attempts, none of the three samples in *Geography* was correctly classified, whereas in the *Cross-Domain* category, at least one of four samples (0.25) was classified correctly. The main reason for this is probably that the number of distinct training samples available in these categories was too low. Although the training dataset was balanced with *SMOTE*, it is a synthetic upsampling method, and is not powerful enough to emulate real samples (see our future work plans on this in Section 5.2.2).

Caveats & Limitations

One limitation of our approach to determine category-wise measure importance described in Zloch et al. (2020) may be that we exclusively employed the random forest classifier and did not experiment with other implementations of classifiers, for example, logistic regression (LR) or support vector machine (SVM). Hence, the numbers reflect measure importance determined by this particular model type. However, random forest is one of the most popular and robust classifiers, and is suited for many "standard" classification setups (Louppe, 2014). A huge advantage of this classifier is that it is an ensemble method – that is, it averages classification performance over combined set of individual models (i.e., decision trees). Our primary concern was to highlight topological differences in the given knowledge domains, not to find the best model for predicting category labels for knowledge graphs. In future work, we want to tackle the bad prediction performance observed in the categories *Geography* and *Cross-Domain* by acquiring more samples (see Section 5.2.2) to re-run the experiment with and to validate our findings. As future prospect, we want to investigate results of the other classifiers mentioned.

4

Towards an Application-Specific RDF Benchmarking Suite

Chapter 2 introduced a software framework enabling us to perform large-scale analyses on the topologies of RDF knowledge graphs. To achieve this, the framework represents a knowledge graph topology as a programmatic graph object. We now want to further benefit from the framework's abilities and to address the third objective of this thesis, which is to leverage these programmatic graph representations to guide and support solutions in related research areas (Zloch et al., 2020).

This chapter takes on the benchmarking use case mentioned in Section 1.2 and introduces two approaches for the creation of custom- and application-specific benchmarks for RDF data. Central to this chapter are two contributions, one of which has already been published, and the other is a work in progress.

Section 4.1 gives a brief introduction to the use case of RDF benchmarking and the recent challenges in this research field concerning query runtime evaluation. We will address this in the two subsequent sections. Section 4.2 describes another functionality of our framework, that is, to utilize graph representations from knowledge graphs to generate customized benchmark queries. Section 4.3 presents our second approach to application-specific RDF benchmarking. The corresponding publication introduces a "business-use-case"-driven approach for query runtime evaluation on state-of-the-art RDF data storage solutions (Zloch et al., 2017).

4.1 Application-Specific Benchmarking

The design of scalable data storage solutions – not only for RDF data – is based on benchmarks suites. Benchmark suites assess a data storage solution's performance with regard to a variety of community-driven use cases and procedures, such as browsing and exploration, business intelligence, reasoning, querying/updating the data, and testing the coverage of features in the query language specification. In the past, business applications had similar requirements, and data had a similar shape. Thus, *domain-specific* benchmarks have been the source of truth for database engineers and designers, and have driven the development of efficient data stores.[12]

However, the Web has become more data-oriented, with an enormous amount of data being collected, prepared, and made available to researchers and the public. Publicly available RDF knowledge graphs are primarily *application-specific* and diverse in their structure (Zloch et al., 2020). This challenges existing domain-specific benchmarking approaches concerning the delivery of meaningful and reliable results that reflect real-world situations (Seltzer et al., 1999).

The assessment of RDF data storage solutions with domain-specific logics and synthetic data of traditional benchmarks can be problematic, primarily for two reasons. First, the flexibility that RDF offers: RDF is designed to be data-oriented, and thus it is inherently flexible regarding vocabulary usage. Compared with the highly structured synthetic data generated by traditional benchmarks, real-world LOD have weak structure (Duan et al., 2011; Saleem et al., 2015b). The second reason is the impedance mismatch problem – that is, the problem of bridging the gap between the native graph model that RDF comes with and the internal storage model of a data store, which is most likely a relational database. There are several strategies to implement a graph model in a relational schema (Aluç et al., 2014b; Bornea et al., 2013; Erling, 2012; Wilkinson et al., 2003). Paired with synthetic queries, which may not necessarily reflect real-world situations, the evaluation of query runtimes of RDF benchmarks may not be meaningful enough and equally meaningful for two different applications (Aluç et al., 2014a).

For this reason, research on RDF benchmarking is favoring *application-specific* benchmarks that employ data and queries from real-world applications and do not generate data synthetically. Application-specific benchmarks in RDF employ particular datasets and vocabularies to design benchmarks for particular use cases. Queries generated by this type of benchmarks are based on query logs, for example, in order to more accurately mimic real-world situations.

[12]DB-Engines Ranking. `https://db-engines.com/en/ranking`. Last accessed on October 21, 2020.

Contributions

In the context of this thesis, we present two contributions to the development of application-specific benchmarks.

- In Section 4.2, we address the issue of synthetic queries used by RDF benchmarks. We propose a flexible query instantiation mechanism leveraging graph representations of real-world knowledge graphs. A corresponding implementation is shipped with our graph-based framework introduced in Zloch et al. (2019). It can generate queries of the types that are frequently found in query logs. This contributes to generating query loads based on real data.
- In Section 4.3, we propose a method that enables to evaluate real-world query results more reliably. In the corresponding paper (Zloch et al., 2017), we show that, in situations where we want to find a suitable data store for our data, it is advantageous to look at application-specific *business use cases* to detect a suitable data store candidate.

4.2 Query Generation for Application-Specific Benchmarks

In this section, we propose a simple and flexible variant of a query instantiation mechanism that is based on programmatic graph representations from knowledge graphs to produce customized benchmarks for query evaluation. We integrate the functionality into our software framework introduced in Zloch et al. (2019) (see Chapter 2). This allows to generate any number of query instances out of the box, after knowledge graph acquisition and preparation. To create a first demonstration of its feasibility, we include a comprehensive list of query templates introduced by the *WatDiv* benchmark, which is a state-of-the-art RDF benchmark that contains a list of real-world query templates (Aluç et al., 2014a). By this means, our approach can generate over 90% of state-of-the-art queries frequently found in today's query logs of RDF data stores. (Bonifati et al., 2017).

The next section, 4.2.1, aligns related work with our approach. The approach itself is described in Section 4.2.2. As this is unpublished work in progress, Section 4.2.3 concludes by mentioning current caveats and limitations.

4.2.1 Related Work

For RDF data, the overall performance of a data store depends on three major factors: (1) the *characteristics* of the data; (2) the implementation of the *data model* in the *database schema*, also known as the storage strategy or physical design; and (3) the *queries* themselves. The structure of the queries, and the strategy with which they

are evaluated, play an essential role in runtime performance evaluations (Zloch et al., 2017).

