TU WIEN Informatics

Influence of Knowledge Graph Characteristics on Embedding-based Recommender Systems

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Data Science

eingereicht von

Elisabeth Harlander, BSc
Matrikelnummer 01106899

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Reka Marta Sabou, Ph.D.
Mitwirkung: Dr.techn. Fajar Juang Ekaputra

Wien, 28. August 2024

[redacted] [redacted]

Elisabeth Harlander Reka Marta Sabou

Informatics

Influence of Knowledge Graph Characteristics on Embedding-based Recommender Systems

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Data Science

by

Elisabeth Harlander, BSc

Registration Number 01106899

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Reka Marta Sabou, Ph.D.

Assistance: Dr.techn. Fajar Juang Ekaputra

Vienna, August 28, 2024

[redacted]
Elisabeth Harlander

[redacted]
Reka Marta Sabou

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Elisabeth Harlander, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 28. August 2024

[redacted]

Elisabeth Harlander

Danksagung

Zuallererst möchte ich meiner Betreuerin Marta Sabou für ihre fortwährende Unterstützung meinen tiefsten Dank aussprechen. Ihre Geduld und Ermutigungen während des gesamten Prozesses waren über die Maßen hilfreich, um die Herausforderungen der Arbeit zu meistern.
Ich möchte mich auch bei meinem Co-Betreuer Fajara Ekaputra bedanken, dessen Expertise und ansteckende Positivität ein Element der Freude in dieses Projekt einbrachte und dafür sorgte, dass der Prozess nicht nur produktiv war, sondern auch Spaß machte.
Besonderer Dank gebührt Kevin Haller, dessen bemerkenswertes Know-How mich immer wieder verblüfft hat. Ich schätze seine Bereitschaft, dieses Wissen mit mir zu teilen und meine Arbeit damit zu bereichern.
Meine Betreuerin und Betreuer standen mir mit ihrer Leidenschaft und ihrem Einfühlungsvermögen immer zur Seite, wenn ich nicht weiterkam oder in einer anderen Form Hilfe brauchte und bildeten so das beste Betreuerteam, das man sich wünschen kann.

Darüber hinaus möchte ich mich bei Karl-Heinz Hudej und meinen Kollegen Patrick Brunmair und Lukas Baumgartner für ihr Verständnis und die Unterstützung bedanken, die es mir ermöglichten, das Schreiben der Diplomarbeit und eine Vollzeit-Arbeitsstelle unter einen Hut zu bringen.

Meiner Familie bin ich zutiefst dankbar, ohne sie wäre diese Arbeit nicht möglich gewesen. Meinen Eltern Elfi und Christian, meinem Bruder Hannes, meinen Großeltern Johann, Anna, Hubert und Johanna, meinem Onkel Robert, meiner Tante Lydia und meiner Cousine Tini – danke euch allen für eure unendliche Unterstützung.

Ein besonderes Dankeschön geht an Clemens, der hier zwar nicht erwähnt werden möchte, aber ich mache es trotzdem. Ohne ihn hätte ich das nicht geschafft.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor Marta Sabou for her guidance and everlasting support. Her patience and encouragement were helpful beyond measure throughout this process, helping me to navigate the challenges of this research.
I would also like to thank my co-supervisor Fajar Ekaputra, whose expertise and contagious positivity brought an element of joy to this project and made this process not only productive but also fun.
Special thanks are due to Kevin Haller, whose remarkable know-how never ceased to amaze me. I highly appreciate his willingness to share his insights, which greatly enriched my work.
With their passion and empathy, my advisors were always there for me when I was stuck or needed assistance in any way, forming the best team of supervisors one could wish for.

Additionally, I would like to express gratitude to Karl-Heinz Hudej and my colleagues Patrick Brunmair and Lukas Baumgartner for their understanding and support, ensuring that I could manage writing a thesis while working full-time.

I am profoundly thankful to my family. Without them, this would not have been possible. To my parents, Elfi and Christian, my brother Hannes, my grandparents Johann, Anna, Hubert and Johanna, my uncle Robert, my aunt Lydia and my cousin Tini – thank you all so very much for your never-ending support.

A special thank you goes to Clemens, who does not want to be mentioned here, but I am doing it nonetheless. I could not have done this without him.

Kurzfassung

Recommender Systeme spielen eine entscheidende Rolle bei der Verbesserung der Benutzererfahrung in Webanwendungen, indem sie personalisierte Vorschläge aus einer wachsenden Anzahl von Optionen liefern. Die Verwendung des reichhaltigen Kontexts von Knowledge Graphen hat im wissenschaftlichen Bereich als Möglichkeit zur Steigerung der Genauigkeit und Erklärbarkeit dieser Empfehlungen an Aufmerksamkeit gewonnen. Es gibt jedoch immer noch einen Mangel an Verständnis darüber, wie distinktive Charakteristiken von Knowledge Graphen die Performance von Recommender Systemen beeinflussen können.

In dieser Arbeit wird der Einfluss von Knowledge Graphen Charakteristiken auf die Performanz von Recommender Systemen anhand folgender Forschungsfragen untersucht:

RQ1 *Was sind die definierenden Charakteristiken eines Knowledge Graphen, die ihn von einem anderen Knowledge Graphen unterscheiden? Wie können diese Charakteristiken quantifiziert werden?*

RQ2 *Welche Knowledge Graphen werden in Knowledge Graph-basierten Recommender Systemen häufig verwendet? Was sind deren Charakteristiken?*

RQ3 *Wie verhält sie die Performanz von Recommender System-Algorithmen bei Knowledge Graphen mit distinktiven Charakteristiken? Welchen Einfluss haben bestimmte Charakteristiken auf die Performanz einer Familie von Recommender-Algorithmen?*

Für die ersten beiden Forschungsfragen wurden semi-systematische Literaturreviews durchgeführt und die dritte Forschungsfrage wurde empirisch untersucht. Konkret bestand die Untersuchung darin, die Performanz unter der Verwendung von drei unterschiedlichen Embedding-Algorithmen (RDF2Vec, TransE und DistMult), distanzbasierten Methoden (LDSD, Resim) und einem auf Informationsgehalt basierenden Ansatz (PICSS) für mehrere Knowledge Graphen mit distinktiven Charakteristiken zu messen.

Aus der theoretischen Analyse für RQ1 konnten wir vier Hauptkategorien von Charakteristiken identifizieren: Qualitätscharakteristiken, statistische Charakteristiken, statistische Verteilungen und Charakteristiken abgeleitet aus der Graphentheorie. Die zweite Review ergab DBpedia, Wikidata und YAGO als favorisierte Knowledge Graphen, die öffentlich zugänglich sind und verwaltet werden (RQ2).

Die Ergebnisse des kontrollierten Experiments, das zur Beantwortung von RQ3 durchgeführt und durch statistische Tests validiert wurde, hebt jene Charakteristiken hervor, welche die Performanz von Recommender Systemen signifikant beeinflussen. Insbesondere die Erhöhung der Verlinkungseigenschaft durch `owl:sameAs` Prädikate und das Hinzufügen von inversen Relationen wirkten sich am stärksten auf die Performanz aus. Bei den Embedding-basierten Algorithmen beeinflussten die inversen Relationen nur die Performanz bei RDF2Vec, während TransE und DistMult von dieser Änderung nicht signifikant betroffen waren. Die Ergänzung von Blank Nodes in den Graphen zeigte bei TransE und auch bei den kleineren Graphen bei RDF2Vec statistische Signifikanz, wohingegen DistMult nur von der Änderung der Verlinkung betroffen war. Die Performanzen bei den distanzbasierten Methoden änderten sich nur bei der Verlinkung und den inversen Relationen signifikant. Die auf dem Informationsgehalt basierende Methode blieb über alle experimentellen Durchführungen hinweg unbeeinflusst.

Diese Forschung trägt zum Verständnis von Knowledge Graphen und ihrer Verwendung in Recommender Systemen bei und bietet eine Grundlage für zukünftige Diskussionen und Erweiterungen dieser Systeme.

Abstract

Recommender Systems (RS) play a crucial role in enhancing user experience in web application by providing personalized suggestions from a growing number of options. Using the rich context provided by a Knowledge Graph (KG) has gained attention in the scientific field as a way to improve the accuracy and explainability of these recommendations. However, there is still a lack of understanding on how distinctive characteristics of KGs can impact the performance of RS.

This thesis investigates the influence of KG characteristics on the performance of RS with the following research questions:

RQ1 *What are the defining characteristics of a KG that distinguishes it from another KG? How can these characteristics be quantified?*

RQ2 *What KGs are commonly used to evaluate KG-based recommender algorithms? What are their characteristics?*

RQ3 *How do the RS algorithms perform on KGs with distinctive characteristics? What influence do certain KG characteristics have on the performance of a family of recommendation algorithms?*

For the first two research questions semi-systematic reviews were conducted and the third research question was empirically investigated. Concretely, this investigation consisted of evaluating the performances using three different embedding algorithms (RDF2Vec, TransE and DistMult), distance-based methods (LDSD and Resim) and one information content-based approach (PICSS) across multiple KGs with distinctive characteristics.

From the theoretical analysis for RQ1 we were able to identify four key categories of KG characteristics: quality characteristics, statistical characteristics, statistical distributions and characteristics obtained from graph theory. The second review revealed DBpedia, Wikidata and YAGO as favored KGs, which are openly accessible and maintained (RQ2).

The results from the controlled experiment conducted to answer RQ3, validated through statistical testing, highlight the characteristics that significantly affect the performance of RS. Specifically the increase of the linkage with `owl:sameAs` predicates and the addition of inverse relations were mostly affecting the performances. Among the embedding-based algorithms, the inverse relations only influenced the performance with RDF2Vec, while TransE and DistMult were not significantly affected by this change. The inclusion of blank nodes in the graphs showed a statistical significance with TransE and also on the smaller graphs with RDF2Vec, while DistMult was only affected by the change of the linkage. The performances on the distance-based methods only changed significantly with the linkage and the inverse relations. The information content-based method remained unaffected over all experimental setups.

This research contributes to the understanding of KG and their utilization in RS, offering a foundation for future discussion and enhancements of these systems.

Contents

CHAPTER 1

Introduction

Recommender Systems (RS) play a vital role in navigating the fast-paced and information-packed digital landscape, tailoring contents to users' unique preferences across streaming platforms, news websites or e-commerce sites. These systems aim to decrease the information overload by highlighting content likely to resonate with individual users.

Although methods such as collaborative or content-based filtering are commonly used approaches, the utilisation of a Knowledge Graph (KG) has evolved as a promising direction in the research of RS. As Chicaiza et al., [CVD21] highlight, there has been a rapid increase in scientific papers regarding KG-enabled RS since 2018. This trend is validated by Figure 1.1, showing a selection of publications we explored for one of our research questions (see Chapter 5).

KGs provide a way to represent complex data in the form of interconnected entities (such as people or things) and relationships between them. By incorporating a KG, the RS model can leverage the additional knowledge encoded in the graph to enhance the prediction accuracy (Zhang et al., [ZWL20]), increase the explainability of the results (Suzuki et al. ,[SOK19]; Suzuki et al., [CMEC17]) and overcome the cold-start problem (Tomeo et al., [TFTCDN17]; Meymandpour et al., [MD15]).

Recommendation models based on a KG can roughly be classified into two groups: traditional algorithms such as path-based models and Knowledge Graph Embedding (KGE) models. The latter translates KGs into a low-dimensional numerical representation of entities and relationships while (partially) preserving the semantic information encoded. This results in semantically similar entities being in close proximity to each other (Ristoski et al., [Ris19]).

Figure 1.2 shows a timeline of some KGE models. The algorithm RESCAL was the first modern KGE algorithm, which was proposed in 2011 by Nickel et al. [NTK$^+$11]. After that, more researchers introduced other versions of KGE models, indicating a rising interest in this topic.

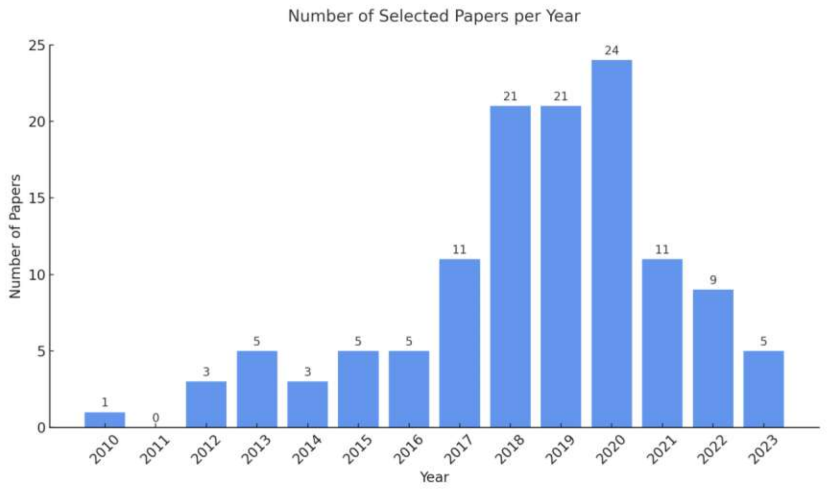


Figure 1.1: Publications per year of the paper which were regarded for the second research question.

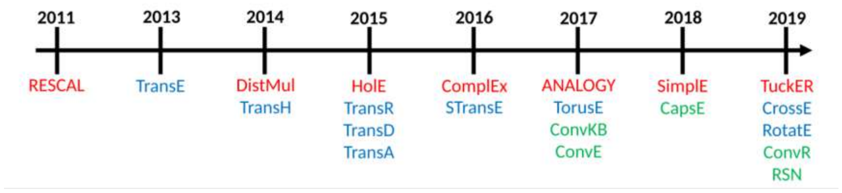


Figure 1.2: Timeline of the publications introduction KGE models (image by Slloris - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=112599192, accessed 05.08.2024). The different colors indicate different families of KGE models (blue: geometric models, red: tensor decomposition models, green: deep learning models). Not depicted in the timeline is RDF2Vec by Ristoski and Paulheim [RP16a], which was first introduced in 2016

Certain characteristics of a KG, which can be attributes and properties that describe its structure or content, can influence the usability and effectiveness of the KG for various applications, including RS (Heinrich et al., [HHL+21]). Understanding these attributes is crucial for grasping their influence on the performance and suitability of RS algorithms.

However, as Formica and Taglino already argued in [FT23], a lot of the research comparability is hampered by variations in datasets, KG subsets and the constant development of some KGs themselves. This underscores the necessity of employing consistent KGs and datasets across experiments.

Despite the observed interest in this research field (see Chapter 3), there is still a gap in understanding how the characteristics of a KG influence the performance of RS. Studies on the characteristics rarely focus on RS, and as a result, they do not evaluate how changes in these characteristics can influence the performance. Although some literature on KG-based RS acknowledge the characteristics of KGs, they do not propose any further research on their impact.

This lack of empirical evidence, combined with the challenges in comparing existing research, hinders the ability of making informed decisions to select the most suitable KG for an optimal use in RS.

This study aims to explore this gap by analyzing the interplay between KG characteristics and recommendation quality. By conducting controlled experiments, the goal of this thesis is to guide future researchers towards more informed Knowledge Graph-based Recommender Systems design and implementation strategies to enhance prediction accuracy and to lead to a greater user satisfaction.

1.1 Aim of the Work

The aim of this thesis is to explore Knowledge Graph (KG) characteristics and their influence on the performance on different algorithms for Recommender Systems (RS).

In order to do this, we first aim to build a foundation on KG characteristics. To gain an understanding about this topic, we want to answer the following research question:

RQ1 *What are the defining characteristics of a KG that distinguishes it from another KG? How can these characteristics be quantified?*

Another subject we want to investigate is the current research on KG-based RS. We want to explore, what KGs researchers use for their RS and if they also focus on their characteristics and analyze them, if they are reported. This goal is summed up in the following research question:

RQ2 *What KGs are commonly used to evaluate KG-based recommender algorithms? What are their characteristics?*

The findings of the previous questions leads to a practical part, where we want to investigate, if we can observe any influence of the KG characteristics in the performance of RS. This experiment is summarized in the following research question:

RQ3 *How do the RS algorithms perform on KGs with distinctive characteristics? What influence do certain KG characteristics have on the performance of a family of recommendation algorithms?*

1.2 Methodology

The methodological approach is explained in detail in Chapter 4.