To measure whether benchmark queries appropriately represent patterns and types of queries observed in real-world systems, the literature introduces several metrics that measure *structural* and *data-driven* feature variability in SPARQL queries (Aluç et al., 2014a; Gallego et al., 2011; Möller et al., 2010; Picalausa and Vansummeren, 2011). Examples of such metrics include the query type, that is, `SELECT`, `CONSTRUCT`, `ASK`, or `DESCRIBE`; the number of triple patterns; the number of join patterns (*star*, *path*, *hybrid*, *sink*); mean degree of join vertices in graph patterns in a query.

Many works consider the aforementioned prevalent *join pattern* of basic graph patterns in SPARQL queries (Harris and Seaborne, 2013) to be an important aspect that influences the performance (Aluç et al., 2014a; Görlitz et al., 2012). Related work considers four different types: *star*, which has multiple outgoing links but no incoming links; *path*, which has precisely one incoming and one outgoing link; *hybrid*, with at least one incoming and outgoing link; and *sink*, the vertex type with multiple incoming links but no outgoing links. Existing data stores behave very differently when evaluating such join patterns that they encounter in queries, as they have different strategies to decompose and map query graph patterns to the internal graph representation.

Saleem et al. (2015a) analyzed logs from four large SPARQL endpoints, involving a total of 1.2 billion triples from DBpedia, Linked Geo Data, Semantic Web Dog Food, and the British Museum. They introduced different characteristics for the observed queries, classified them, and reported on each class's usage statistics. They found that 66% of the queries that they analyzed on the four logs, contained no join vertex type at all, and that 33% of all queries could be considered queries of the type *star*. This situation may be different for other datasets and endpoints, as queries are highly dependent on the underlying vocabulary used in the dataset, and thus may be more diverse (Duan et al., 2011; Gallego et al., 2011).

To increase the diversity of queries in RDF benchmark suites, Aluç et al. (2014a) propose a flexible synthetic data and query generator suite based on a data description language that covers a large number of the above-mentioned query features. Their *Waterloo SPARQL Diversity Test Suite* (WatDiv) provides 20 query templates of four different categories: *linear*, *star*, *snowflake*, and *complex*. They vary in their number of triple patterns and overall complexity. *WatDiv*, however, requires to generate a synthetic dataset first in order to instantiate the queries. Further, RDF predicates in triple patterns of SPARQL query templates are fixed and cannot be instantiated dynamically.

Saleem et al. (2015b) developed *FEASIBLE* to overcome the issue of benchmark suites to generate synthetic queries. The authors claimed that *FEASIBLE* can construct queries of all four query types (i.e., `SELECT`, `CONSTRUCT`, `ASK`, and `DESCRIBE`), all based on real data. They performed a comprehensive study with four other popular benchmarks, constructing queries of the types mentioned. They demonstrated that

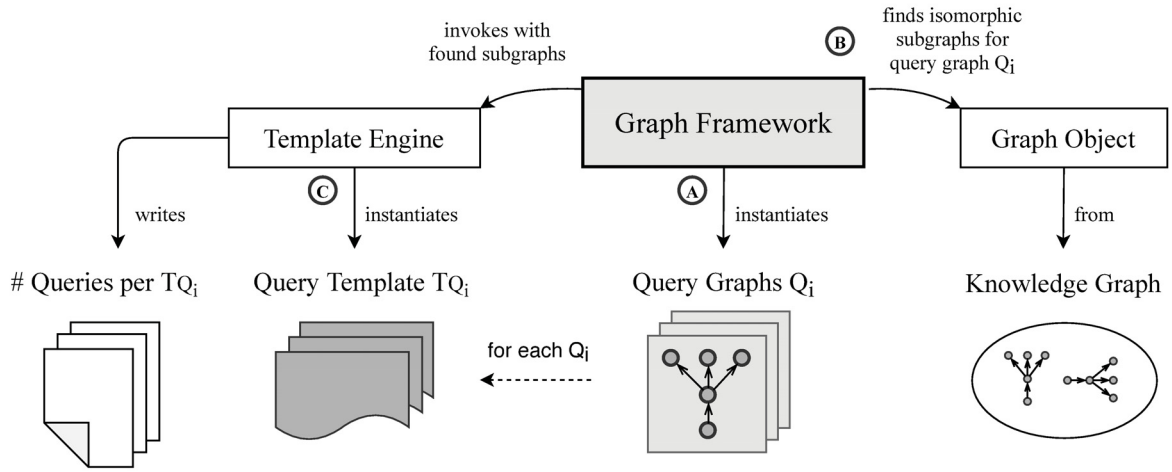


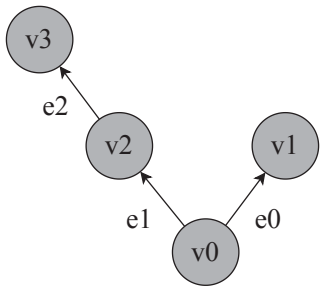
Figure 4.1: **Finding isomorphic subgraphs** for query generation by means of a template engine. The whole process pipeline is illustrated.

data stores perform differently when queried with queries based on real data, and that this affects the decision about a suitable data store for a real RDF dataset (Zloch et al., 2017). The authors proposed a query generator based on query logs from SPARQL endpoints. However, query logs cannot be accessed easily, as data provided deny the access to their logs.

4.2.2 Query Generation by Finding Graph Isomorphisms

Regarding the importance of join patterns for the runtime evaluation of data stores mentioned in Section 4.2.1 and their application in existing solutions, such as *FEASIBLE* and *WatDiv*, we propose a query generation approach based on real data from knowledge graphs and query templates. The templates are built from query structures observed in query logs of data stores in production, and instantiated from real-world data found in today's knowledge graphs. As the generated queries are based on templates from real queries, they will have the corresponding characteristics as well as the desired variability and diversity. In contrast to other works, our approach is more flexible as it does not require any data to be loaded into any RDF store beforehand. Also, the template-based approach enables benchmark designers to vary SPARQL queries to the desired extent, which enables them to address specific features to be tested. The integration into our graph-based framework (Zloch et al., 2019) allows to generate any number of query instances directly after knowledge graph acquisition and preparation.

Our approach works by representing SPARQL queries themselves as directed labeled graphs, similar to existing approaches (Aluç et al., 2014a). To eventually generate queries from these representations, we leverage the mechanism of finding isomorphic subgraphs in the original knowledge graphs. Graph isomorphism detection is well known in graph databases (Lee et al., 2012) and has also found application for query



(a) An example query graph having a *linear* join pattern.

```
SELECT ?v0 ?v2 ?v3
WHERE {
  ?v0 {{ e0 }} {{ v1 }} .
  ?v2 {{ e2 }} ?v3 .
  ?v0 {{ e1 }} ?v2 .
}
```

(b) A query template that is instantiated with a subgraph match of the query graph shown on the left.

Figure 4.2: **Example query graph (a) and corresponding query template (b)** taken from the *WatDiv* benchmark suite (Aluç et al., 2014a; query label `l1`).

processing in RDF (Kim et al., 2015). Concerning our goal of generating queries, the problem can be defined more precisely as follows: Given a query graph Q_G, we seek to find all subgraphs in our original graph G that are isomorphic to Q_G. We call these isomorphic subgraphs *matches* of Q_G in G.