To outline the process, before investigating the performance of RS algorithms on KGs it is necessary to determine how KGs can be characterized. By quantifying these characteristics, a distinction between two KGs can be drawn. Therefore, RQ1 will be the first research question that is investigated. Chapter 5 addresses this question by exploring methods to characterize and measure distinctive features of KGs in the form of a semi-systematic review as outlined by Snyder [Sny19].

RQ2 aims to identify commonly used KGs for RS in research. Additionally, the characteristics of these KGs are need to be reported and summarized. The execution of this semi-systematic review according to Snyder [Sny19] is laid out in Chapter 6.

After the defining characteristics have been explored and the most commonly used KGs have been identified, the performance of various RS algorithms to recommend N items can be compared using the Precision@N, Recall@N and the F-measure ([GS15]). This will be explored using an experiment following the guidelines by Wohlin et al. [WRH+12]. The algorithms used in this experiment are explained in Chapter 2.7 and the practical part of this thesis is summarized in Chapter 8.

Figure 1.3 visualizes the methodology for this thesis and the relationship between the research questions.

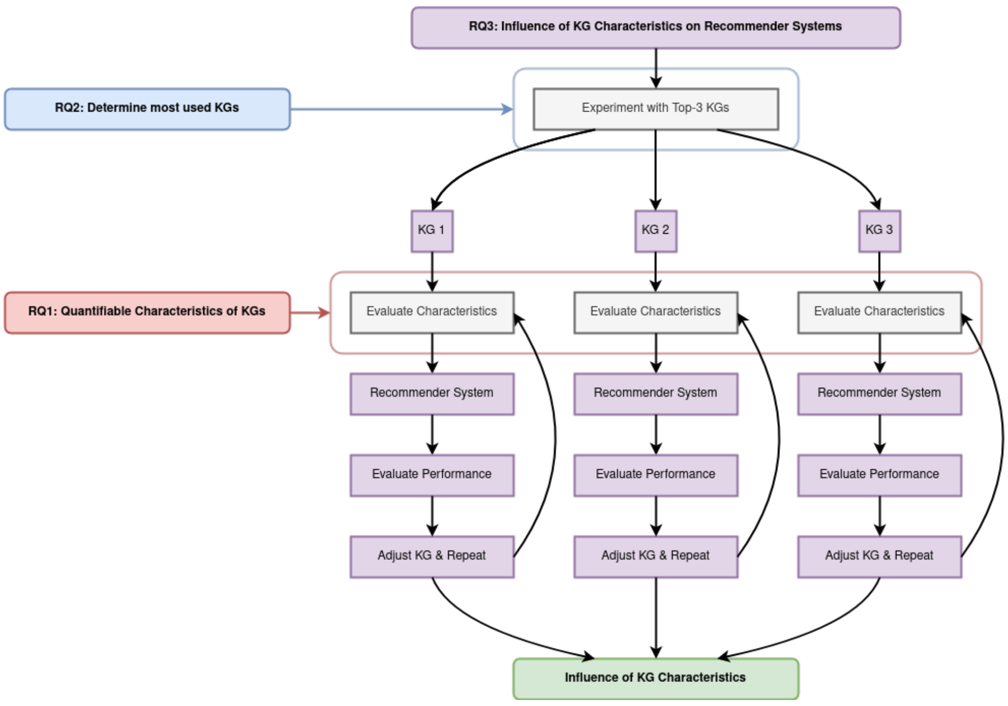


Figure 1.3: Methodological approach for the exploration of the research questions. The results from the first two research questions (blue and red boxes) are needed to conduct the experiment for the third research question (violet boxes).

1.3 Contributions

This thesis aims to contribute in the following way to the field of KG-based RS:

- **Characterization of KGs:** By performing a semi-systematic review of the metric to characterize KGs, we aim to achieve a deeper understanding on how to quantify and distinguish multiple KGs.
- **Review of used KGs:** With the second semi-systematic review of the KGs used by researchers for the task of RS, we seek to identify popular KGs and summarize the characteristics which are already used.
- **Empirical evaluation:** Through experimentation, this thesis intent is to evaluate the impact of different characteristics on the performance of KG-based RS, throwing light on the interplay of the characteristics and algorithm performance.
- **Guidance of best practices:** The findings of the thesis aim to contribute to best practices for selecting or preprocessing a KG for RS.
- **Discussion for future research:** By identifying key factors which influence the performance of RS with KGs, we aspire to create a foundation for discussion and future research.

Overall, by studying the influence of certain characteristics on the performance of KG-based recommender systems, we aim to gain new insights on the strengths and limitations of the algorithms as well as the relationship between their performance and the KG characteristics and want to build a foundation for discussion.

1.4 Structure of the Work

This diploma thesis is structured as follows:

Chapter 2: Background provides an overview to the relevant theory and key concepts of this thesis. This chapter also contains the theoretical background of the algorithms which were used in the experiment.

Chapter 3: State of the Art discusses the existing literature and research regarding KG-based RS, Knowledge Graph Embedding and traditional algorithms for KG-based RS. This chapter also summarizes challenges and shortcomings identified by the researchers.

Chapter 4: Methodology describes the methods used to gain insights for the theoretical research questions. It also contains the scientific background used for the implementation and evaluation of the metrics used to perform the experiment.

Chapter 5: Key Characteristics of Knowledge Graphs explores the first research question by identifies the defining characteristics of KGs.

Chapter 6: Commonly used Knowledge Graphs in Recommender Systems reviews existing work on KG-based RS with a focus on the KGs that were used by the researches.

Chapter 7: Experimental Evaluation outlines the experimental setup and illustrates the practical implementation of the proposed approach which is influenced by the insights gained from the chapters 5 and 6. The results obtained from the implementation are presented and analysed and their implications are discussed.

Chapter 8: Conclusion and Future Work revisits the research questions and summarizes the contributions and insights of this thesis. Potential future research directions are identified and open questions are summarized.

CHAPTER 2

Background

This chapter lays the foundation for understanding the key concepts referenced in the subsequent chapters such as Recommender Systems (RS), Knowledge Base (KB), Knowledge Graph (KG) and Knowledge Graph Embedding (KGE).

2.1 Recommender Systems

A RS serves as a tool or a technique designed to provide relevant and customized suggestions for users by analyzing historical data and user preferences. Various domains such as e-commerce, entertainment, social networks and content streaming rely on RS to overcome the problem of information overload by recommending users content based on their interests.

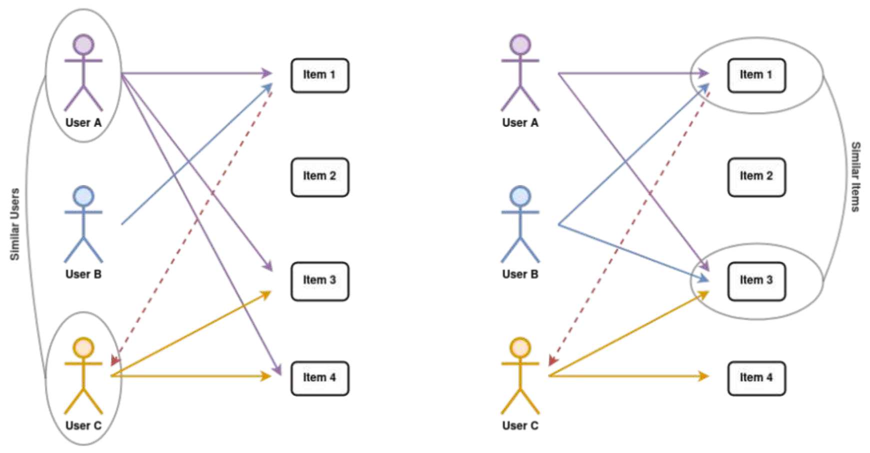
Chicaiza et al. ([CVD21]) categorize RS in three groups: the first generation consists of the methods Collaborative Filtering (CF), Content-based Filtering (CBF) and hybrid methods. RS of the second generation are context-based methods and the third generation is dedicated to semantic models. The first and last generation are explained in more detail in the following sections.

2.1.1 Collaborative Filtering

Collaborative Filtering (CF) leverages the collective user behavior of the interactions with items under the assumption that a users' preference remains stable and consistent over time. Furthermore, the methods expects that similar users are likely to prefer similar items.

User-based CF assesses the collective behaviour of users, while item-based CF focuses on the similarity of items. Two items are considered similar if they receive a similar engagement (e.g. a rating as explicit feedback or a click as implicit feedback) by users.

While CF can increase the serendipity of recommendations, the method also struggles with the cold-start problem due to the lack of data for new users or items.



(a) User-based CF (b) Item-based CF

Figure 2.1: Comparison of user-based (left) and item-based (right) CF. The solid lines depict an interaction of a user with an item and the dashed line show a recommendation. In the left diagram (a), Item 1 gets recommended to User C based on the similarity of this user to User A because they both interacted with Item 3 and Item 4. In the right diagram (b), Item 1 and Item 3 are considered similar because User A and User B interacted with both of them and therefore Item 1 might be a good fit for User C, who also interacted with Item 3.

2.1.2 Content-based Filtering

In Content-based Filtering (CBF) methods the system recommends items similar to those the user has liked in the past, based on the content of items. The user preferences are described in the user profile which are compared to the attributes of an item content such as tags, textual information or attributes. A major benefit of this method is that there is no cold-start problem for items, since there is no user feedback required. The drawback is that the users get only a limited diversity of recommendations and this method relies on the richness of item descriptions.

2.1.3 Knowledge-based Filtering

Knowledge-based filtering aims align recommendations with user requirements by using explicit knowledge about both the users and items. This method allows for a higher explainability of the resulting recommendations and sidesteps the cold-start problem but demands more complex development efforts.

2.1.4 Hybrid Methods

Hybrid Methods are a combination of two or more filtering methods which can result in more accurate recommendations by compensating the limitations of a single method. Due to the higher complexity this method requires more computational power than a single recommender system.

2.2 Resource Description Framework

The Resource Description Framework (RDF) was developed in the late 1990s as a framework the representation of semantic information in the Web. The RDF 1.1 specification follows recommendations by World Wide Web Consortium (W3C)[1], where statements are in the form of RDF triples. Each triple consists of a *subject*, a *predicate* and an *object* (see Figure 2.2).

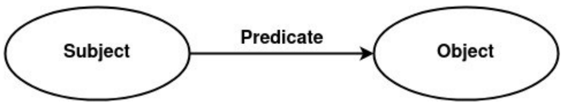


Figure 2.2: RDF triple visualized as a small graph (from https://www.w3.org/TR/rdf11-concepts/, accessed 05.08.2024).

A subject denotes a resource which can be in the form of an Internationalized Resource Identifier (IRI), an Uniform Resource Identifier (URI) or represented as a blank node. Objects are similar to subjects but can also be in the form of a literal, encompassing strings, numbers or dates. The predicate denotes a relationship between the subject and the object. As an example, the phrase "Alice is a friend of Bob" can be translated as the RDF triple "Alice" (subject), "friendOf" (predicate), "Bob" (object).

RDF data can be serialized into various formats, including RDF/XML, N3/Turtle, N-Triples and JSON-LD. For a detailed definition of the RDF 1.1 framework see the W3C recommendations from 2014[2].

2.3 Linked Open Data

Linked Data (LD) refers to a set of best practices for publishing and interlinking structured data on the web based on the RDF standards. The main prospect is to facilitate the connectivity of data across different domains in a format that is accessible and readable by machines. If the data is released under an open licence, it is called Linked Open Data (LOD). Figure 2.4 depicts the LOD-cloud of linked data sets from the year 2023.

Tim Berners-Lee has proposed a five-star model as a rating system to guide the publication of "good" LOD[3]:

[1]https://www.w3.org/, accessed 05.08.2024
[2]https://www.w3.org/TR/rdf11-concepts/, accessed 05.08.2024
[3]https://www.w3.org/DesignIssues/LinkedData, accessed 05.08.2024

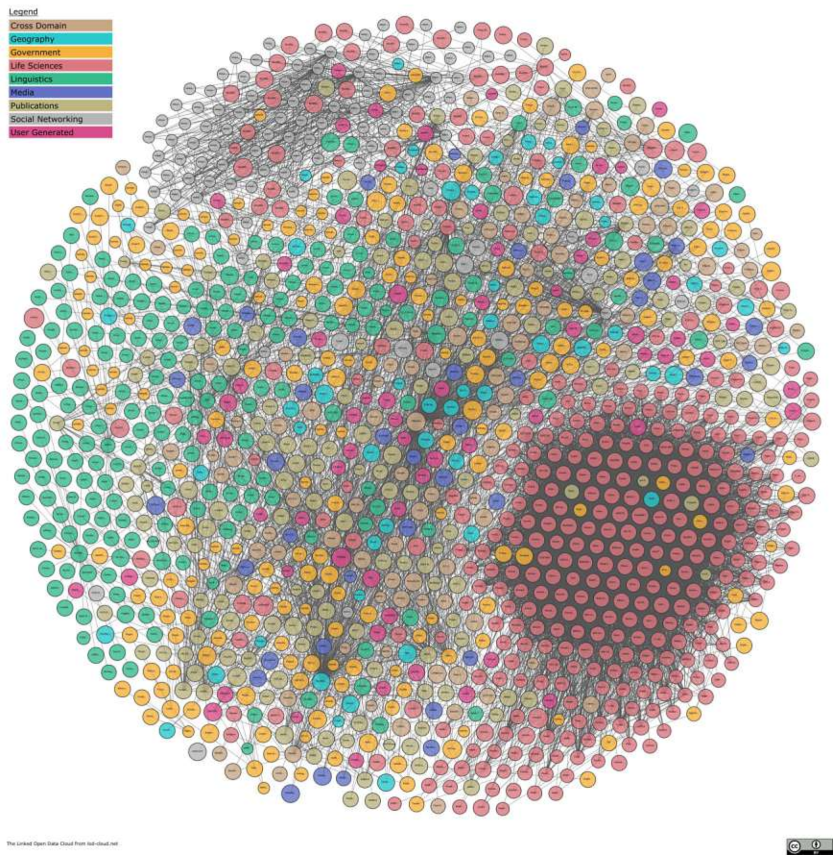


Figure 2.3: LOD cloud diagram of 2023 visualizing the available linked data sets (from https://lod-cloud.net, accessed 05.08.2024).

1. Data should be available on the web.
2. Data should be machine-readable and structured.
3. Data should be in a non-proprietary format (e.g., CSV).
4. Data should be identifiable using open standards from W3C.
5. Data should be linked.

2.4 Knowledge Base

A Knowledge Base (KB) contains various types of knowledge such as rules, facts, axioms, definitions, statements, and primitives in a dataset with formal semantics ([EW16]). Kejriwal et al. [KKS21] argue that a KB is similar to a KG, but without structure. As part of a knowledge-based system, a KB is essentially a storage system for knowledge which does not necessarily possess the capabilities to integrate new information sources or derive knowledge through reasoning ([EW16]).

2.5 Knowledge Graph

A Knowledge Graph (KG) is a powerful way to store and represent knowledge according to RDF standards, where entities (nodes) are interconnected through relations (edges). As opposed to other data storage such as Relational Database Management System (RDBMS), KGs serve as a structured form of knowledge representation, offering insights through interrelations among entities.

There have been mentions of KGs in the literature since 1972, but after the announcement in 2012 of the Google Knowledge Graph[4], this concept gained more interest in the scientific field ([HBC+21]).

Several definitions of a KG can be found, describing it as a machine-readable way of representing information or a set of triples ([KKS21]), as networks of entities, their semantic types, properties and the relationships between them ([KW16]) or as a RDF graph ([FBMR18], see also Chapter 2.7).

Hogan et al. define a KG as *"a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent potentially different relations between these entities"* (from [HBC+21]).

Ji et al. [JPC+21] see the differentiating factor of a KB and KG only in the graph structure of a KG and use both terms interchangeably as synonyms.

Ehrlinger and Wöß [EW16] were aiming to find a definition of a KG that is free from ambiguity and can serve as a common basis for discussion. According to these authors, the terms KB and KG are often used interchangeably. They define a KG as follows: *"A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge"* (from [EW16]), which also distinguishes a KG from a KB.

KGs have several key advantages compared to other data representation models, including:

- KGs capture semantic relationships between entities, resulting in a higher understandability and explainability of the data.
- Through reasoning capabilities and the use of ontologies a KG can infer new knowledge from existing data.
- A KG can combine data from diverse sources.

KGs are used in many applications across different research domains, the literature review from Ji et al. [JPC+21] resulted in the following categorization (see Figure 2.6):

- Knowledge representation learning

[4]https://blog.google/products/search/introducing-knowledge-graph-things-not/, accessed 18.02.20224

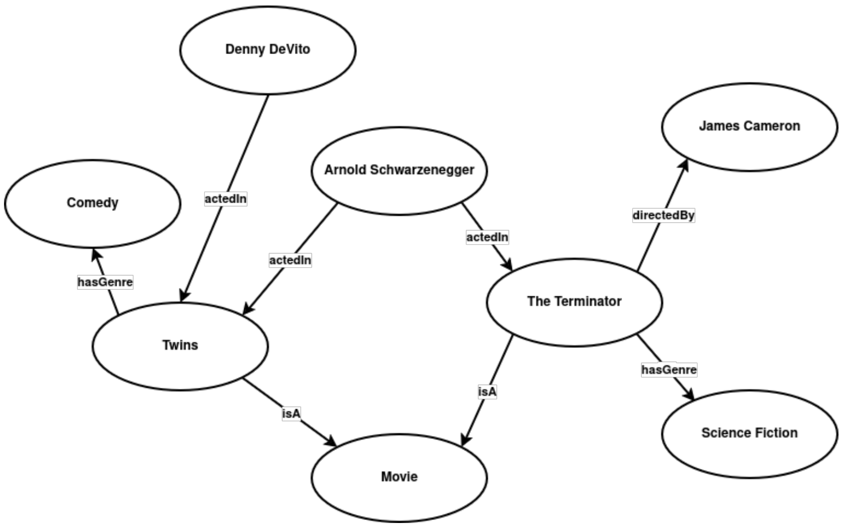


Figure 2.4: Example of a small KG.

- Knowledge acquisition
- Temporal knowledge graphs
- Knowledge-aware applications

As mentioned in above, the Google Knowledge Graph increased the attention towards KGs. For the web search on Google[5], the data is enhanced with structured, linked data to not only improve the search results, but also provide so-called "knowledge panels"[6] (see Figure 6.2) of famous person or geographical entities [KKS21].

These panels contain information derived from sources such as Wikipedia and also include the current context of the search (e.g. search history or the location where the search has been executed)[KKS21]. The blog post introducing the Google Knowledge Graph[7] used the search value "Taj Mahal" as an example to outline the versatility of viewing search entries as things (entities) and not as strings: typing in these words while being situated in Vienna, the knowledge panel not only gives information about the Indian monument, but also about a restaurant in the 7th district.

According to the original blog post from 2012[8], the KG contains more than 500 million objects and more than 3.5 billion facts and relationships between them. As of the latest blog post from 2020[9], it has grown to a size of more than 5 billion entities and 500 billion facts.

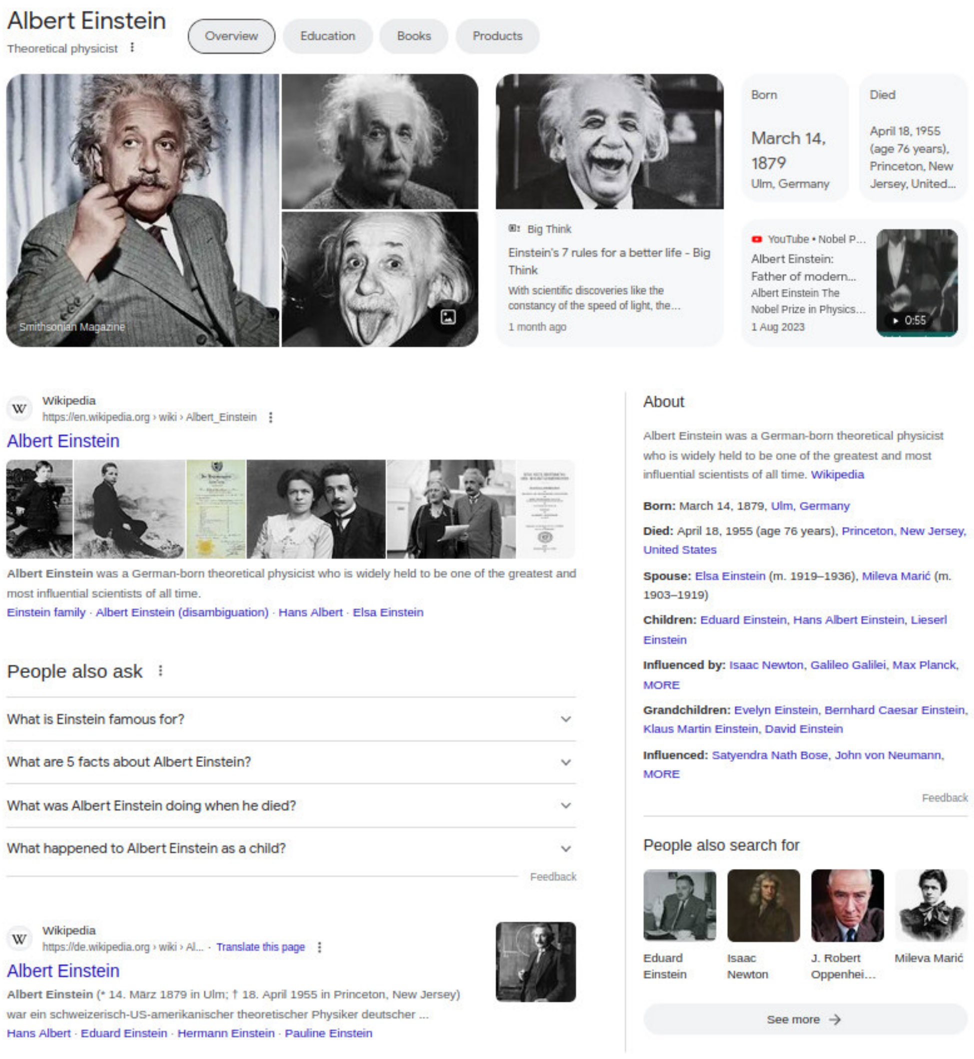
[5]https://www.google.com/, accessed 05.08.2024

[6]https://blog.google/products/search/about-knowledge-graph-and-knowledge-panels/, accessed 11.06.2024

[7]https://blog.google/products/search/introducing-knowledge-graph-things-not/, accessed 18.02.2024

[8]see footnote 7

[9]see footnote 6



Albert Einstein
Theoretical physicist

Born
March 14, 1879
Ulm, Germany

Died
April 18, 1955 (age 76 years), Princeton, New Jersey, United...

Big Think
Einstein's 7 rules for a better life - Big Think
With scientific discoveries like the constancy of the speed of light, the...
1 month ago

YouTube • Nobel P...
Albert Einstein: Father of modern...
Albert Einstein The Nobel Prize in Physics...
1 Aug 2023
0:55

Wikipedia
https://en.wikipedia.org › wiki › Albert_Einstein
Albert Einstein

Albert Einstein was a German-born theoretical physicist who is widely held to be one of the greatest and most influential scientists of all time.
Einstein family · Albert Einstein (disambiguation) · Hans Albert · Elsa Einstein

People also ask

What is Einstein famous for?

What are 5 facts about Albert Einstein?

What was Albert Einstein doing when he died?

What happened to Albert Einstein as a child?

Feedback

Wikipedia
https://de.wikipedia.org › wiki › Al... · Translate this page
Albert Einstein
Albert Einstein (* 14. März 1879 in Ulm; † 18. April 1955 in Princeton, New Jersey) war ein schweizerisch-US-amerikanischer theoretischer Physiker deutscher ...
Hans Albert · Eduard Einstein · Hermann Einstein · Pauline Einstein

About

Albert Einstein was a German-born theoretical physicist who is widely held to be one of the greatest and most influential scientists of all time. Wikipedia

Born: March 14, 1879, Ulm, Germany

Died: April 18, 1955 (age 76 years), Princeton, New Jersey, United States

Spouse: Elsa Einstein (m. 1919–1936), Mileva Marić (m. 1903–1919)

Children: Eduard Einstein, Hans Albert Einstein, Lieserl Einstein

Influenced by: Isaac Newton, Galileo Galilei, Max Planck, MORE

Grandchildren: Evelyn Einstein, Bernhard Caesar Einstein, Klaus Martin Einstein, David Einstein

Influenced: Satyendra Nath Bose, John von Neumann, MORE

Feedback

People also search for

Eduard Einstein
Isaac Newton
J. Robert Oppenhei...
Mileva Marić

See more →

Figure 2.5: A knowledge panel displayed on Google describing Albert Einstein. Besides pictures and key facts about his life (born, died, etc.), it also has shortcuts to his Education, books and products related to Albert Einstein.

2.5.1 Ontology

An ontology is a formally represented conceptualization of a domain that is characterized by a semantic expressiveness. The key components of an ontology are:

- **Concepts:** these are the main concepts of the domain (e.g. Person)
- **Concept hierarchies:** hierarchical structures (e.g. Man is a Person)
- **Relationships:** associations (e.g. Man has Child)

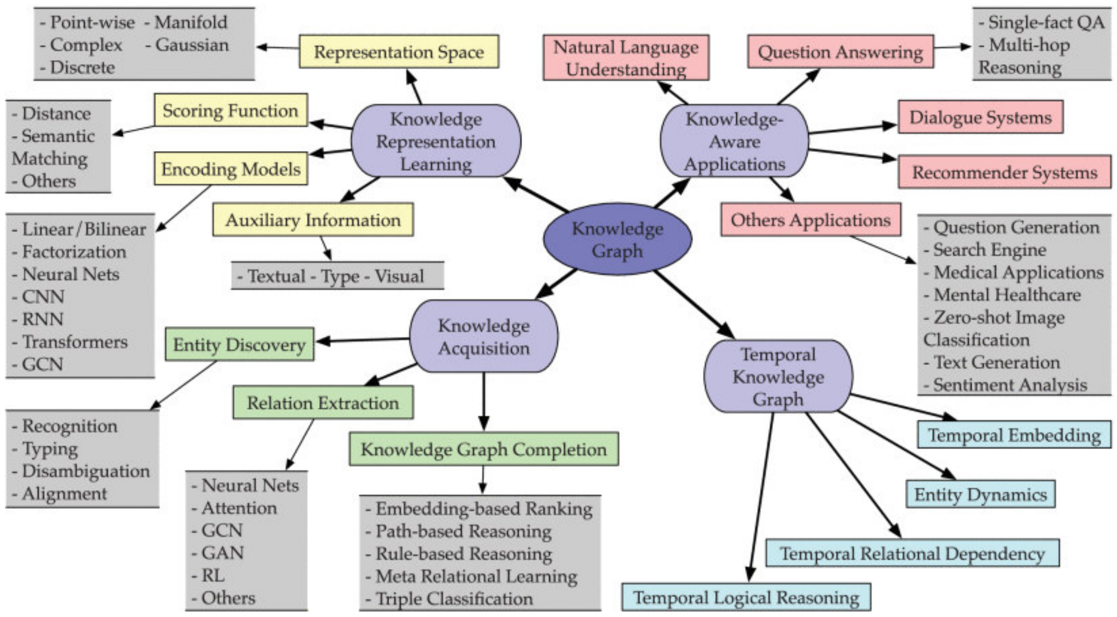


Figure 2.6: Research applications of KGs (from [JPC+21] as categorized by the authors).

- **Restrictions on relations:** those can be type or cardinal restrictions (e.g. Person has exactly one birthday)
- **Instances:** these are concrete entities (e.g. Bob, Alice)

The semantic web language Web Ontology Language (OWL) is designed to represent machine-readable knowledge about things, groups of things and the relationships between them.

Noy and McGuinnes [NM+01] designed guidelines based on their experience for developing ontologies.

2.6 SPARQL

SPARQL Protocol and RDF Query Language (SPARQL) is a language for querying and manipulate data which is stored in the RDF standard. The current standard is SPARQL 1.1 as defined by the W3C[10]. A query can take the form of a `SELECT`, `ASK`, `DEFINE` or `CONSTRUCT` and allows the definition of prefixes (see listing 2.1).

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name (COUNT(?friend) AS ?count)
WHERE {
```

[10]https://www.w3.org/TR/sparql11-overview/, accessed 18.02.2024