Figure 4.1 illustrates the SPARQL query generation process for the found matches, which is described in detail in the following. Let I be the set of all types of queries, for example, $I = \{star, linear, ..\}$, introduced in Saleem and Ngonga Ngomo, 2014. First, each query type i is instantiated as a query graph Q_i by the framework (A in Figure 4.1). An example query graph instance of the join pattern *linear* is shown in Figure 4.2a. Our framework finds the desired number of isomorphic subgraphs in the original knowledge graph (possibly prepared and instantiated itself by the framework), for each of the given query graphs (B). To achieve this, we employ methods provided by the *graph-tool* library (Peixoto, 2014), which we already employed for graph instantiation and graph measure computation in Zloch et al. (2019). Therefore, all query graphs need to be available beforehand as graph objects, for example, in individual files.

Finally, to generate the final queries from the found subgraphs we employ a template engine. A template engine works by passing a data structure (also called *model*) to a string-based *template*. The template engine then instantiates the template by filling in variable placeholders around static content placed in the template. Our templates are SPARQL queries that may contain variable placeholders in any part of a basic graph pattern, that is, the subject, predicate, or object. The data model passed to the template is thus one single isomorphic subgraph found in the original graph. An example SPARQL query template that we employed from the *WatDiv* benchmark suite is shown in Figure 4.2b. Consequently, for each of the query graphs there must exist a corresponding query template T_{Q_i} (below C).

After finding the desired number of subgraphs for each query graph Q_i, we pass each subgraph to the template engine, which in turn passes it as a data model to the

corresponding query template (C). The queries are then written to disk accordingly. The desired number of query graphs and number of subgraphs to find, as well as the number of queries per query graph to generate is configurable at the time of execution of the framework.

We tested our design and implementation on queries from the *Waterloo Diversity Benchmark Suite* (WatDiv), a state-of-the-art benchmark suite (Aluç et al., 2014a). All queries provided by *WatDiv* are implemented as query graphs and query templates in the submodule `query.watdiv` of our framework (see Section 2.2).

Please refer to list items 3 and 4 in Section 5.2.3 for our future work plans for this feature.

4.2.3 Caveats & Limitations

Although the proposed approach for query instantiation is part of our framework and its latest releases (Zloch, 2020), the feature has unfortunately not yet been published. Preliminary tests showed that all queries from *WatDiv* that we integrated into the framework can be instantiated and return reliable results when executed against data loaded into a data store. This has been tested on various knowledge graphs acquired from the LOD Cloud acquired in Zloch et al. (2019). However, for a publication, a robust evaluation of the generated queries concerning their variability and a demonstration of their feasibility are lacking.

During our preliminary tests, we encountered high runtimes when generating queries of the type *complex* (Aluç et al., 2014a). This was in fact due to the complexity of the queries and the resulting query (sub)graphs for which a match needed to be found in the original knowledge graph. Additionally, our programmatic knowledge graph representations are highly optimized to lower hard disk and memory consumption. To this end, vertices and edges labels are encoded by employing a non-cryptographic hashing strategy (Zloch et al., 2019). After finding the corresponding subgraph matches, we thus need to decode the vertices and edges of the subgraph using the original RDF dataset. Our framework offers the corresponding functionality for that. However, this is a crucial point.

Speaking of limitations, we can see that in the current implementation of our query templates, variables can be placed only within any part of the basic graph pattern, that is, the subject, predicate, or object of any triple pattern occurring in the SPARQL query. The usage of such variables in other SPARQL query clauses – for example, FILTER – has not yet been tested. Further, although the code is designed to be generic – that is, to merely point to source packages and folders with the corresponding query graphs and query templates – we did not test it to generate queries from other benchmarks.

Section 5.2.3 presents our future plans regarding this feature.

4.3 Towards a Use Case Driven Evaluation of Database Systems for RDF Data Storage

Matthäus Zloch, Daniel Hienert and Stefan Conrad.

In: *The Semantic Web – 16th International Semantic Web Conference (ISWC 2017), Joint Proceedings of BLINK 2017: 2nd International Workshop on Benchmarking Linked Data and NLIWoD3: Natural Language Interfaces for the Web of Data.*. Vol. 1932. CEUR-WS.org. URL: `http://ceur-ws.org/Vol-1932/#paper-08.`

Following up on the application-specific benchmarking setting, the primary concern of this publication was to vary the evaluation strategy of queries when executed against data stores.

Summary

In this paper, we propose a customizable benchmarking approach and framework to evaluate queries against overall query runtime performance, what we call the *standard benchmark approach*, and a novel *use-case-driven approach*, where query groups from application-specific use cases are respected. The overall hypothesis is that such groups have similar structural characteristics. This means that they (a) target similar (disjunct) parts of a domain model (and thereby the database schema), and (b) stress query evaluation techniques offered by the data stores in different ways (highly structured vs. plain tabular).

Our approach is to compare results for query runtimes for a comparably large number of data storage solutions of different types (triple stores, relational and graph-based databases, column- and row-stores) and real data from a productive web application. For the experiment, we transformed all data and the corresponding queries into the format and language required by the database solutions we evaluated – for example, N-Triples and SPARQL for RDF stores, and edgelists and Cipher for the graph-based database Neo4j. All transformed data and translated queries are available for further reuse.[13]

Based on the two approaches – that is, the use-case driven and the classical "non use-case driven" approach – we developed a configurable, extendable, and property-based query evaluation framework in order to automatically run query runtime evaluations. By this means, one can compare query runtimes for the configured number of data stores and make a decision about a suitable solution for one application. The framework is further able to generate query load sequence mixtures of use-case-relevant queries and non-relevant queries, for the ratios of 100-0 (use-case-only queries) and 50-50 (half of the queries are use-case-specific), beforehand.

[13] `https://github.com/mazlo/blink17`. Last accessed on October 29, 2020.

Importance and Impact on This Thesis

This publication represents the foundation for the ideas subsequently introduced in Zloch et al. (2019) and Zloch et al. (2020). After observing the different behavior of data stores concerning query runtimes of query group loads in comparison with considering all queries, our aim was to find correlations between characteristics adhering to the actual data, features of the used queries, and the corresponding query runtimes. Our aim was to derive implications from that in order to possibly predict a suitable data store by observing particular characteristics in the data beforehand. In this sense, the work done in this publication had a significant impact on the subsequent works described in Chapter 2 and Chapter 3.

Author Contributions

The first author, Matthäus Zloch, programmed the parameterizable framework to evaluate both query runtime evaluation approaches. He created the experimental environment, mapped and transformed all data and queries into the type and language required by the target data store, and performed all experiments. Matthäus Zloch also wrote most of the content of the paper and created all the figures.

Daniel Hienert advised Matthäus Zloch during the experiments, and contributed to analyzing the results. He proposed the inclusion of further data stores for evaluation. Daniel Hienert also contributed to the content in Sections 4 and 5.