```
   ?person foaf:name ?name .
   ?person foaf:knows ?friend .
} GROUP BY ?person ?name
```

Listing 2.1: An example of a SPARQL query to return the names of persons and a count of their friends using the ontology foaf (friend of a friend) from https://www.w3.org/TR/sparql11-overview/

The ability of SPARQL to retrieve and manipulate complex data structures from KGs makes it a vital tool for interacting with the graphs, enabling efficient data retrieval and integration.

2.7 Knowledge Graph-based Recommender Systems

KG-based Recommender Systems (RS) exploit the structured representation of knowledge to gain additional knowledge about the users or the items for the discovering of new connections [CVD21]. With the additional use of KGs, the cold-start-problem (i.e. when there is no information about a new user) and data sparsity can be overcome to increase the accuracy and explainability of recommendations [CVD21].

To cover a variety of methods, three groups of algorithms were chosen: semantic distance measures (Section 2.7.1), information content-based metrics (Section 2.7.2) and Knowledge Graph Embedding (KGE) (Section 2.7.3). This selection was driven by their effectiveness in previous research (see Chapter 3 for detailed summaries) and the potential to provide meaningful contributions to the study of the impact of KG characteristics. They represent a diverse set of approaches for analyzing and leveraging KG-based RS, providing a robust foundation for the experimental part of this thesis.

To further explore the algorithms in this chapter we first need a formal definition of a Knowledge Graph (KG) as a Linked Open Data (LOD) dataset, analogous to [Pas10]:

Definition 2.7.1 (LOD dataset)**.** A LOD dataset is a graph G such as $G = (R, L, I)$, where $R = \{r_1, r_2, \ldots, r_n\}$ is a set of resources identified by their URI, $L = \{l_1, l_2, \ldots, l_n\}$ is a set of links identified by their URI and $I = \{i_1, i_2, \ldots, i_n\}$ is a set of instances of these links between resources, such as $i_i = \langle l_j, r_a, r_b \rangle$.

2.7.1 Distance-based Methods

Distance-based methods leverage the semantic distance of entities to obtain the relatedness between entities [PB16]. The core concept of these measures is, that two relations are more related if they are more connected in the graph ([ALG17]). This also leads to a higher explainability of the produced recommendations ([Pas10]), as the similarity of entities can be illustrated with the links between them.

For this thesis two distance measures were regarded in the experiment and discussed in the following sections: Linked Data Semantic Distance (LDSD) and Resource Similarity (Resim).

LDSD

LDSD was proposed by Passant [Pas10] and later expanded by Alfarhood et al. [ALG17] as Propagated Linked Data Semantic Distance (PLDSD). The LDSD algorithm leverages the direct and indirect links between two resources to calculate the distance between them.

Based on Alfarhood et al. [ALG17] we need the definitions of the direct and indirect links in a LOD dataset (see definition 2.7.1).

We define the *direct distance* C_d of two resources r_a and r_b with a link l_i between them as follows:

$$C_d(l_i, r_a, r_b) = \begin{cases} 1 & \text{if the link } \langle l_i, r_a, r_b \rangle \text{ exists,} \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

The *indirect incoming distance* C_{ii} between two resources r_a and r_b equals 1 if there is a resource r_c that is directly connected to both r_a and r_b via an outgoing node l_i:

$$C_{ii}(l_i, r_a, r_b) = \begin{cases} 1 & \{\exists r_c | \langle l_i, r_c, r_a \rangle \wedge \langle l_i, r_c, r_b \rangle\} \\ 0 & \text{otherwise} \end{cases} . \tag{2.2}$$

The *indirect outgoing distance* C_{io} between two resources r_a and r_b equals 1 if there is a resource r_c that is directly connected to both r_a and r_b via an incoming node l_i:

$$C_{io}(l_i, r_a, r_b) = \begin{cases} 1 & \{\exists r_c | \langle l_i, r_a, r_c \rangle \wedge \langle l_i, r_b, r_c \rangle\} \\ 0 & \text{otherwise} \end{cases} . \tag{2.3}$$

Figure 2.7 is an example of a small graph and the different kind of distances (direct and indirect) between two resources.

To lessen the impact on the most popular link, the distances between the resources are normalized by the logarithm of the sum of the links.

The *weighted direct distance* $DC(r_a, r_b)$ between two resources r_a and r_b is then given by:

$$DC(r_a, r_b) = \sum_i \frac{C_d(l_i, r_a, r_b)}{1 + \log(\sum_n C_d(l_i, r_a, r_n))} \tag{2.4}$$

The *weighted incoming indirect distance* $IC_i(r_a, r_b)$ between two resources r_a and r_b is

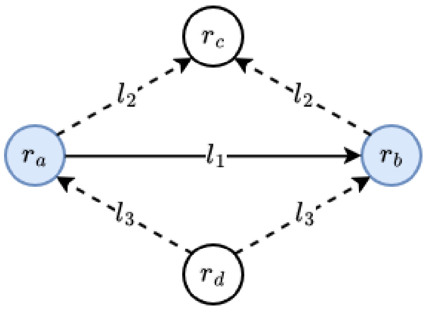


Figure 2.7: Graphical example of the distances between two resources r_a and r_b: the link l_1 denotes a direct link between the two resources, the dotted lines are the links l_2 and l_3 which also connect r_a and r_b indirectly via the nodes r_c and r_d respectively (adapted from [Pas10]).

defined as:

$$IC_i(r_a, r_b) = \sum_i \frac{C_{ii}(l_i, r_a, r_b)}{1 + \log(\sum_n C_{ii}(l_i, r_a, r_n))} \tag{2.5}$$

Similarly, we have the *weighted outgoing indirect distance* $IC_o(r_a, r_b)$ between two resources r_a and r_b:

$$IC_o(r_a, r_b) = \sum_i \frac{C_{io}(l_i, r_a, r_b)}{1 + \log(\sum_n C_{io}(l_i, r_a, r_n))} \tag{2.6}$$

If measures from formulas 2.4, 2.5 and 2.6 are combined, the direct and indirect relationships between two resources r_a and r_b in a graph are considered and this gives the definition for the *Linked Data Semantic Distance (LDSD)*:

$$LDSD(r_a, r_b) = \frac{1}{1 + DC(r_a, r_b) + DC(r_b, r_a) + IC_i(r_a, r_b) + IC_o(r_a, r_b)} \tag{2.7}$$

The distance measure takes values from 0 to 1, where a low value denotes a short distance and therefore higher relatedness[11].

The drawback of this distance measure is that two resources which are not directly or indirectly linked, are automatically considered to be unrelated (see Figure 2.8). This has been addressed by Alfarhood et al. [ALG17] using PLDSD and Piao et al. [PAB16] with the Resim measure (see also the next section).

[11]Although the LDSD does not satisfy the symmetry axiom of distance measures – i.e. $dist(a, b) = dist(b, a)$ for a distance measure $dist$ between two nodes a and b – the term "distance" is still used in the literature

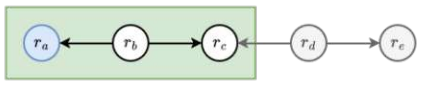


Figure 2.8: In this example, the reachable nodes for resource r_a in regards to LDSD are the nodes r_b and r_c. The resources r_4 and r_5 do not influence the distance measure, because they are considered unrelated to the algorithm (redrawn image from [ALG17]).

The LDSD algorithm helps in assessing the role of directly and indirectly connected entities in the performance of RS. The use of this algorithms can provide insights in how entity connectivity can influence the outcomes of recommendation. Passant [Pas10] found in his application of LDSD, that the recommendations also lead to unexpected, yet valid similarities. One of the possible enhancement the author identified, was to add weights to particular relations to give a higher or a lower importance to links (e.g. lessen the importance of geographical locations of two bands). For the experimental part of this thesis, we expect to capture nuanced relationships between entities and improve the accuracy of recommendations.

Resim

In 2016, Piao et al. [PAB16] proposed the Resource Similarity (Resim), a distance-based similarity measure which aims to overcome some of the limitations of LDSD and other proposed distance measures.

The Resim algorithm improves upon LDSD by satisfying the symmetry axiom of distance measures. To achieve this, the weighted incoming indirect distance $IC_i(r_a, r_b)$ between two resources r_a and r_b (formula 2.5) was adapted to the *symmetric weighted incoming indirect distance* $IC'_i(r_a, r_b)$:

$$IC'_i(r_a, r_b) = \sum_i \frac{C_{ii}(l_i, r_a, r_b)}{1 + \log(\sum_n \frac{C_{ii}(l_i, r_a, r_n) + C_{ii}(l_i, r_b, r_n)}{2})} \tag{2.8}$$

The weighted outgoing indirect distance $IC_o(r_a, r_b)$ between two resources r_a and r_b (formula 2.6) was similarly changed to the *symmetric weighted outgoing indirect distance* $IC'_o(r_a, r_b)$:

$$IC'_o(r_a, r_b) = \sum_i \frac{C_{io}(l_i, r_a, r_b)}{1 + \log(\sum_n \frac{C_{io}(l_i, r_a, r_n) + C_{io}(l_i, r_b, r_n)}{2})} \tag{2.9}$$

Analogous to formula 2.7, we reach the definition for the *symmetric Linked Data Semantic Distance (LDSD)* by combining the formulas 2.4, 2.8 and 2.9:

$$LDSD'(r_a, r_b) = \frac{1}{1 + DC(r_a, r_b) + DC(r_b, r_a) + IC'_i(r_a, r_b) + IC'_o(r_a, r_b)} \tag{2.10}$$

The similarity is computed as follows:

$$LDSD'(r_a, r_b)_{sim} = 1 - LDSD'(r_a, r_b) \tag{2.11}$$

The Resim algorithm also incorporates property similarity, which also includes resources only reachable over more than two links (see also Figure 2.8). For the property similarity, we first need the *shared incoming links* $C_{sip}(l_i, r_a, r_b)$ between two resources r_a and r_b. It equals 1 if an incoming link l_i that exists for both r_a and r_b:

$$C_{sip}(l_i, r_a, r_b) = \begin{cases} 1 & \exists r_c, r_d : \langle l_i, r_c, r_a \rangle \wedge \langle l_i, r_d, r_b \rangle \\ 0 & \text{otherwise} \end{cases} . \tag{2.12}$$

Similarly, the *shared outgoing links* $C_{sop}(l_i, r_a, r_b)$ between two resources r_a and r_b equals 1 if an outgoing link l_i that exists for both r_a and r_b:

$$C_{sop}(l_i, r_a, r_b) = C_{sip}(l_i, r_a, r_b) = \begin{cases} 1 & \exists r_c, r_d : \langle l_i, r_a, r_c \rangle \wedge \langle l_i, r_b, r_d \rangle \\ 0 & \text{otherwise} \end{cases} . \tag{2.13}$$

The Property similarity is then given by:

$$Property_{sim}(r_a, r_b) = \frac{\sum_i \frac{C_{sip}(l_i, r_a, r_b)}{\sum_n C_d(l_i, r_n, r_n)}}{C_{ip}(r_a) + C_{ip}(r_b)} + \frac{\sum_i \frac{C_{sop}(l_i, r_a, r_b)}{\sum_n C_d(l_i, r_n, r_n)}}{C_{op}(r_a) + C_{op}(r_b)}, \tag{2.14}$$

where C_{ip} denotes the sum of the incoming links for the node r_a:

$$C_{ip}(r_a) = \sum_i \text{ of the links } l_i \text{ incoming to } r_a \tag{2.15}$$

and C_{op} is the sum of the outgoing links for this node:

$$C_{op}(r_a) = \sum_i \text{ of the links } l_i \text{ outgoing from } r_a \tag{2.16}$$

The shared incoming and outgoing links are normalized by the total number of distinct instances of the links l_i between two resources r_a and r_b to give lower weight to properties appearing more often in the graph. As Piao et al. [PAB16] concluded to lead to better recommendation results as a normalization by the logarithm.

Given two weights ω_1 and ω_2, the weighted arithmetic mean of $Property_{sim}$ and $LDSD'_{sim}$ is then given by:

$$WPropLDSD(r_a, r_b) = \frac{\omega_1 \times Property_{sim}(r_a, r_b) + \omega_2 \times LDSD'_{sim}(r_a, r_b)}{\omega_1 + \omega_2}, \tag{2.17}$$

This leads to the final definition of Resim between two resources r_a and r_b, which equals 1 if the nodes share the same URI or if they are connected via the link `owl:sameAs`. In the other cases Resim is defined as the weighted arithmetic mean of $Property_{sim}$ and $LDSD'_{sim}$:

$$Resim(r_a, r_b) = \begin{cases} 1 & URI(r_a) = URI(r_b) \text{ or } r_a \ \texttt{owl:sameAs} r_b \\ WPropLDSD(r_a, r_b) & \text{otherwise} \end{cases}. \tag{2.18}$$

The resulting Resource Similarity (Resim) measure satisfies the symmetry property, to ensure – in contrast to LDSD – that the equation $Resim(r_a, r_b) = Resim(r_b, r_a)$ is valid.

In their study, Piao et al. [PAB16] were able to address some of LDSDs limitations and mitigate them with Resim. The Resim algorithm extends LDSD by including the similarity of entities with no direct or indirect relationship, providing a more comprehensive view of entity similarity in KGs. For the experiment in Chapter 7, we aim to evaluate whether the modifications introduced by Resim lead to improved recommendations or if they represent a trade-off between performance and accuracy.

2.7.2 Information Content-based Methods

While distance-based methods rely on the structure of the KG, information content-based metrics focus on the semantic information contained in the graph [MD16].

For this experiment, PIC-based Semantic Similarity (PICSS) as defined by Meymandpour and Davis [MD16]. was chosen to be evaluated on the graphs.

PICSS

The PICSS metric is based on Tversky's characterization of concepts [Tve77], measuring the similarity between two concepts based on commonalities between their features. In their approach, Meymandpour and Davis [MD16] expanded this method by incorporating a factor of importance using the Information Content (IC) of features.

Based on definition 2.7.1, features and resources in LOD are defined as follows:

Definition 2.7.2 (Features in LOD)**.** A feature f of a resource $r_a \in R$ is an instance of $l\langle l_j, r_b, D\rangle$, where r_b is the resource directly connected to r_a with the link $l_j \in L$ and D denotes the direction of the l_j (incoming/outgoing).

A resource $r_a \in R$ is therefore denoted as the set of its features F_{r_a}, defined as the union of the incoming and outgoing links:

$$F_{r_a} = F_{r_a}^{out} \cup F_{r_a}^{in} \tag{2.19}$$
$$F_{r_a}^{out} = \{\forall \langle l_i, r_i, Out\rangle, l_i \in L, r_i \in R \mid \exists \langle r_a, l_i, r_i\rangle \in \text{LD}\} \tag{2.20}$$
$$F_{r_a}^{in} = \{\forall \langle l_i, r_i, In\rangle, l_i \in L, r_i \in R \mid \exists \langle r_i, l_i, r_a\rangle \in \text{LD}\}. \tag{2.21}$$

As mentioned before, Meymandpour and Davis [MD16] added in the IC to give higher weight to less frequent features and les weight to more common ones. Based on the definition by Shannon [Sha48], who defined IC as the logarithm of the inverse of the probability of an event, it translates to LOD as follows:

Definition 2.7.3 (Information Content (IC) of features in LD)**.** The IC of a feature f in Linked Data (LD) is given by:

$$IC(f) = \log\left(\frac{1}{\pi(f)}\right) = -\log\left(\pi(f)\right), \tag{2.22}$$

where $\pi(f)$ denotes the probability of f.

Meymandpour and Davis [MD16] compute the probability of a feature f_i by using the ratio of the frequency of the feature $\phi(f_i)$ and the total number of features N:

$$\pi(f_i) = \frac{\phi(f_i)}{N}. \tag{2.23}$$

Because of the *multiplication rule of probabilities* for independent events, the probability π of the occurrence of two features f_1 and $_2$ holds the following equation:

$$\pi(f_1, f_2) = \pi(f_1)\pi(f_2). \tag{2.24}$$

Also regarding the *product rule of logarithm*:

$$\log(x \cdot y) = \log(x) + \log(y), \tag{2.25}$$

the IC for both features f_1 and f_2 is equal to the sum of their individual IC values:

$$\begin{aligned} IC(f_1, f_2) &= -\log(\pi(f_1, f_2)) \\ &= -\log(\pi(f_1)) - \log(\pi(f_2)) \\ &= IC(f_1) + IC(f_2). \end{aligned} \tag{2.26}$$

Regarding the properties 2.24 and 2.25, the probability for a set of features F_r can be written as the following:

$$\begin{aligned}\pi(F_r) &= \pi\left(f_1, f_2, \dots, f_{|f_r|}\right) \\ &= \pi(f_1)\pi(f_2)\cdots\pi\left(f_{|f_r|}\right) \\ &= \prod_{\forall f_i \in F_r} \pi(f_i) \end{aligned} \tag{2.27}$$

Combining the equations 2.22 and 2.27 leads to a definition for the IC of this set of features:

$$\begin{aligned} IC(F_r) &= -\log(\pi(F_r)) \\ &= -\log\left(\prod_{\forall f_i \in F_r} \pi(f_i)\right) \\ &= \sum_{\forall f_i \in F_r} -\log(\pi(f_i)). \end{aligned} \tag{2.28}$$

This results in the definition for the Partitioned Information Content (PIC).

Definition 2.7.4 (Partitioned Information Content (PIC) in LD)**.** Let r be a resource in LD. The Partitioned Information Content (PIC) of this resource can be described as the sum of the Information Content (IC) for all of its features $f_i \in F_r$:

$$\begin{aligned} PIC(r) &= \sum_{\forall f_i \in F_r} IC(f_i) \\ &= \sum_{\forall f_i \in F_r} -\log\left(\frac{\phi(f_i)}{N}\right). \end{aligned} \tag{2.29}$$

Finally, combining Tversky's similarity of two concepts [Tve77] and the adjustments using the IC of resources, Meymandpour and Davis [MD16] concluded in the following definition for their proposed PIC-based similarity measure:

Definition 2.7.5 (PICSS for LD)**.** The similarity of two resources $r_a \in R$ and $r_b \in R$, represented by their set of features F_{r_a} and F_{r_b} is defined as follows:

$$PICSS(r_a, r_b) = \frac{PIC(F_{r_a} \cap F_{r_b})}{PIC(F_{r_a} \cap F_{r_b}) + PIC(F_{r_a} - F_{r_b}) + PIC(F_{r_b} - F_{r_a})}. \tag{2.30}$$

The PICSS measure normalizes similarities between 0 and 1, with 0 representing no similarity and 1 standing for perfect similarity. The measure is not a metric in a mathematical definition[12], because it does not satisfy the triangle inequality, meaning

[12]A function d is a metric, if it satisfies the following axioms:

if two resources r_a and r_b are similar and r_b is also similar to another resource r_b, it is not given that r_a is also similar to r_c. However, similar to Resim and contrary to LDSD (see Section 2.7.1), PICSS does satisfy the symmetry property $PICSS(r_a, r_b) = PICSS(r_b, r_a)$.

By weighting features based on their informativeness, we expect PICSS to focus on the most important features. Meymandpour and Davis [MD16] reported on a higher accuracy compared to other LOD-based methods, though they also experienced a sensibility towards noise in their experiments.

2.7.3 Knowledge Graph Embeddings

Knowledge Graph Embedding (KGE) is an approach to convert the entities of a KG into a set of sequences [RP16a]. This technique reduces the dimension by transforming the data into a continuous vector space, while still capturing the semantic information [YYH+14]. Once the embeddings are trained, they can be directly used by machine learning models and due to their independence on specific tasks, even be reused. This low-dimensional representation allows the use of a variety of machine learning tasks [RP16a].

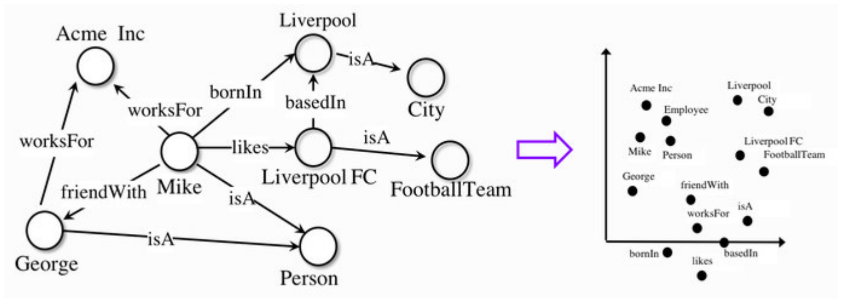


Figure 2.9: Visualization of the process to transform a graph into a low-dimensional vector representation (image taken from https://docs.ampligraph.org/en/1.1.0/, accessed 02.08.2024). We can see that "Liverpool" and "City" appear close to each other in the vector space, indicating a relatedness.

We used three different embedding algorithms: RDF2Vec, TransE and DistMult. These models have been selected due to their widespread use and effectiveness in prior research. The following sections provide a basic description of the algorithms, for a more detailed explanation, the reader should refer to the cited literature.

- Non-negativity: $d(a, b) \geq 0$
- Coincidence-axiom: $d(a, b) = 0$ if and only if $a = b$
- Symmetry: $d(a, b) = d(b, a)$
- Triangle inequality: $d(a, c) \leq d(a, b) + d(b, c)$

RDF2Vec

RDF2Vec was first introduced by Ristoski and Paulheim [RP16a] in 2016. The algorithm leverages methods from natural language processing to represent a KG as a low-dimensional vector. The graph data is first transformed into sequences of entities to enable the use of neural language models to generate the embeddings [RP16a]. The authors propose two approaches for generating the sequences of entities: random graph walks and the Weisfeiler-Lehmann Subtree RDF Graph Kernels.

To generate random walks P_r of depth d for each resource $r \in R$ by exploring outgoing links and their connected resources $r \to l_1 \to r_1 \to l_2 \to r_2 \to \ldots$, with $l_i \in L$ by using the breadth-first search algorithm.

The Weisfeiler-Lehmann Subtree RDF Graph Kernels creates subtrees of depth d for each resource $r \in R$ with the pattern $r \to T_1 \to T_2 \to \ldots \to T_d$, where T_d is a subtree that appears on depth d. This is repeated up to a maximum of h iterations that leads to a final set of sequences, which is the union of the sequences for all resources: $\bigcup_{i=1}^{h} \bigcup_{r \in R} P_r$.

The sequences of one of the previous approaches is then used to train the embeddings with the natural language model word2vec, either using the Continuous-Bag-of-Words (CBOW) or the skip-gram model. With a window size c, the CBOW model predicts a target resource $r_t \in R$ by maximizing the average log probability:

$$\frac{1}{T} \sum_{t=1}^{T} \log p(r_t \mid r_{t-c}, \ldots, r_{t+c}), \tag{2.31}$$

where the softmax function is used to calculate the probability:

$$p(r_t \mid r_{t-c}, \ldots, r_{t+c}) = \frac{\exp(\mathbf{v}_t^T \mathbf{v}')}{\sum_{r \in R} \exp(\mathbf{v}_r^T \mathbf{v}')}, \tag{2.32}$$

with the averaged input vector $\mathbf{v}_t$ for the context resources, and the output vector for the target resource $\mathbf{v}'$.

The skip-gram model is the reverse of the CBOW, as it tries to predictthe context resources for given a target resource, by maximizing the log probability:

$$\frac{1}{T} \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log p(r_{t+j} \mid r_t), \tag{2.33}$$

where the probability $p(r_{t+j} \mid r_t)$ is similarly calculated using the softmax function.

In [RRDN+19], Ristoski et al. compared RDF2Vec to other embedding methods, including TransE (but not DistMult) and were able to outperform many state-of-the-art algorithms. For our experiment, we hope to see, if the differing performances of TransE and RDF2Vec are universally appearing in all KGs or only in specific settings of KG characteristics.

TransE

TransE, which belongs to a group of translational embedding models, represents relations as translations in the embedding space [KKS21]. This model was proposed by Bordes et al. [BUGD+13] and each resource $r \in R$ is represented as vector $\mathbf{r} \in \mathbb{R}^k$ and each link $l \in L$ is represented as a vector $\mathbf{l} \in \mathbb{R}^k$, with the hyper parameter k.

For a triple $\langle l_j, r_a, r_b \rangle$ TransE tries to satisfy $\mathbf{r_a} + \mathbf{l_j} \approx \mathbf{r_b}$, means that the vector for the target resource r_b should be close to the vector for the source resource r_a translated by the link vector l_j.

To learn the embeddings, the margin-based ranking criterion is minimized over the training set:

$$L = \sum_{(r_a,l_j,r_b)\in I} \sum_{(r'_a,l_j,r'_b)\in I'_{(r_a,l_j,r_b)}} [\gamma + d(\mathbf{r_a} + \mathbf{l_j}, \mathbf{r_b}) - d(\mathbf{r'_a} + \mathbf{l_j}, \mathbf{r'_b})]_+ \,, \tag{2.34}$$

with the dissimilarity measure d (usually the L_1 or L_2-norm), $[x]_+$ as the positive part of x, the margin-parameter γ. I' represents the set of corrupted instances generated by replacing either the source r_a or target resource r_b with a random resource r'_a or r'_b.

TransE represents relationships between entities as translations in the embedding space, with a primary focus on hierarchical structures [BUGD+13]. In their experiments, Bordes et al. [BUGD+13] did not perform a recommendation task but they were able to outperform other methods in a link prediction task (note that TransE was proposed in 2013, so before most of the other well-known embedding models, as Figure 1.2 shows).

DistMult

The DistMult model, as introduced by Yang et al. [YYH+14] in 2014, uses a diagonal matrix to capture the interactions between entities and relations. Similar to TransE, the resources and links are represented as vectors $\mathbf{r} \in \mathbb{R}^k$ and $\mathbf{l} \in \mathbb{R}^k$. The model considers the bilinear scoring function

$$\text{score}(l_j, r_a, r_b) = \mathbf{r}_a^\top \cdot \text{diag}(\mathbf{l}_j) \cdot \mathbf{r}_b, \tag{2.35}$$

where $\text{diag}(\mathbf{l}_j)$ is a diagonal matrix of the link vector $\mathbf{l}_j$ in $\mathbb{R}^{k\times k}$.

The model uses a margin-based ranking loss to optimize the embeddings so that correct triples are scored higher than incorrect ones:

$$\sum_{\langle l_j,r_a,r_b\rangle\in T} \sum_{\langle l_j,r'_a,r'_b\rangle\in T'} \max(0, \gamma + \text{score}(l_j, r'_a, r'_b) - \text{score}(l_j, r_a, r_b)), \tag{2.36}$$

where T represents the set of positive triples, T' the set of corrupted triples and the hyper parameter γ.

As opposed to TransE, DistMult uses a bilinear scoring function to capture pairwise interactions between entities and relations [YYH+14]. Yang et al. [YYH+14] compared their proposed algorithm to TransE, also in the setting of a link prediction task and were able to achieve a better performance on DistMult.

CHAPTER 3

State of the Art

In this chapter, the work related to the thesis are reviewed and discussed. Section 3.1 focuses on research on the characteristics of a Knowledge Graph (KG), which is relevant for the first research question. For the second research question, Section 3.2 summarizes existing surveys on KG-based Recommender Systems (RS).

The practical part of this thesis, addressing the third research question, consists of an experiment using different types of algorithms. Section 3.3 reviews studies on the embedding-based algorithms that will be used in the scope of this work. Research on distance-based and information content-based methods are outlined in Section 3.4.

3.1 Characteristics of Knowledge Graphs

Cheng et al. [CWX$^+$15] studied different KGs and social networks based on their characteristics to help selecting appropriate technologies, especially for specific storage and indexing strategies. For this, thirteen statistics and four statistical distributions are analyzed for benchmarking and comparing KGs.

Zaveri et al. [ZRM$^+$16] performed a systematic review to study the quality of Linked Data (LD). In this survey, the authors provide a list of 18 qualitative dimensions and 69 metrics to assess the data quality. The quality dimension are categorized into four groups: accessibility, intrinsic, contextual and representations dimensions. As a result the study provides a framework for the quality assessment of LD, where it is stated for each metric whether they can be measured quantitatively or qualitatively.

A similar framework for the data quality of a KG is proposed by Färber et al. in [FBMR18]. This study provides an adapted framework from Zaveri et al. [ZRM$^+$16] for analyzing and comparing different KGs to determine the best fit for a given use case. The authors also provide key statistics of the KGs DBpedia, Freebase, OpenCyc, Wikidata and YAGO. Färber and Rettinger also provided a simplified summary of this framework in [FR18] with rules of thumb for a reasonable KG selection.

In [HHRP20], Heist et al. compared multiple publicly available KGs according to different measures, including the amount of contained information and metrics which are defined for the schema or ontology level. One of the key findings of this study is a significant variation in size and coverage of the different KGs. Due to the common origin for many publicly available KGs from Wikipedia, the authors reported on a high degree of linkage and complementary information between different KGs (see Figure 3.1). This suggests that a combination of multiple KGs can be beneficial for specific application that require a broad coverage of a domain.

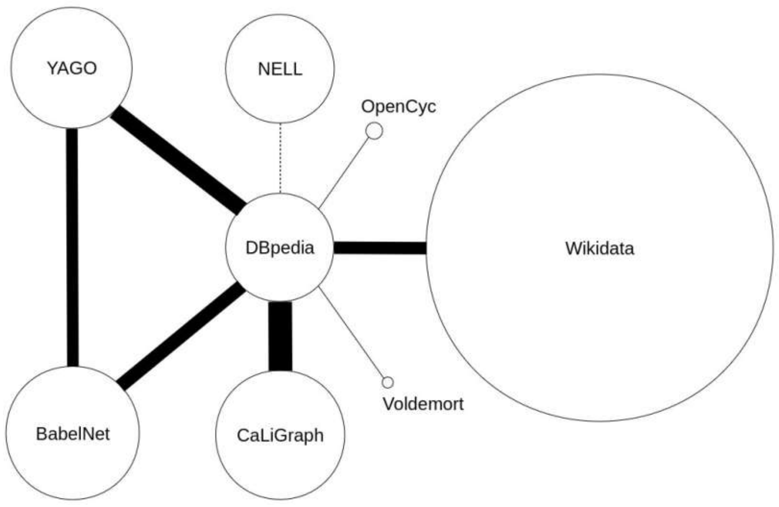


Figure 3.1: Depiction of the size of different KGs and their degree of linkage to one another. The size of the circles relates to the number of instances in the KG and the line strength is depicts the number of links from one to another. NELL is not directly linked to DBpedia, but links to Wikipedia can be translated to DBpedia links (from [HHRP20]).

Heinrich et al. [HHL$^+$21] investigated the influence of the completeness data quality in the performance of RS. The theoretical model of the authors is tested on two real-world datasets with the conclusion, that an increased completeness positively impacts the performance of recommendations, particularly by adding features and their values as well as adding missing feature values.

3.2 Surveys on Knowledge Graph-based Recommender Systems

There are some researchers who created collections of surveys regarding KG-based Recommender Systems (RS). In their survey. Ristoski and Paulheim [RP16b], explored the research field and possibilities of leveraging Linked Open Data (LOD) in data mining and machine learning applications and evaluated a RS as an example use case. The authors conclude that many evaluated papers used custom-built ontologies and do not make use of open data from the web.

Guo et al. [GZQ$^+$20] conducted a systematic survey to investigate the utilization of KGs for accurate and explainable recommendation. The authors collected published papers and grouped them into one of three categories: embedding-based, connection-based and propagation-based methods which were then further divided according to the specifics of the studied experiments. The resulting collection of this survey not only summarize the datasets, external Knowledge Base (KB) (such as DBpedia), type KG (item- or user-based) but also the basic characteristics of the used datasets: the number of users and items as well as the overall density.

Chicaiza and Valdiviezo-Diaz [CVD21] conducted another survey on this topic that provides a similar overview of 38 studies in various application domains. The studied papers were divided into two groups of technologies and later into the level of development as well as the type of evaluation. In at least 8 of the reviewed papers the authors gathered experiments which reported on improved performances against baseline or traditional models. Chicaiza and Valdiviezo-Diaz do not report on any KG characteristics but they mention a shortage of proposals which include implementation and evaluation parts of their proposals, with only 43% of the papers containing references or descriptions.