The paper underwent proofreading by all authors. The whole work was under the supervision of Daniel Hienert and Stefan Conrad.

5

Conclusion and Future Work

This chapter concludes with a summary of the achievements made during the Ph.D. studies and provides some ideas and directions for future work.

The main objective of the thesis was to facilitate graph-related tasks on RDF datasets, to study graph measure effectiveness and importance, and to find possible applications of our previous findings in related research areas. We provide use cases and research areas that can benefit from the knowledge about the topological structure(s) of individual knowledge graphs as well as from knowledge-domain-dependent knowledge graphs. We report on the topological structure of a large number of real-world datasets and investigate measure meaningfulness. In the final step of the thesis, we take on the benchmarking use case and show how to leverage a knowledge graph's programmatic graph representation to generate realistic queries for application-specific benchmarking suites.

During the Ph.D. studies three papers with particular relevance for this thesis were published – two appeared in the peer-reviewed proceedings of international conferences (Zloch et al., 2019; Zloch et al., 2017) and one was published in an international peer-reviewed journal (Zloch et al., 2020).

Along with these papers, we published all code for the developed software, datasets, and scripts under open source licenses on popular code and data hosting platforms, such as GitHub and Zenodo. Both web services provide search interfaces, which makes the code and all results web-findable.

5.1 Summary of Results

We summarize the results by referring to the overall objectives of the thesis and its contributions, which are outlined in Sections 1.2 and 1.3, respectively.

5.1.1 Facilitating Graph-Related Tasks on Knowledge Graphs

Our first objective was to facilitate graph-related tasks on knowledge graphs, such as the large-scale study of graph topologies. Thus, our first contribution is the development of a software framework offering various capabilities to support graph-related tasks on RDF knowledge graphs, such as dataset acquisition; dataset cleaning and preparation; instantiation; and graph measure computation (Section 2.1). The framework has the capacity to deal with typical issues known from public knowledge graph acquisition – for example, various formats, unofficial and unsupported media and file types, and compressed archives. It is designed to support large-scale analyses, particularly when a large number of datasets are analyzed in parallel. In this thesis, we also described in detail the framework's technical architecture (Section 2.2) and the supported graph measures (Section 2.3).

To demonstrate the feasibility of the framework, we successfully used it to acquire a representative sample of 280 real-world knowledge graphs from a popular source of publicly available data, namely, the LOD Cloud. We then conducted a systematic graph-based analysis with regard to the available 54 graph-based measures, and analyzed the topological differences of the acquired graphs in popular knowledge domains. For all the acquired knowledge graphs, we generated topological profiles that are available to the research community for further use (Zloch and Acosta, 2018).

On the one hand, this comprehensive study enabled us to make general observations about the graph-based structure of RDF knowledge graphs. For instance, it showed that, on average, the vertex degree is approximately 8 and that the distribution of vertex degrees in many graphs can be described with a power-law function. This confirms previous findings of related works (Ding and Finin, 2006). On the other hand, we observed characteristics specific to knowledge domains and individual datasets, and assumed a dependency on the employed RDF vocabularies. For instance, in most knowledge domains, the average degree is not affected by the size of the graph (in terms of number of edges).

During this study, we found that not all graph measures are equally meaningful – especially when it comes to describing RDF knowledge graphs effectively and concisely.

5.1.2 Assessment of Graph Measure Effectiveness

Our second objective was to deepen the investigation of graph topologies and their specificities, especially for knowledge graphs within distinct knowledge domains. In particular, we assessed graph measure performance in terms of their capacity to

discriminate knowledge graphs from popular knowledge domains (see Section 3.1). Our assumption was that measures performing well on this task can be considered useful and important for a particular category. By this means, we identified a set of effective and important measures for knowledge graph characterization.

To make the study more comprehensive, we extended our framework with another set of RDF-graph-based measures, which was defined by Fernández et al. (2018), and repeated our graph-based analysis with all knowledge graphs acquired in the previous study (Zloch et al., 2019). From the initial set of 54 graph-based measures, we identified 29 measures that are effective, distinct, and meaningful. From this set, 13 measures have the capacity to discriminate dataset categories with particular impact. The majority of the measures are RDF-graph-based measures. To determine graph measure importance in the individual knowledge domains, we employed a state-of-the-art machine learning classification model. Additionally, in this thesis, we described the challenges faced during the experiments, and detailed the performance of the model employed in the experiment (Section 3.2).

We concluded our graph measure investigation and assessment with two aspects that shape the prevalent structure of knowledge graph topologies: (1) the characteristics that adhere to knowledge graphs modeled in RDF in particular, as the topology of the graphs differs from that of other types of graphs (e.g., social graphs); and (2) the compliance with a (knowledge-domain-dependent) standardized RDF vocabulary, as the vocabulary shapes the way in which data are modeled, thereby leading to similarities in the graph topologies (see Zloch et al., 2020).

5.1.3 Leveraging Graph Representations for Other Tasks

Finally, we took on the benchmarking use case and provided two contributions to the generation of application-specific benchmarking suites. Both contributions provide approaches against tailored benchmarks suites that prefer to consider synthetic data and query generators, which cannot reproduce the observed variability of real-world SPARQL queries.

First, we proposed an approach to leverage graph representations programmatically to facilitate the generation of application-specific benchmark query loads (Section 4.2). Our software framework provides a simple yet flexible template-based implementation and integrates SPARQL query templates from the *WatDiv* benchmark suite (Aluç et al., 2014a) as a proof of concept. Aluç et al. (2014a) claimed that the employed queries exhibit the required variability that synthetic queries are lacking, and that they are based on query structures observed in real-world query logs. The integration of this feature into our framework (Section 2.1) provides a designated code base for the out-of-the-box generation of queries after knowledge graph acquisition and preparation. In preliminary experiments, we successfully generated all types of queries provided by the *WatDiv* benchmark for several exemplary knowledge graphs. Unfortunately, as this is

a work in progress, it has not yet been published.

Our second contribution in the context of RDF benchmarks is our open source framework facilitating application-specific benchmarking of RDF data. The related publication considers a use-case-driven approach to evaluate query loads executed on different RDF data stores (Zloch et al., 2017). To this end, we employed an RDF data model, data, and queries from a real-world application and investigated query runtimes of well-designed query workloads on different types of data stores. We showed that grouping queries according to application-specific use cases partly yields shorter query runtimes, compared with the standard benchmarking approaches, which obtain query runtimes over the total set of queries.

5.2 Future Work

Based on the results of each of the publications, we derived plans for the future directions of our research in order to further strengthen our achievements and to propose improvements in related research fields. The ideas for future work outlined below go beyond the propositions stated in the individual publications. Some of the items mentioned originated from discussions in the peer-review processes of our submitted papers.

5.2.1 Graph-Based Framework

Multiple improvements to the software framework are currently undergoing or planned.