3.3 Knowledge Graph Embeddings

Piao and Breslin [PB18] studied the performance of multiple deep learning and Knowledge Graph Embedding (KGE)-based methods (including TransE) on different aspects of DBpedia for the music and book domain. They focused on three different aspects of knowledge: textual information, homogeneous graphs and heterogeneous graphs (Figure 3.2. Textual knowledge was derived from entities such as the abstract of a movie. A homogeneous graph contains knowledge derived from Wikipedia[1] with the property `dbo:wikiPageWikiLink`, which represent hyperlinks in the texts of articles[2]. The knowledge in a heterogeneous graph consists of domain-specific entities such as *starring* or *writer* as pictured in Figure 3.2.

The authors of this study learned embeddings from these aspects of a KB and focused on the extent to which they reflect the similarities between the entities. The results showed that embeddings learned from different subgraphs or combined embeddings can capture different types of similarities among the entities with varying performances on the datasets. The best performance in this experiment was observed by using the homogeneous graph powered by the `dbo:wikiPageWikiLink` property. A concatenation of all embeddings however, did not show a performance improvement of the recommender system.

[1]https://wikipedia.org
[2]https://dbpedia.org/ontology/wikiPageWikiLink

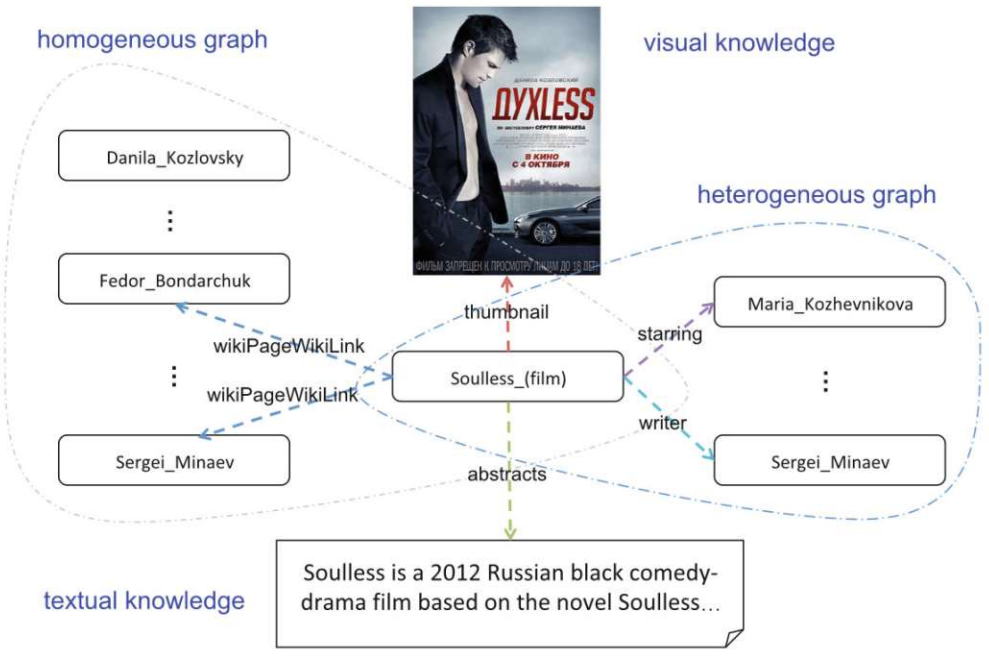


Figure 3.2: Different aspects of knowledge from a movie of DBpedia (from [PB18])

3.3.1 TransE

In 2013, as one of the first modern KGE method (see Figure 1.2), Bordes et al. [BUGD+13] proposed TransE, an approach to learn embeddings by interpreting the relationships in the KG as translations operating on the entity embeddings. The authors evaluated this algorithm by predicting links in the KGs but also mention RS as a possible field of interest.

Piao and Bresling [PB18] also used the TransE algorithm for their experiments on the homogeneous and heterogeneous graph. They observed that a concatenation of embeddings learned with TransE and those learned with `Doc2Vec` from textual knowledge resulted in an imporvement of the performance as compared to using TransE alone.

3.3.2 DistMult

Shortly after TransE, Yang et al. proposed DistMult in the year 2014 [YYH+14] which extends the previous method to bilinear mapping functions (see also Chapter 2.7.3). The authors evaluated different embedding models and found, that their proposed algorithm can outperform other models such as TransE in knowledge inference tasks.

3.3.3 RDF2Vec

In [P+13], Paulheim studied the augmentation of datasets with features from LOD and reported an improvement of the results of data mining problems for several use cases.

Since many data mining methods require a feature vector representation of the data and LOD are graphs, Ristoski and Paulheim [RP16a] proposed RDF2Vec as an approach to represent each entity as a vector of latent numerical features. The authors adapted the language modeling approach `Word2Vec`, by converting a graph into a set of sequences of entities and training them with this method. The resulting vectors are a representation of the graph as numerical features due to this projection, similar entities appear closer to each other (see also Chapter 2.7.3). These vector representations were also validated with a visualization of a small number of vectors on which Principal Component Analysis (PCA) was performed to project them in a two dimensional feature space. The authors experimented with various small and large RDF datasets, performing classification and regression learning tasks, measuring the performance using accuracy (classification) and root mean squared error (RMSE) (regression). The vector representation using RDF2Vec outperformed standard feature generation approaches in their evaluation.

Rosati et al. [RRDN+16] used the RDF2Vec approach to generate feature vectors of entities in RDF graphs to build a content-based RS. The experiment was performed using DBpedia and Wikidata on two different domains, movies and books. The authors compared this method to baseline strategies for feature extraction as well as to collaborative filtering algorithms. In the case of the dataset of the book domain, the content-based method of this study was able to outperform the collaborative-filtering RS.

In his Ph.D. thesis [Ris19], Ristoski studied the use of KGE as item representations from LOD for RS. The proposed method approach of RDF2Vec was further continued in [RRDN+19] were it was able to outperform state-of-the-art content-based as well as collaborative and hybrid RS.

In Ristoski et al. [RRDN+19] – an extension of the two aforementioned studies and the thesis – the RDF2Vec approach is evaluated on three different tasks, including content-based recommender systems (as did Rosati et al. [RRDN+16]), but also on one additional domain. As opposed to [RRDN+16] the top-N recommendation was not only performed with content-based but also with hybrid RS. Additionally, the authors also compared their algorithm to state of the art KGE algorithms, namely TransE, TransH and TransR. The authors provide the characteristics of the resulting KGs and compared the performance of the algorithms with Precision@N, Recall@N, the F-measure and the normalized Discounted Cumulative Gain (nDCG)@N. Although RDF2Vec outperformed traditional or other embedding methods in some cases, the authors conclude that there is not one best approach that fits every variant. Nonetheless, this study shows a wide variety of use cases of KGE and that they are universal, meaning they have to be built only once and can then be re-used for different tasks.

3.4 Traditional Algorithms for Knowledge Graph-based Recommender Systems

Formica and Taglino conducted a study in 2023 ([FT23]), where they compared 10 knowledge-based methods to assess semantic relatedness of resources, using 14 datasets on the same release of DBpedia. This approach was chosen to ensure comparability of the methods, addressing the issue identified in their survey that different DBpedia releases used in the literature affect the reliability of comparisons of the results in existing experiments. The authors selected methods based on adjacent resources, triple patterns (distance-based measures, see 2.7.1) and triple weights to conduct their survey. Furthermore, the influence of the predicate `dbo:wikiPageWikiLink` was examined as well by performing the experiment on two DBpedia data collections, one containing triples with this predicate and one without (with sizes 61 GB and 33 GB respectively). Due to the size of these KGs, the usage Knowledge Graph Embedding (KGE) were out of scope for this survey.
The authors conclude, that their experiment with methods based on triple weights, which contain information content-based measures (see Section 2.7.2), show the best overall performance. They also highlight the importance of the triples containing the `dbo:wikiPageWiki` predicates, as the Spearman's and Pearson's correlation to the golden datasets increases significantly compared to the KG without these triples. This is a similar observation as from Piao and Bresling in [PB18] who also reported on performance improvements of a graph containing this predicate.

3.4.1 Distance-based Methods

Piao and Breslin [PB16] explored various distance metrics based on Linked Data Semantic Distance (LDSD) (see also Chapter 2.7.1) and compared their performances in the music domain to other distance-based RS. Passant [Pas10] also compared different extensions of LDSD to various domains, but neither of these studies additionally use KGEs.

Because LDSD violates axioms of similarity measures in a mathematical sense, Piao et al. [PAB16] propose Resource Similarity (Resim) (see also Chapter 2.7.1), which overcomes this shortcoming. Additionally, the authors also consider the similarity of properties to include relationships of resources which would not be available otherwise and lead to an increasing performance of calculating similarities for top-N recommendations in the music domain.

The LDSD algorithm only considers resources that are directly or indirectly (through a resource in between) as relates. To expand the distance measure to include resources further away from each other, Alfarhood et al. proposed Propagated Linked Data Semantic Distance (PLDSD). By increasing this ability they were able to obtain a higher accuracy in their experiment as with LDSD and Resim.

3.4.2 Information Content-based Methods

Meymandpour and Davis [MD16] proposed the PIC-based Semantic Similarity (PICSS) (see also Chapter 2.7.2) measure which combines feature and information content-based approaches, based on the the similarity ratio model by Tversky [Tve77]. The authors evaluated the model on two different movie datasets: MovieLens100K and MovieLens1M and used conventional and state-of-the-art RS as baseline models. Although the PICSS model outperformed the baseline models, a comparison to embedding-based RS was not explored.

3.5 Conclusion

From the summarized literature we can grasp an interest in KGs and RS across various domains and applications.

There have been some surveys conducted which address the characteristics of KGs: Cheng et al. [CWX+15] focused on statistical characteristics and distributions, while Zaveri et al. [ZRM+16], Färber et al. [FBMR18] as well as Färber and Rettinger [FR18] focus on quality characteristics. Heist et al. [HHRP20] focused on the linkage of publicly available KG and Heinrich et al. [HHL+21] investigtated the completeness characteristic.

Regarding surveys specializing on RS with KGs we also highlighted three research groups: Ristoski and Paulheim [RP16b], Guo et al. [GZQ+20] and Chicaiza and Valdiviezo-Diaz [CVD21]. However, none of these survey evluated a possible influence of KG characteristics on the performances.

We also explored proposals of algorithms of both embedding-based and traditional methods and the results of their experimental applications. The survey from Formica and Taglino [FT23] identified the issue of the comparability of knowledge-based methods due to the use of different KG releases.

As the literature highlights, there has been identified some importance to the characteristics of KGs and the impact they have on certain applications. However, there is still a gap in examining them in the context of RS and their influence on the performance using various algorithms of different types to compare.

CHAPTER 4

Methodology

This chapter outlines the methodological approach chosen for this thesis. First, it covers the theoretical background following established guidelines from the literature and then describes it in the context of the thesis.

4.1 Theoretical Background

4.1.1 Semi-systematic Review

The first two questions are explored using a semi-systematic (also called narrative review) process, which follows the guidelines by Snyder [Sny19]:

Phase 1: Designing of the review
In the first step appropriate search engines and search terms need to be selected. The type of papers that are regarded for the review should also be defined.

Phase 2: Conducting the review
Using the defined strategy, the second phase consists of the review itself. The search process is further divided into different steps to reduce initial findings to a smaller number of relevant works. A protocol should be written to make the process transparent.

Phase 3: Analyzing the results
After the conduction of the review, the chosen research papers have to be analyzed and the relevant information extracted.

Phase 4: Conclusion
In the last step, the findings should be summarized according to their relevance to the research question.

4.1.2 Experiment Process

The experimental part that is necessary to explore the third research question, follows the experimentation guidelines by Wohlin et al. [WRH+12]:

Phase 1: Experiment scoping
In the first phase, the experiment needs to be scoped in terms of the problem, objective and goals. The hypothesis for the experiment already needs to be clear in this step.

Phase 2: Experiment planning
This phase lays the foundation for the practical part as it is the design of the experiment itself. The hypothesis should be stated formally and the variables need to be determined.

Phase 3: Experiment operation
The third phase is the execution of the experiment according to the design of the previous step.

Phase 4: Analysis and interpretation
In this step, the outcome of the operation phase should be analyzed and interpreted. This contains descriptive statistics and hypothesis testing.

4.2 Research Question 1: Quantifiable Characteristics of KGs

RQ1 *What are the defining characteristics of a KG that distinguishes it from another KG? How can these characteristics be quantified?*

To explore the first research question, an analysis of quantifiable characteristics of a Knowledge Graph (KG) will be performed following the process of a semi-systematic review as outlined before. The findings from this analysis will later be used in the implementation phase (7).

Chapter 5 addresses the first research question by exploring literature regarding this topic.

4.3 Research Question 2: Determine Most Used KGs

RQ2 *What KGs are commonly used to evaluate KG-based recommender algorithms? What are their characteristics?*

The primary objective of the second research question is the semi-systematic review of current research in the types of KGs used for Recommender Systems (RS) algorithms. The main focus should be on KGs that are publicly available to allow repeatability and accessibility. The outcome of this study will impact the selection of KGs and dataset that would be used in Chapter 7.

Chapter 6 explores the process of the review for the second research question.

4.4 Research Question 3: Influence of KG Characteristics on Recommender Systems

RQ3 *How do the RS algorithms perform on KGs with distinctive characteristics? What influence do certain KG characteristics have on the performance of a family of recommendation algorithms?*

Scoping

The main goal is to use the insights from the first two research questions and conduct a controlled experiment to explore the influence of KG characteristics on the performance of RS [WRH+12]. In Chapter 2.7 the algorithms that will be used are explained in detail, which lays the theoretical foundation for the experiment in Chapter 7.

Planning

The objectives are to evaluate the performances of the algorithms, identify influential characteristics, comparing the methods and formulate guideline based on the conclusions.