1. **Semantic attributes in graphs**. So far, a graph representation represents an RDF knowledge graph as a simple structure of vertices and edges. However, vertices and edges of RDF graphs contain semantics. Finding vertices of a certain type (e.g., subjects or objects), filtering, and building subgraphs, involves time-consuming iterations over the set of edges or vertices. Therefore, in order to facilitate the aforementioned tasks, we want to add RDF-graph-specific attributes to the graphs' vertices and edges during the graph preparation. Attributes attached to edges could include the edge type, such as *relationship* or *attribute*, indicating whether the edge connects another RDF resource or RDF literal. Attributes attached to vertices could, for instance, include the *join vertex type* (i.e., *star*, *path*, *hybrid*, *sink*, or *simple*) or the *join vertex degree* (Saleem and Ngonga Ngomo, 2014).

2. **Extensions to sampling functionality**. RDF graph sampling is an active research field in the Semantic Web community. Our framework supports the sampling of vertices and edges in a very basic form (see Section 2.2). We want to extend the provided feature to implement more comprehensive sampling strategies, particularly taking into account the above-mentioned semantic enrichment of vertices and edges. With that in mind, it should be possible to generate subgraphs

containing vertices and edges of a certain type, for example. In the long term, we want to add a feature to sample from large graphs, respecting the "proportions" and ratios of particular graph measures.

3. **Further extensions**. We want to add support for the obtained results from our investigation of effective graph measures – for example, to add further command line parameters, such as `--meaningful` and `--category`, in order to limit the number of measures and focus on the meaningful ones upfront, which are essential for a particular knowledge domain. This will save time during graph measure computation.

5.2.2 Investigations on Graph Topologies

Regarding investigations of knowledge graph topologies, we aim to reach out in the following directions:

1. **Extend sources for knowledge graph acquisition**. Using our framework, we aim to conduct further studies to investigate topological aspects of knowledge graphs. So far, we have considered dataset dumps of knowledge graphs only (Zloch et al., 2019). Moreover, in order to find additional samples for small categories (e.g., *Cross-Domain* and *Geography*), and to validate our findings afterwards, we also want to acquire knowledge graphs from other sources, for example, public SPARQL endpoints. Additionally, we want to analyze other types of graphs, such as social graphs and retweet networks, in order to compare them with non-RDF datasets and to extend and deepen our knowledge about the unique structure of RDF graphs.

2. **Comparison of synthetic dataset designs**. Synthetic RDF dataset generators claim to be compliant with real-world datasets. This is true for the employed RDF vocabularies, but not necessarily for the topological structures they generate. We aim to empirically investigate the ability of existing dataset generators to follow statistical distributions of graph measures that we observed in popular knowledge domains. Further, as dataset generators are designed to scale up, we aim to investigate the graph-evolution irregularities concerning particular graph measures that occur during upsampling. All of the popular synthetic dataset generators for RDF data – for example, BSBM (Bizer and Schultz, 2009), Sp2Bench (Schmidt et al., 2010), LUBM (Guo et al., 2005) – will be subjected to this investigation. As a result, we aim to propose an RDF graph generator that is able to take as input – for example, a knowledge domain and a graph size – and generate a graph with similar properties to the ones in the LOD cloud.

3. **Evolutionary and qualitative aspects of graph topologies**. Data quality assessments of knowledge bases are an evolving research area. A central aspect of

data quality is the congruent use of the vocabulary in a knowledge base regarding the schema definition (Bobed et al., 2020). As vocabulary usage significantly impacts a graph topology's shape, we plan to align our graph measures with quality metrics for vocabulary usage obtained with other tools. This will allow to study graph measure importance from a different angle and can shed light on possible correlations, which can be broken down to the level of knowledge domains. Consequently, we will be able to derive implications for our measures and underline or revise our statements about measure importance in the studied categories.

We also plan to study evolutionary aspects in terms of structural growth of different versions of a knowledge base. Our framework offers a platform to monitor and investigate the evolution of knowledge base inter-linkage, structural growth, and the assessment of qualitative metrics. These are primarily data-driven tasks and have become a continuous challenge for researchers (Bobed et al., 2020; Rashid et al., 2019). Ongoing work is to investigate changes to the topology of a large co-citation graph from Springer Nature.[14]

5.2.3 Application-Specific RDF Benchmarking

1. **A data store recommender**. Query runtime estimation is a key feature of query evaluators that reside within a data store. Recent efforts focus on training predictive models to predict the performance of SPARQL queries (Hasan, 2014; Zhang et al., 2016). Such approaches aim to represent a SPARQL query as a feature vector (respecting the query characteristics) and to employ known query runtimes from query logs.

   Besides the effect of data volume, query runtime performance is affected mainly by the data model and the index structures created by data storage solutions. Thus, we propose to also employ advanced characteristics of the data (i.e., the graph topology) to improve prediction models for query runtimes. Building on this, we aim to create a data store recommender that predicts a suitable data store based on a given dataset and a group of queries (Zloch, 2016). Such a recommender is beneficial for users (researchers, developers, database engineers) who need to choose one data storage solution prior to the storage of data. A comprehensive training dataset can be built from real-world queries generated with the approach described in Section 4.2. Accordingly, query runtimes can be evaluated with the proposed framework described in Section 4.3.

2. **General-purpose query groups**. In the experiment described in Section 4.3, we used data, queries, and business use cases from *one* real-world application, and distributed queries across the groups *Navigation*, *Statistics*, *Validation*, and

[14] SN SciGraph, a Linked Open Data platform for the scholarly domain. `https://www.springernature.com/gp/researchers/scigraph`. Last accessed on October 26, 2020.

User Query. These use cases are specific to the application data we used for the experiment. Although queries are highly individual to an application, and to the data model used, investigating a general-purpose mechanism for query group building in future work may be beneficial. To this end, one might investigate clustering mechanisms on query features, for example. With regard to the above-mentioned data store recommender, it might be interesting to investigate how such general-purpose query groups perform, and whether the obtained results are comparable with state-of-the-art benchmark results.

3. **Dataset of exemplary queries**. Regarding the query generation feature of our framework, we want to offer a *dataset of exemplary queries* for the purpose of providing a systematic benchmarking dataset based on real-world RDF datasets and real queries. The generated queries would be tailored to a large number of real-world RDF datasets and made available to the research community. Such a dataset would be beneficial for database engineers and data storage designers to validate and benchmark new solutions, for instance. Further, it would leverage the development of new solutions in the context of data stores, such as the above-mentioned data store recommender.

   The creation of such a benchmarking dataset requires to invoke our query generation module on some RDF datasets, preferably all of the provided 280 datasets that have already been acquired, prepared, and instantiated as graph objects from the LOD Cloud (see Section 2.1).

4. **Extension of query templates**. So far, our framework offers query templates provided by the *WatDiv* benchmark suite (Aluç et al., 2014a), as it is one of the state-of-the-art benchmarking frameworks supporting query types and structures found in real-world queries. Thus, all queries generated are based on these query templates. In the future, we plan to implement further templates from recent investigations on the patterns found in real-world SPARQL queries, such as those found by Saleem et al. (2015a). In addition, to extend the diversity of the queries supported by our framework, we want to add support for other SPARQL query types (e.g., `ASK`-, `CONSTRUCT`) as well as more advanced SPARQL query features (e.g., `OPTIONAL`-, `UNION`, and `FILTER`-clauses).

Knowledge graphs have fundamentally changed the way in which we represent and browse human knowledge. Research on their shape and topological properties has great potential to provide new insights and to improve existing solutions in related research areas and in industry. We hope that we have motivated other researchers and students to engage with the topic, too.

Bibliography

Güneş Aluç, Olaf Hartig, M. Tamer Özsu, and Khuzaima Daudjee (2014a). Diversified stress testing of RDF data management systems. In: *The Semantic Web – ISWC 2014*. Vol. 8796. Lecture Notes in Computer Science. Cham: Springer, pp. 197–212. ISBN: 978-3-319-11964-9. DOI: 10.1007/978-3-319-11964-9_13 (cit. on pp. 38–43, 49, 53).

Güneş Aluç, M. Tamer Özsu, and Khuzaima Daudjee (2014b). Workload matters: why RDF databases need a new design. In: *Proceedings of the VLDB Endowment* 7.10, pp. 837–840. ISSN: 2150-8097. DOI: 10.14778/2732951.2732957 (cit. on p. 38).

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives (2007). DBpedia: a nucleus for a web of open data. In: *The Semantic Web – 6th International Semantic Web Conference (ISWC 2007)*. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 722–735. ISBN: 978-3-540-76298-0. DOI: 10.1007/978-3-540-76298-0_52 (cit. on p. 1).

Mohamed Ben Ellefi, Zohra Bellahsene, John G. Breslin, Julian Szymanski, Elena Demidova, Stefan Dietze, and Konstantin Todorov (2018). RDF dataset profiling – a survey of features, methods, vocabularies and applications. In: *Semantic Web* 9.5, pp. 677–705. DOI: 10.3233/SW-180294 (cit. on pp. 3, 4).

Christian Bizer and Andreas Schultz (2009). The Berlin SPARQL benchmark. In: *International Journal on Semantic Web and Information Systems (IJSWIS)* 5 (2), p. 24. ISSN: 1552-6283. DOI: 10.4018/jswis.2009040101 (cit. on p. 51).

Carlosa Bobed, Pierrec Maillot, Peggyd Cellier, and Ferré Sébastien (2020). Data-driven assessment of structural evolution of RDF graphs. In: *Semantic Web* 11.5, pp. 831–853. DOI: 10.3233/SW-200368 (cit. on pp. 3, 4, 52).

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. SIGMOD '08. Vancouver, Canada: ACM, pp. 1247–1250. ISBN: 978-1-605-58102-6. DOI: 10.1145/1376616.1376746 (cit. on p. 1).

Angela Bonifati, Wim Martens, and Thomas Timm (2017). An analytical study of large SPARQL query logs. In: *Proceedings of the VLDB Endowment* 11.2, pp. 149–161. ISSN: 2150-8097. DOI: 10.14778/3149193.3149196 (cit. on p. 39).

Mihaela A. Bornea, Julian Dolby, Anastasios Kementsietsidis, Kavitha Srinivas, Patrick Dantressangle, Octavian Udrea, and Bishwaranjan Bhattacharjee (2013). Building an efficient RDF store over a relational database. In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. SIGMOD '13. New York, NY, USA: ACM, pp. 121–132. ISBN: 978-1-450-32037-5. DOI: 10.1145/2463676.2463718 (cit. on p. 38).

Leo Breiman (2004). Random Forests. In: *Machine Learning* 45, pp. 5–32 (cit. on p. 29).

Xi Hang Cao, Ivan Stojkovic, and Zoran Obradovic (2016). A robust data scaling algorithm to improve classification accuracies in biomedical data. In: *BMC bioinformatics* 17, p. 359. DOI: 10.1186/s12859-016-1236-x (cit. on pp. 31, 32).

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer (2002). SMOTE: synthetic minority over-sampling technique. In: *Journal of artificial intelligence research* 16, pp. 321–357. ISSN: 1076-9757. DOI: 10.1613/jair.953 (cit. on p. 31).

Dimitar Dimitrov, Erdal Baran, Pavlos Fafalios, Ran Yu, Xiaofei Zhu, Matthäus Zloch, and Stefan Dietze (2020). TweetsCOV19 – a knowledge base of semantically annotated tweets about the COVID-19 pandemic. In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. CIKM '20. New York, NY, USA: ACM, pp. 2991–2998. ISBN: 978-1-450-36859-9. DOI: 10.1145/3340531.3412765 (cit. on p. 10).

Li Ding and Tim Finin (2006). Characterizing the Semantic Web on the Web. In: *The Semantic Web – 5th International Semantic Web Conference (ISWC 2006)*. Vol. 4273. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 242–257. ISBN: 978-3-540-49055-5. DOI: 10.1007/11926078_18 (cit. on p. 48).

Xin Luna Dong (2018). Challenges and innovations in building a product knowledge graph. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '18. London, United Kingdom: ACM, p. 2869. ISBN: 978-1-450-35552-0. DOI: 10.1145/3219819.3219938 (cit. on p. 1).

Songyun Duan, Anastasios Kementsietsidis, Kavitha Srinivas, and Octavian Udrea (2011). Apples and oranges: a comparison of RDF benchmarks and real RDF datasets. In: *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*. SIGMOD '11. Athens, Greece: ACM, pp. 145–156. ISBN: 978-1-450-30661-4. DOI: 10.1145/1989323.1989340 (cit. on pp. 3, 38, 40).

Orri Erling (2012). Virtuoso, a hybrid RDBMS/graph column store. In: *IEEE Data Engineering Bull.* 35.1, pp. 3–8 (cit. on p. 38).

Javier D. Fernández, Sabrina Kirrane, Axel Polleres, and Simon Steyskal (2020). HDTcrypt: compression and encryption of RDF datasets. In: *Semantic Web* 11.2, pp. 337–359. DOI: 10.3233/SW-180335 (cit. on p. 3).

Javier D. Fernández, Miguel A. Martínez-Prieto, Pablo de la Fuente Redondo, and Claudio Gutiérrez (2018). Characterising RDF data sets. In: *Journal of Information Science* 44.2, pp. 203–229 (cit. on pp. 2, 20, 22, 27, 49).

Javier D. Fernández, Jürgen Umbrich, Axel Polleres, and Magnus Knuth (2019). Evaluating query and storage strategies for RDF archives. In: *Semantic Web* 10.2, pp. 247–291. DOI: 10.3233/SW-180309 (cit. on p. 3).

Mario A. Gallego, Javier D. Fernández, Miguel A. Martínez-Prieto, and Pablo de la Fuente (2011). An empirical study of real-world SPARQL queries. In: *CoRR* abs/1103.5043 (cit. on p. 40).