The hypotheses for the experiment follow the evaluation of the influence of KG characteristics on the performance of RS. The dataset, characteristics and KBs to use will be selected according to the findings of the first two research questions.

The independent variables includes the characteristics of the created KGs. The dependent variable is the performance of the algorithms, measured by the metrics Precision@N (formula 7.1), Recall@N (formula 7.2) and the F-measure (formula 7.3), which are useful measures to acquire the accuracy of RS [GS15].

Operation

The procedure of the experiment consists of implementing the algorithms, building the KGs with varying characteristics based on the dataset and the KBs, produce recommendations, evaluate the performance and compare the results to draw conclusions. This approach is visualized in Figure 1.3.

Interpretation

Using statistical analysis, consisting of descriptive statistics and hypothesis testing, the results of the experiment are summarized and interpreted.

CHAPTER 5

Key Characteristics of Knowledge Graphs

Before diving into to the practical part, it is essential to understand the characteristics of a Knowledge Graph (KG) by addressing the first research question:

RQ1 *What are the defining characteristics of a KG that distinguishes it from another KG? How can these characteristics be quantified?*

The exploration of this question follows a semi-systematic review consisting of the four phases as described in Section 4.1.1. The following sections provide a detailed report of each phase in the review process.

5.1 Design of the Review

To identify potentially relevant papers, we used the Semantic Web – Interoperability, Usability, Applicability (in short Semantic Web journal)[1], ACM Digital Library[2] and Google Scholar[3] as search engines.

- **Semantic Web journal:** Covers topics related to the semantic web, making it a suitable search engine.
- **ACM Digital Library:** Offers a broad collection of computer science topics, providing a more comprehensive search than the Semantic Web journal.
- **Google Scholar:** Used for its extensive coverage to find additional resources not indexed by the other two search engines.

[1]https://semantic-web-journal.net/
[2]https://dl.acm.org/
[3]https://scholar.google.com/

The used search terms used across all of the search engines were:

- "Knowledge graph characteristics"
- "Linked data quality"
- "Knowledge graph statistical characteristics"

Peer-reviewed articles, conference papers, and surveys were pre-selected and included if they indicated relevance to the research question. The Semantic Web journal employs an open review process, so only studies which received the status "Accept" after reviewing were considered potential findings.

Given that the topic is quite specific, the aim was to gather 5-10 papers per search term and as an additional strategy, the references in selected surveys were also scanned to further increase the number of potentially relevant papers. The Semantic Web journal's focus on specialized topics were expected to yield fewer results, so the target was to find 1-5 papers.

5.2 Execution of the Review

5.2.1 Performing the Search

The first selection of papers was made based on the title and type alone, the refinement is documented in Section 5.2.2. For each search term, the Semantic Web journal was searched first, followed by the ACM Digital Library and finally Google Scholar. Duplicates were already disregarded on the initial search and added to the result count of the first search engine where they were found.

The first search term "Knowledge graph characteristics" lead to various results on the search engines, Table 5.1 summarizes the breakdown of the 15 findings in total.

	Article	Conference Papers	Survey	Total
Semantic Web Journal	1		2	3
ACM Digital Library		4	1	5
Google Scholar	3	3	1	7

Table 5.1: Breakdown of a total of 15 found papers across different databases for the search term "Knowledge graph characteristics"

The second search term "Linked data quality" resulted in 19 initial findings, which are summarized in Table 5.2.

The last search term "Knowledge graph statistical characteristics" lead to an additional 15 papers, the summary can be obtained from Table 5.3.

	Article	Conference Papers	Survey	Total
Semantic Web Journal	2		2	4
ACM Digital Library	3	8		11
Google Scholar	3		1	4

Table 5.2: Breakdown of a total of 19 found papers across different databases for the search term "Linked data quality"

	Article	Conference Papers	Survey	Total
Semantic Web Journal			1	1
ACM Digital Library	1	5	2	8
Google Scholar		5	1	6

Table 5.3: Breakdown of a total of 15 found papers across different databases for the search term "Knowledge graph statistical characteristics"

5.2.2 Refining the Initial Selection and Additional Considerations

To refine the initial selection, the abstracts of the pre-selected papers were skimmed to determine their relevance. The considered papers from surveys of the regarded options were further searched for additional sources and added to the list after with the same selection criteria as the initial select. The refined selection can be found in Table 5.4.

	Article	Conference Papers	Survey	Total
Semantic Web Journal	2		4	6
ACM Digital Library	3	10	1	14
Google Scholar	5	5	2	12
Additional Considerations	1			

Table 5.4: Breakdown of a total of 33 papers across different databases that were further regarded after reading the abstracts of the initial selections. One additional consideration was obtained from the survey papers.

5.2.3 The Final Selection of Considerations

For the final selection of the relevant literature the 33 papers (see Table 5.4) were reviewed using the following process:

1. **Skimming the outline:** reading the abstract again, followed by the introduction and the conclusion.
2. **Examining the key sections:** focus on the methods, the discussion and the contributions.

3. **Evaluating the relevance:** asses the possible contribution to the research question.

After each step a decision is made to either proceed with the next step or drop the paper. The final selection can be found in Table 5.5.

	Article	Conference Paper	Survey
Semantic Web Journal	[RMGCGP18]		[ZRM+16] [FBMR18] [MLU22]
ACM Digital Library	[HFKP20] [DAL16] [MT18]	[MMB12] [KWA+14] [WK17] [HM18] [FH11]	
Google Scholar	[GLX+19]	[YHXL22] [CWX+15] [CCCD19]	[Had18] [XZ22]
Additional Considerations	[PB14]		

Table 5.5: Final selection of 19 papers from different databases for each search term after refining and adding additional considerations

The publication years of these works span from the year 2011 to 2022, where the most papers have been published in 2018 (see Figure 5.1).

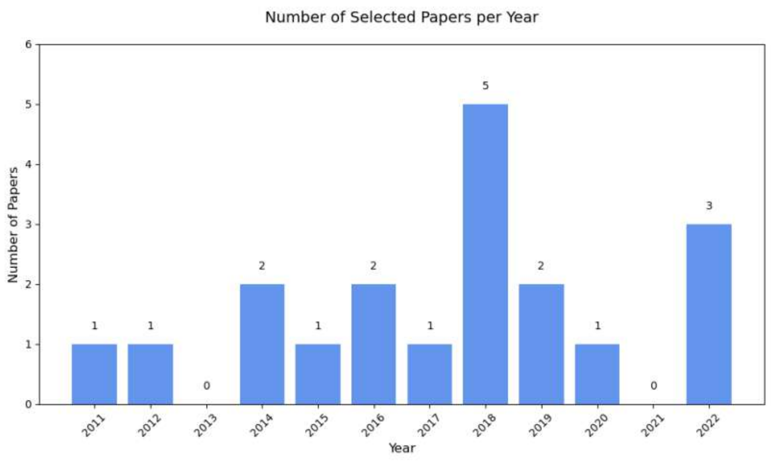


Figure 5.1: Selected papers per year for the first research question

Figure 5.2 visualizes the review process, leading to a total of 19 papers which were analyzed and summarized.

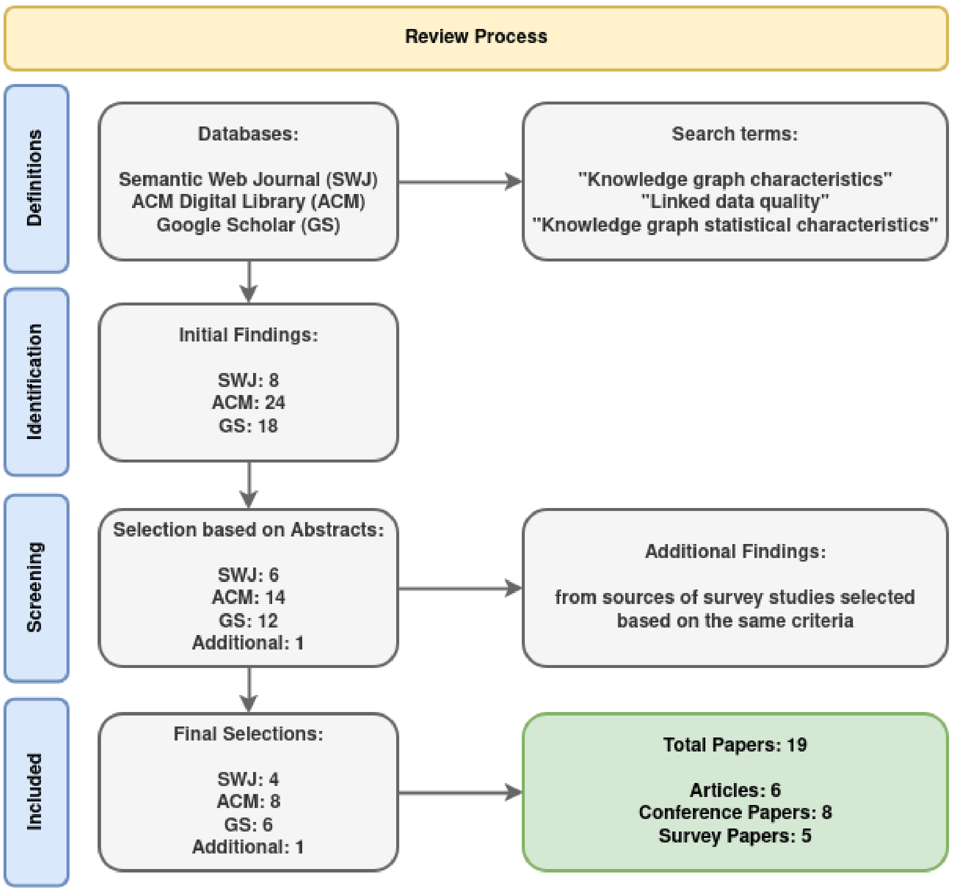


Figure 5.2: Summary of the review process for RQ1. Adapted from the flow diagram from the PRISMA guidelines for systematic reviews (https://www.prisma-statement.org/, accessed 05.08.2024)

5.3 Analysis of Quantifiable Knowledge Graph Characteristics

After selecting the relevant papers (see Table 5.5), four main categories of characteristics have been identified: quality characteristics, statistical characteristics, statistical distributions and characteristics obtained from graph theory.

The following sections describe each of them in detail.

5.3.1 Quality Characteristics

Quality characteristics determine the overall quality, the "fitness for use" of the underlying data for a specific use case [ZRM$^+$16]. Zaveri et al. [ZRM$^+$16] performed a systematic review of existing quality measures for Linked Data (LD) and divided them into four main groups (visualized in Figure 5.3):

- Accessibility dimensions
- Intrinsic dimensions
- Contextual dimensions
- Representational dimensions

The authors also distinguish between quantitative and qualitative metrics, i.e. those that are evaluated based on personal perception.

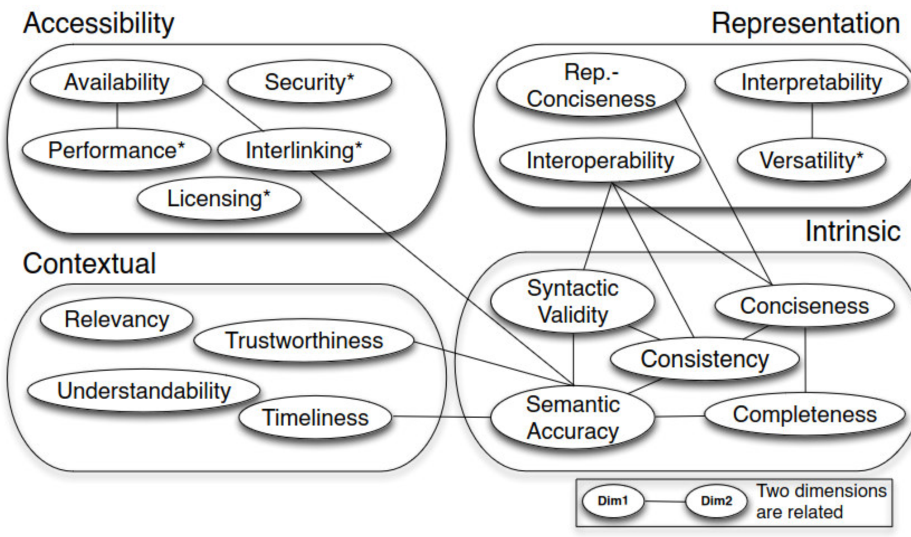


Figure 5.3: LD quality dimensions, where the ones marked with a "*" are specific for LD (from [ZRM+16]). Some of the dimensions are related to one another. For instance, a KG that uses invalid re-usage of vocabularies (interoperability) can lead to data inconsistencies (consistency) [ZRM+16].

This categorization was evaluated (Debattista et al. [DAL16], Weichselbraun and Kuntschik [WK17], Mendes et al. [MMB12], Kontokostas et al. [KWA+14]), explored (HG et al. [HM18], Hadhiatma [Had18], Färber et al. [FBMR18], Mountantonakis and Tzitzikas [MT18], Fürber and Hepp [FH11], Xue and Zou [XZ22]) and expanded (Radulovic et al. [RMGCGP18], Chen et al. [CCCD19], Haller et al. [HFKP20]) by other researchers.

Yang et al. [YHXL22] also handle noise and data sparsity, while Haller et al. [HFKP20] expand further on the different types of links. The most important characteristics in the scope of this thesis are summarized in Table 5.6.

Dim.	Category	Type	Reference
A	Availability	QN	[ZRM+16], [RMGCGP18], [CCCD19], [FBMR18], [DAL16], [WK17], [HFKP20]
A	Licensing	QN	[ZRM+16], [RMGCGP18], [FBMR18], [DAL16], [WK17]
A	Interlinking	QN	[ZRM+16], [CCCD19], [FBMR18], [DAL16], [WK17], [MT18], [HFKP20]
A	Security	QN	[ZRM+16], [DAL16], [WK17]
A	Performance	QN	[ZRM+16], [RMGCGP18], [CCCD19], [WK17]
A	Dereferenceability	QN	[RMGCGP18], [DAL16], [HFKP20]
A/I	Accuracy	QN	[RMGCGP18], [GLX+19], [Had18], [FBMR18], [MT18], [FH11]
I	Syntactic validity	QN	[ZRM+16], [CCCD19], [WK17], [FH11]
I	Semantic accuracy	QN	[ZRM+16], [DAL16], [WK17], [FH11]
I	Consistency	QN	[ZRM+16], [RMGCGP18], [CCCD19], [Had18], [FBMR18], [DAL16], [WK17], [MMB12], [FH11], [XZ22]
I	Conciseness	QN	[ZRM+16], [CCCD19], [DAL16], [WK17], [MMB12], [XZ22]
C/I	Completeness	QN	[ZRM+16], [RMGCGP18], [CCCD19], [Had18], [FBMR18], [WK17], [MMB12], [FH11], [XZ22]
I	Compliance	QN	[RMGCGP18]
C	Relevancy	QN	[ZRM+16], [CCCD19], [FBMR18], [WK17], [MT18], [KWA+14]
C/I	Trustworthiness	QN/QL	[ZRM+16], [RMGCGP18], [CCCD19], [FBMR18], [WK17]
C/R	Understandability	QN/QL	[ZRM+16], [RMGCGP18], [CCCD19], [FBMR18], [DAL16], [WK17]
C	Timeliness	QN	[ZRM+16], [RMGCGP18], [CCCD19], [FBMR18], [DAL16], [WK17], [MMB12], [MLU22], [FH11], [XZ22]
C	Diversity	QN	[CCCD19]
R	Representational conciseness	QN	[ZRM+16], [DAL16], [WK17]
R	Interoperability	QN/QL	[ZRM+16], [RMGCGP18], [FBMR18], [DAL16], [WK17]
R	Interpretability	QN/QL	[ZRM+16], [CCCD19], [DAL16], [WK17]
R	Versatility	QN	[ZRM+16], [DAL16], [WK17], [MLU22]
R	Robustness	QN	[CCCD19], [YHXL22]
	Noise	QN	[YHXL22]
	Hyper-relational structure	QN	[MLU22]

Table 5.6: Different dimensions of quantifiable quality characteristics for KGs and the literature in which they are mentioned. The first column corresponds to the Dimensions as defined by Zaveri et al. [ZRM+16]: A - Accessibility, I - Intrinsic, C - Contextual, R - Representational. QN - quantitative (quantifiable), QL qualititave (not quantifiable).

Not all of the dimensions of LD quality can be fully quantified, some of them depend of personal perceptions. Some methods to quantify them are the following ([ZRM+16], [RMGCGP18], [CCCD19], [FBMR18], [DAL16], [WK17], [HFKP20], [MT18], [GLX+19], [Had18], [FH11], [MMB12], [XZ22], [KWA+14], [MLU22], [YHXL22]):

- **Availability**: checking the response using SPARQL queries
- **Licensing**: detection of a licence
- **Interlinking**: usage of `owl:sameAs` links
- **Security**: detection of a digital signature
- **Performance**: number of request per second
- **Dereferenceability**: use of dereferenceable links
- **Accuracy**: no RDF pattern errors
- **Syntactic validity**: no malformed literals
- **Semantic validity**: no inaccurate annotations
- **Consistency**: valid use of `owl:DatatypeProperty`
- **Conciseness**: usage of unambiguous annotations
- **Compliance**: correct HTTP redirects
- **Relevancy**: coverage by number of entities in a dataset
- **Trustworthiness**: checking a list of trusted providers
- **Understandability**: human-readable labels
- **Timeliness**: difference between last modified time of the data source and data set
- **Diversity**: usage multi-typed data
- **Representational conciseness**:
- **Interoperability**: reuse of existing terms or vocabularies
- **Interpretability**: usage of blank nodes
- **Versatility**: using multilingualism
- **Robustness**: easily expandible
- **Noise**: usage of low-quality entities
- **Hyper-relational structure**: statements are specified by more information than a single relation (e.g. Wikidata)

5.3.2 Statistical Characteristics

Statistical characteristics represent basic statistics of the underlying data and they were only mentioned by three research groups: HG and Mishra [HM18], Paulheim and Bizer [PB14] and Färber et al. [FBMR18]. Table 5.7 provides an overview of the used metrics.

Characteristic	Reference
Number of instances	[HM18], [PB14], [FBMR18]
Number of type assertions	[HM18], [PB14]
Number of relations	[HM18], [PB14], [FBMR18]
Quantity of distinct classes	[HM18], [PB14], [FBMR18]
Quantity of distinct properties	[HM18], [PB14]
Mean depth of class hierarchy	[HM18], [PB14]
Fraction of untyped instances	[HM18], [PB14]
Average number of types per typed instance	[HM18], [PB14]
Average number of instances per class	[HM18], [PB14]
Average number of incoming/outgoing properties per instance	[HM18], [PB14]
Number of triples	[FBMR18]
Distribution of classes w.r.t. the number of their corresponding instances	[FBMR18]
Coverage of classes with at least one instance per class	[FBMR18]
Covered domains w.r.t. entities	[FBMR18]
Number of entities	[FBMR18]
Number of entities per class	[FBMR18]
Number of unique subjects	[FBMR18]
Number of unique predicates	[FBMR18]
Number of unique objects	[FBMR18]

Table 5.7: Statistical characteristics on KGs and the paper in which they are referenced

The characteristics that fall under this category can be quantified using SPARQL queries.

5.3.3 Statistical Distributions

These type of characteristics analyse distributions of a graph $G = (V, E)$, where V is the set of nodes (entities) and E is the set of edges (relations). The definitions and how they can be obtained from Table 5.8.

Distribution	Description	Reference
SDType	Uses the statistical distribution of a type in the subject and object position of the property for predicting the instance's types	[PB14]
SDValidate	Uses the statistical distributions to assess the correctness of statements which relate two resources	[PB14]