Olaf Görlitz, Matthias Thimm, and Steffen Staab (2012). SPLODGE: systematic generation of SPARQL benchmark queries for Linked Open Data. In: *Proceedings of the 11th International Semantic Web Conference (ISWC'12)*. Vol. 7649. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 116–132. ISBN: 978-3-642-35175-4. DOI: 10.1007/978-3-642-35176-1_8 (cit. on p. 40).

Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin (2005). LUBM: a benchmark for OWL knowledge base systems. In: *Journal of Web Semantics* 3.2, pp. 158–182. DOI: 10.1016/j.websem.2005.06.005 (cit. on p. 51).

Steve Harris and Andy Seaborne, eds. (2013). SPARQL 1.1 query language. https://www.w3.org/TR/sparql11-query. W3C Recommendation. Last accessed on November 12, 2020 (cit. on p. 40).

Rakebul Hasan (2014). Predicting SPARQL query performance and explaining Linked Data. In: *The Semantic Web: Trends and Challenges (ESWC 2014)*. Vol. 8465. Lecture Notes in Computer Science. Cham: Springer, pp. 795–805. ISBN: 978-3-319-07443-6. DOI: 10.1007/978-3-319-07443-6_53 (cit. on p. 52).

Tom Heath and Christian Bizer (2011). Linked data: evolving the web into a global data space. Vol. 1. Synthesis Lectures on the Semantic Web: Theory and Technology 1. Morgan & Claypool Publishers, pp. 1–136. DOI: 10.2200/S00334ED1V01Y201102WBE001 (cit. on p. 2).

Halil Kilicoglu, Dongwook Shin, Marcelo Fiszman, Graciela Rosemblat, and Thomas C. Rindflesch (2012). SemMedDB: a PubMed-scale repository of biomedical semantic predications. In: *Bioinformatics* 28.23, pp. 3158–3160. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bts591 (cit. on p. 1).

Jinha Kim, Hyungyu Shin, Wook-Shin Han, Sungpack Hong, and Hassan Chafi (2015). Taming subgraph isomorphism for RDF query processing. In: *Proceedings of the VLDB Endowment* 8.11, pp. 1238–1249 (cit. on p. 42).

Jérôme Kunegis (2013). KONECT: the Koblenz network collection. In: *Proceedings of the 22nd International Conference on World Wide Web*. WWW '13 Companion. New York, NY, USA: ACM, pp. 1343–1350. ISBN: 978-1-450-32038-2. DOI: 10.1145/2487788.2488173 (cit. on p. 21).

Jinsoo Lee, Wook-Shin Han, Romans Kasperovics, and Jeong-hoon Lee (2012). An in-depth comparison of subgraph isomorphism algorithms in graph databases. In: *Proceedings of the vldb endowment* 6, pp. 133–144 (cit. on p. 41).

Jure Leskovec and Christos Faloutsos (2006). Sampling from large graphs. In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '06. New York, NY, USA: ACM, pp. 631–636. ISBN: 1-59593-339-5. DOI: 10.1145/1150402.1150479 (cit. on p. 4).

Gilles Louppe (2014). Understanding random forests: from theory to practice. PhD thesis. DOI: 10.13140/2.1.1570.5928 (cit. on pp. 29, 36).

Frank Manola, Eric Miller, and Brian McBride, eds. (2004). RDF primer 1.1. https://www.w3.org/TR/rdf11-primer/. W3C Recommandation. Last accessed on August 22, 2020 (cit. on p. 2).

Knud Möller, Michael Hausenblas, Richard Cyganiak, and Gunnar Aastrand Grimnes (2010). Learning from linked open data usage: patterns & metrics. In: *Proceedings of the WebSci10: Extending the Frontiers of Society On-Line*. Vol. 159 (cit. on p. 40).

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). Scikit-learn: machine learning in Python. In: *Journal of Machine Learning Research* 12, pp. 2825–2830 (cit. on pp. 26, 29, 31–33).

Tiago P. Peixoto (2014). The graph-tool python library. http://figshare.com/articles/graph_tool/1164194. Last accessed on November 21, 2020. DOI: 10.6084/m9.figshare.1164194 (cit. on pp. 7, 18, 42).

François Picalausa and Stijn Vansummeren (2011). What are real SPARQL queries like? In: *Proceedings of the International Workshop on Semantic Web Information Management*. SWIM '11. New York, NY, USA: ACM. ISBN: 978-1-4503-0651-5. DOI: 10.1145/1999299.1999306 (cit. on p. 40).

Mohammada Rashid, Marcoa Torchiano, Giuseppeb Rizzo, Nandanac Mihindukulasooriya, and Oscarc Corcho (2019). A quality assessment approach for evolving

knowledge bases. In: *Semantic Web* 10.2 (Special Issue on Benchmarking Linked Data), pp. 349–383. DOI: 10.3233/SW-180324 (cit. on pp. 4, 52).

Paolo Rosso, Dingqi Yang, and Philippe Cudré-Mauroux (2018). Knowledge graph embeddings. In: *Encyclopedia of Big Data Technologies*. Ed. by Sherif Sakr and Albert Zomaya. Cham: Springer, pp. 1–7. ISBN: 978-3-319-63962-8. DOI: 10.1007/978-3-319-63962-8_284-1 (cit. on p. 3).

Muhammad Saleem, Intizar Ali, Aidan Hogan, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo (2015a). LSQ: the linked SPARQL queries dataset. In: *The Semantic Web – The 14th International Semantic Web Conference (ISWC 2015)*. Vol. 9367. Lecture Notes in Computer Science. Cham: Springer, pp. 261–269. ISBN: 978-3-319-25009-0. DOI: 10.1007/978-3-319-25010-6_15 (cit. on pp. 9, 40, 53).

Muhammad Saleem, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo (2015b). FEASIBLE: a feature-based SPARQL benchmark generation framework. In: *The Semantic Web – The 14th International Semantic Web Conference (ISWC 2015)*. Vol. 9366. Lecture Notes in Computer Science. Cham: Springer, pp. 52–69. ISBN: 978-3-319-25006-9. DOI: 10.1007/978-3-319-25007-6_4 (cit. on pp. 38, 40).

Muhammad Saleem and Axel-Cyrille Ngonga Ngomo (2014). HiBISCuS: Hypergraph-based source selection for SPARQL endpoint federation. In: *The Semantic Web: Trends and Challenges – 11th Extended Semantic Web Conference (ESWC 2014)*. Cham: Springer, pp. 176–191. ISBN: 978-3-319-07443-6. DOI: 10.1007/978-3-319-07443-6_13 (cit. on pp. 42, 50).

Michael Schmidt, Thomas Hornung, Michael Meier, Christoph Pinkel, and Georg Lausen (2010). SP2Bench: a SPARQL performance benchmark. In: *Semantic Web Information Management: A Model-Based Perspective*. Berlin, Heidelberg: Springer, pp. 371–393. ISBN: 978-3-642-04329-1. DOI: 10.1007/978-3-642-04329-1_16 (cit. on p. 51).

Margo Seltzer, David Krinsky, Keith Smith, and Xiaolan Zhang (1999). The case for application-specific benchmarking. In: *Proceedings of the The Seventh Workshop on Hot Topics in Operating Systems*. HOTOS '99. USA: IEEE Computer Society, p. 102. ISBN: 0-76950-237-7. DOI: 10.1109/HOTOS.1999.798385 (cit. on p. 38).

Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang (2015). An overview of microsoft academic service (mas) and applications. In: *Proceedings of the 24th International Conference on World Wide Web*. WWW '15 Companion. Florence, Italy: ACM, pp. 243–246. ISBN: 978-1-450-33473-0. DOI: 10.1145/2740908.2742839 (cit. on p. 1).

Rita T. Sousa, Sara Silva, and Catia Pesquita (2020). Evolving knowledge graph similarity for supervised learning in complex biomedical domains. In: *BMC Bioinformatics* 21.1, p. 6. DOI: 10.1186/s12859-019-3296-1 (cit. on p. 4).

Charles Spearman (1904). The proof and measurement of association between two things. In: *The American Journal of Psychology* 15.1, pp. 72–101. ISSN: 0002-9556. DOI: doi.org/10.2307/1412159 (cit. on p. 26).

Michael P. H. Stumpf, Carsten Wiuf, and Robert M. May (2005). Subnets of scale-free networks are not scale-free: sampling properties of networks. In: *Proceedings of the National Academy of Sciences of the United States of America* 102.12, pp. 4221–4224. ISSN: 1091-6490. DOI: 10.1073/pnas.0501179102 (cit. on p. 31).

Andon Tchechmedjiev, Pavlos Fafalios, Katarina Boland, Malo Gasquet, Matthäus Zloch, Benjamin Zapilko, Stefan Dietze, and Konstantin Todorov (2019). ClaimsKG: a knowledge graph of fact-checked claims. In: *The Semantic Web – 18th International Semantic Web Conference (ISWC 2019), Proceedings Part II*. Vol. 11779. Lecture Notes in Computer Science. Springer, pp. 309–324. ISBN: 978-3-030-30795-0. DOI: 10.1007/978-3-030-30796-7_20 (cit. on p. 11).

Kevin Wilkinson, Craig Sayers, Harumi Kuno, and Dave Reynolds (2003). Efficient RDF storage and retrieval in Jena2. In: *Proceedings of the First International Conference on Semantic Web and Databases*. SWDB'03. Berlin, Germany: CEUR-WS.org, pp. 120–139 (cit. on p. 38).

Thomas H. Wonnacott and Ronald J. Wonnacott (1990). Introductory Statistics. 5th Edition. New York: Wiley. ISBN: 0-471-61518-8 (cit. on p. 31).

Marcin Wylot, Manfred Hauswirth, Philippe Cudré-Mauroux, and Sherif Sakr (2018). RDF data storage and query processing schemes: a survey. In: *ACM Comput. Surv.* 51.4. ISSN: 0360-0300. DOI: 10.1145/3177850 (cit. on p. 3).

Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer (2016). Quality assessment for linked data: a survey. In: *Semantic Web* 7.1, pp. 63–93. DOI: 10.3233/SW-150175 (cit. on p. 3).

Wei Emma Zhang, Quan Z. Sheng, Kerry Taylor, Yongrui Qin, and Lina Yao (2016). Learning-based SPARQL query performance prediction. In: *Web Information Systems Engineering – WISE 2016*. Vol. 10041. Lecture Notes in Computer Science. Cham: Springer, pp. 313–327. ISBN: 978-3-319-48740-3. DOI: 10.1007/978-3-319-48740-3_23 (cit. on p. 52).

Matthäus Zloch (2016). Methods for automatic selection of database systems for optimized query performance. In: *46. Jahrestagung der Gesellschaft für Informatik (INFORMATIK 2016)*. Vol. P-259. Lecture Notes in Informatics (LNI) – Proceedings. Bonn: Gesellschaft für Informatik e.V., pp. 2019–2024. ISBN: 978-

3-88579-653-4. URL: https://dl.gi.de/20.500.12116/1097 (cit. on pp. 3, 11, 52).

Matthäus Zloch (2018). Browsable version of results from a large-scale graph-based analysis on 280 knowledge graphs. https://data.gesis.org/lodcc/2017-08/. Last accessed on November 7, 2020 (cit. on pp. 7, 17).

Matthäus Zloch (2020). Lodcc: a software framework for the graph-based analysis on RDF graphs. In: DOI: 10.5281/zenodo.2109469. URL: https://github.com/mazlo/lodcc (cit. on pp. 6, 16, 17, 43).

Matthäus Zloch and Maribel Acosta (2018). Lod-graph-analysis: results on the graph-based analysis of 280 real-world knowledge graphs, including plots, scripts, and dataset topology profiles. In: DOI: 10.5281/zenodo.2203826 (cit. on pp. 7, 16, 48).

Matthäus Zloch, Maribel Acosta, Daniel Hienert, Stefan Conrad, and Stefan Dietze (2020). Characterizing RDF graphs through graph measures – framework and assessment. In: *Semantic Web* – Pre-press. DOI: 10.3233/SW-200409 (cit. on pp. 3, 9, 22, 29–32, 34–38, 45, 47, 49).

Matthäus Zloch, Maribel Acosta, Daniel Hienert, Stefan Dietze, and Stefan Conrad (2019). A software framework and datasets for the analysis of graph measures on RDF graphs. In: *The Semantic Web – 16th Extended Semantic Web Conference (ESWC 2019)*. Vol. 11503. Lecture Notes in Computer Science. Springer, pp. 523–539. ISBN: 978-3-030-21348-0. DOI: 10.1007/978-3-030-21348-0 (cit. on pp. 3, 10, 15, 16, 18, 21, 39, 41–43, 45, 47, 49, 51).

Matthäus Zloch, Daniel Hienert, and Stefan Conrad (2017). Towards a use case driven evaluation of database systems for RDF data storage – a case study for statistical data. In: *Joint Proceedings of BLINK 2017: Benchmarking Linked Data and NLIWoD3: Natural Language Interfaces for the Web of Data (BLINK 2017-NLIWoD3)*. CEUR Workshop Proceedings 1932. Aachen: CEUR-WS.org. URL: http://ceur-ws.org/Vol-1932/#paper-08 (cit. on pp. 10, 37, 39–41, 47, 50).

Mussaba Zneika, Dana Vodislav, and Dimitris Kotzinos (2019). Quality metrics for RDF graph summarization. In: *Semantic Web* 10.3 (Special Issue on Intelligent Exploration of Semantic Data), pp. 555–584. DOI: 10.3233/SW-190346 (cit. on pp. 3, 7).

List of Figures

List of Tables