Degree Distribution	The degree distribution of G is $p(d) = \frac{n_d}{\|V\|}$, where n_d is the number of nodes whose degree is d and $\|V\|$ donates the number of nodes in G. In many graphs the degree follows a power-law distribution of the form $p(d) \propto L(d)d^{-\alpha}$, where $\alpha > 1$ and $L(d)$ is a slowly varying function.	[CWX+15]
Distribution of Hops	For a path $P = \{v_1, v_2, \ldots, v_h\}$ in G. The hop of a path P is defined as $Hops(P) = h - 2$, where h is the number of nodes in P. The distribution of hops reflects the connectivity cost inside a graph.	[CWX+15]
Distribution of Connected Components	There are strongly and weakly connected components in graph theory. A strongly connected component is a community in which any pair of nodes are reachable. A weakly connected component is a set of nodes in which any two nodes are reachable regardless of the edges' direction. The connected components reflect the connectivity of a graph.	[CWX+15]
Distribution of Clustering Coefficient	The definition of the clustering coefficient w.r.t. a nodes v_i is $C_i = \frac{\|\{e_{jk} : v_j, v_k \in N_i, e_{jk} \in E\}\|}{\|N_i\|(\|N_i\|-1)}$, where e_{jk} is the edge between v_j and v_k $(j \neq k)$ and N_i is the set of neighbour nodes of v_i. The clustering coefficient measures the nodes' tendency to cluster together.	[CWX+15]

Table 5.8: Statistical distributions and and the papers in which they are referenced.

5.3.4 Graph Theory

Cheng et al. [CWX+15] also mention statistics from graph theory as a way to characterize a KG. Although they can be translated to statistical characteristics, there was no other mention of graph theory in any of the other analyzed research papers. Table 5.9 summarizes these characteristics.

Characteristic	Description
Number of nodes	Number of entities
Number of edges	Number of relations
Density	Also explored by Yang et al. [YHXL22] as sparsity: $D(G) = \frac{\|E\|}{\|V\|(\|V\|-1)}$
Number of nodes with zero in-degree	Number of entities with no incoming relations
Number of nodes with zero out-degree	Number of entities with no outgoing relations
Number of bidirectional edges	Bidirectional relations
Number of closed triangles	A Trio of entities, where each is related to the other two
Number of open triangles	A Trio of entities, where each is related to at least one of the two
Average clustering coefficient	$\mathrm{C} = \frac{3\times\text{Number of closed triangles}}{\text{Number of open triangles}}$
Fraction of nodes in max weakly connected component	Each set of two entities is reachable regardless of the direction of the relations
Fraction of nodes in max strongly connected component	Each set of two entities is reachable following the direction of the relations
Approximately full diameter	Maximum distance between entities
90 percentile effective diameter	Minimum number of hops in which 90% of all connected pairs of entities are reachable

Table 5.9: Characteristics obtained from graph theory and their translations to KGs. Apart from the density, they were only referenced by Cheng et al. [CWX+15].

5.4 Conclusion

For this semi-systematic review three different search engines were used to gather research studies that have discussed KG characteristics. Figure 5.2 displays the process followed for this review, leading to a total of 19 analyzed papers. In Figure 5.1 the frequencies of the publication years across the selected papers are shown.

From the final selection of studies considered for this review, we identified four different types of characteristics: quality characteristics, statistical characteristics, statistical distributions, and characteristics derived from graph theory. We found that many of these can be quantified and used to distinguish one KG from another.

CHAPTER 6

Commonly used Knowledge Graphs in Recommender Systems

To determine which Knowledge Graph (KG) is often or seldom used for recommender systems in research, we conducted a semi-systematic review to address the second research question:

RQ2 *What KGs are commonly used to evaluate KG-based recommender algorithms? What are their characteristics?*

Similar to the first research question, the exploration of RQ2 follows the guidelines from Section 4.1.1. The following sections describe each phase of the review process in detail.

6.1 Design of the Review

We used the same search engines as for the first research questions: the Semantic Web – Interoperability, Usability, Applicability (referred to as Semantic Web journal)[1], ACM Digital Library[2] and Google Scholar[3].

We used the following three different search terms for the review:

- "Knowledge Graph recommender systems"
- "Linked data recommender"
- "Graph recommender systems"

[1]https://semantic-web-journal.net/, accessed 02.08.2024
[2]https://dl.acm.org/, accessed 02.08.2024
[3]https://scholar.google.com/, accessed 02.08.2024

The inclusion criteria for the types of research papers focused again of peer-reviewed articles, conference papers and surveys. An initial inclusion was made based on the title, followed by an assessment of the abstracts and finally the full texts. Papers found on the Semantic Web journal were again only regarded if they had already received the status "Accept".

Unlike the first research question, RQ2 covers a broader subject and therefore we were expecting to find more relevant papers (10-20) for the individual search terms from the ACM Digital Library and Google Scholar. Regarding the Semantic Web journal we still assumed to have a smaller outcome (1-5). Additionally we also expected to find more duplicates than in the previous review process. If a survey paper was selected, references within them were also regarded using the same criteria. If a reference of a survey paper was another survey, it was not added to the final list in order to prevent a large snowballing effect.

6.2 Execution of the Review

An initial selection was performed alone by their types and titles, a further refinement is documented in Section 6.2.1. Each search term was queried first in the Semantic Web journal, followed by the ACM Digital Library and finally Google Scholar. If any search lead to duplicates, they were disregarded on the initial search, such findings were added to the totals of the search engine were they were first discovered.

The number of findings for the first search term, "Knowledge graph characteristics", are summarized in Table 6.1, in total 50 papers were obtained.

	Article	Conference Papers	Survey	Total
Semantic Web Journal	3		1	4
ACM Digital Library	4	27		31
Google Scholar	6	4	4	14

Table 6.1: Breakdown of a total of 50 found papers across different databases for the search term "Knowledge Graph recommender systems"

For the second search term "Linked data recommender", a total of 36 was achieved across the search engines, which are summarized in Table 6.2.

Finally, with the last search term "Graph recommender systems" additional 6 papers were found, a summary can be seen in Table 6.3.

Scanning the libraries for the third search term yielded mostly duplicates from the previous searches, therefore the resulting total is a lot smaller than expected.

	Article	Conference Papers	Survey	Total
Semantic Web Journal				0
ACM Digital Library		22		22
Google Scholar	7	7		14

Table 6.2: Breakdown of a total of 36 found papers across different databases for the search term "Linked data recommender"

	Article	Conference Papers	Survey	Total
Semantic Web Journal				0
ACM Digital Library		2		2
Google Scholar	3			3

Table 6.3: Breakdown of a total of 5 found papers across different databases for the search term "Graph recommender systems"

6.2.1 Refining the Initial Selection and Additional Considerations

For the refinement the abstracts of the pre-selected papers were skimmed and based on the information it was decided whether to include or drop the paper. Additional sources obtained from survey papers were added to the list following the same selection criteria as the initial select. The refined selection of 127 papers can be found in Table 6.4.

	Article	Conference Papers	Survey	Total
Semantic Web Journal	1			1
ACM Digital Library	4	48		52
Google Scholar	11	7	3	21
Additional Considerations	14	39		53

Table 6.4: Breakdown of a total of 127 papers across different databases that were further regarded after reading the abstracts of the initial selections. 53 additional consideration was obtained from the survey papers.

6.2.2 The Final Selection of Considerations

Similar to the process of the first research question, the selection of 127 papers (see Table 6.4) were reviewed again using the following steps:

1. **Skimming the outline:** reading the abstract again, followed by the introduction and the conclusion.
2. **Examining the key sections:** focus on the methods, the discussion and the contributions.

3. **Evaluating the relevance:** asses the possible contribution to the research question.

If at any stage of these three steps a paper was considered irrelevant, it was excluded from the reviewing process.

Since this method lead to a lot of results, we decided to further make a constraint of the publication year by excluding papers that were published before 2019, so we still had a 5-year span (2019 - 2023) of research papers to explore.

The final selection of 67 publications can be found in Table 6.5, Figure 6.1 shows the process of this review and the selections and refinements of each step.

	Article	Conference Paper	Survey
SWJ	[RRDN+19]		
ACM	[WZX+19], [WZW+19], [LHL23]	[LAS23], [ZZB+20],[CLZ+22], [SCZ+20], [TCW+21], [WZZ+19], [DJM21], [CWT22], [WLZL21], [YHXL22], [WLF+21], [YHXH23], [HHC21], [LXJ+20], [HS19], [MMM21], [GNLC21], [XCZ+22], [WHW+21], [HLR+21], [FLH+23], [LMC22], [SA19], [ZDC+20], [ADNDS+21]	
GS	[EH22], [KLC23], [FZZG22], [WLZW22], [ZHY+19]	[KS21]	[GZQ+20], [CVD21],
Add.	[LTW+19], [LCPR19], [STC+20], [WWH+19], [ZWL20], [SSZ21]	[XHZ+19], [CWH+19], [TWYS19], [MZC+19], [YWY+19], [WL20], [CQA+19a], [CQA+19b], [WZZH20], [GSZT19], [SOK19], [WXZY20], [HTWX19], [BSY+20], [DLD19], [ZLSW19], [NPAB20], [WWX+19], [DTRP19], [ZZG+19], [XFM+19], [WHC+19], [HFQ+19], [QBZ+19]	

Table 6.5: Final selection of 67 papers from different databases for each search term after refining and adding additional considerations. SWJ - Semantic Web journal, ACM - ACM Digital Library, GS - Google Scholar, Add. - Additional Considerations.

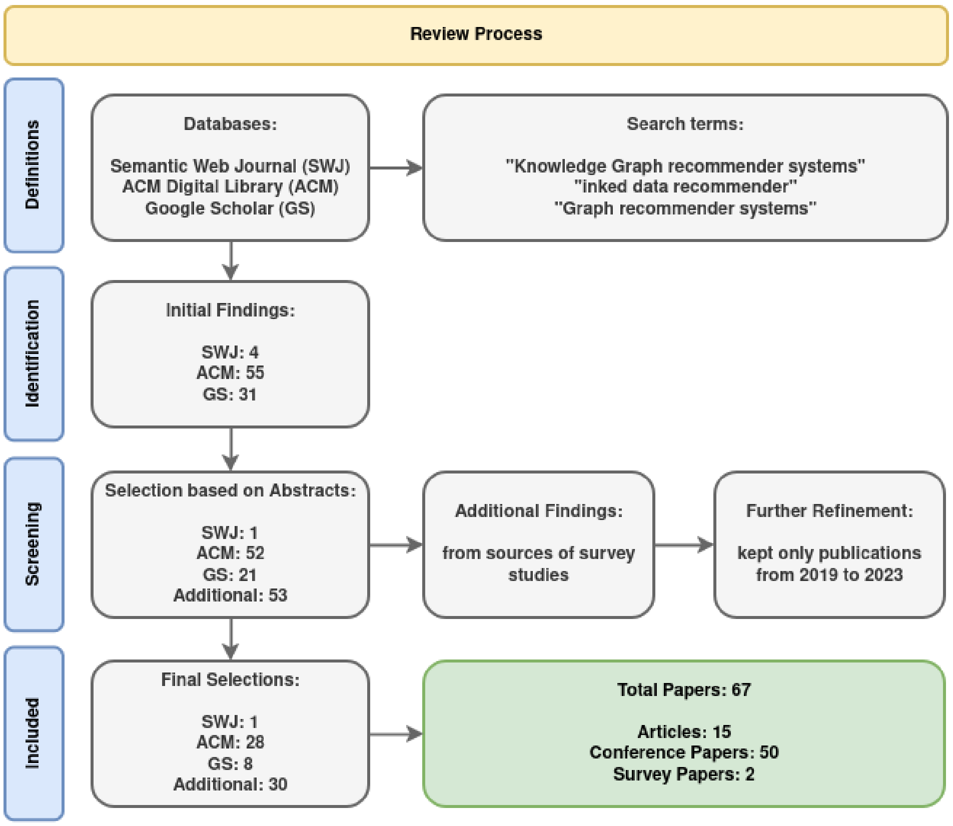


Figure 6.1: Summary of the review process for RQ2. Adapted from the flow diagram from the PRISMA guidelines for systematic reviews (https://www.prisma-statement.org/, accessed 05.08.2024)

6.3 Analysis of Most Used KGs

From the final selection as depicted in Table 6.5 the used KGs, domains and the reported characteristics were obtained and collected in Table 6.6.

Knowledge Graph	Unique Publications	Total Usage in all Publications
Microsoft Satori	11	30
DBpedia	6	11
Freebase	13	21
YAGO	2	4
Wikidata	3	4
HowNet	1	1
Custom-built	35	59

Table 6.6: Counts of occurrences of KGs in the selected papers. The column "Unique Publications" shows the number of unique publications in which the KGs were used, the column "Total Usage in all Publications" is the overall usage of the KG.

6.3.1 Microsoft Satori

Similar to Google Knowledge Graph (see Section 2.5, Microsoft Satori is integrated into the search engine from Microsoft's Bing[4] to improve the user experience for search queries. In its announcement in the year 2013[5], Bing's equivalent of Google's knowledge panels, "Snapshots" were introduced. Later, they were expanded to "Page Zero", a query completion method for a faster way to search entities[6].

In the analyzed publications Microsoft Satori was used 30 times in 11 unique publications and six different domains (see Table 6.7).

[4]https://www.bing.com/, accessed 02.08.2024

[5]https://blogs.bing.com/search/March-2013/Understand-Your-World-with-Bing, accessed 02.08.2024

[6]https://blogs.bing.com/search/September-2013/Page-Zero-A-Deep-Dive, accessed 02.08.2024

Domain	Dataset	References
Movies	MovieLens1M	[LTW+19], [QBZ+19], [TWYS19], [TCW+21], [WZZ+19]
Movies	MovieLens10M	[SCZ+20]
Movies	MovieLens20M	[WZW+19], [WZX+19], [WLZW22], [ZDC+20]
Movies	Amazon Movies	[LHL23]
Books	Book Crossing	[LTW+19], [LHL23], [QBZ+19], [TWYS19], [WZW+19], [WZX+19], [WZZ+19], [WLZW22], [ZDC+20]
Books	Amazon Books	[LHL23]
Music	Last.FM	[TCW+21], [WZX+19], [WZZ+19], [WLZW22]
Music	Amazon Music	[LHL23]
News	Bing News	[WZW+19], [WZZ+19]
Social	Dianping	[SCZ+20]
POI & Tourism	Yelp	[TCW+21]

Table 6.7: Overview of Microsoft Satori in various domains and their associated literature.

Details about Microsoft Satori are hard to acquire and therefore not easily accessible for private use [KKS21].

6.3.2 DBpedia

DBpedia[7] is a community-curated project that extracts structured data from Wikipedia[8] and publishes this knowledge in a multilingual Knowledge Base (KB) on the web [LIJ+15]. The first release was made in the year 2007 and it is published as LD in RDF format which can be queried using a SPARQL endpoint.

Domain	Dataset	References
Movies	MovieLens1M	[ADNDS+21], [CWH+19], [LAS23], [RRDN+19]
Movies	IMDB	[SA19]
Movies	Rotten Tomatoes	[SA19]
Books	DBBook	[CWH+19]
Books	Facebook Books	[ADNDS+21]
Books	LibraryThing	[RRDN+19]
Music	Last.FM	[RRDN+19]
Social	REDIAL	[ZZB+20]

Table 6.8: Overview of DBpedia in various domains and their associated literature.

[7]https://www.dbpedia.org/, accessed 02.08.2024
[8]https://wikipedia.org/; accessed 02.08.2024

DBpedia is published under the same licence as Wikipedia and therefore easily accessible for personal use[9]. A SPARQL-endpoint is alsow provided for querying the data[10].

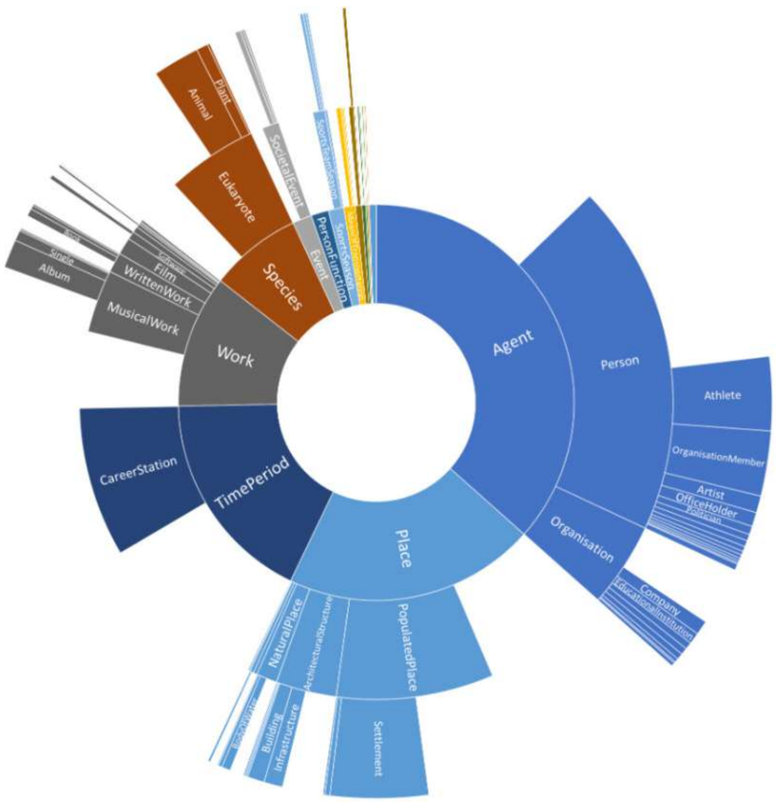


Figure 6.2: Instances in DBpedia (from [HHRP20]). It currently consists of about 4,233,000 instances, 768 classes and 3,000 properties (https://www.dbpedia.org/resources/ontology/, accessed 02.08.2024).

6.3.3 Freebase

Freebase was a KB mainly curated by user entries that existed from 2007 until it was shut down in 2015[11]. In 2014 the data from Freebase was moved to Wikidata and in 2015 it was also replaced by the Google Knowledge Graph. A total of 1.9 billion Freebase triples is still available as data dump for download[12].

[9]https://www.dbpedia.org/imprint/, accessed: 01.08.2024

[10]https://dbpedia.org/sparql, accessed 01.08.2024

[11]https://web.archive.org/web/20170729170416/https://plus.google.com/109936836907132434202/posts/bu3z2wVqcQc?
accessed 02.08.2024

[12]https://developers.google.com/freebase, accessed 02.08.2024

Domain	Dataset	References
Movies	MovieLens20M	[HFQ+19], [QBZ+19]
Movies	MovieLens1M	[WXZY20], [ZHY+19]
Books	Amazon Book	[FZZG22], [QBZ+19], [WHC+19], [WXZY20], [WHW+21], [ZHY+19]
Music	Last.FM	[FZZG22], [LXJ+20], [SSZ21], [WHC+19], [WXZY20], [YHXH23], [ZHY+19]
Products	Amazon Review	[MZC+19], [XFM+19], [ZZG+19]
POI & Tourism	Yelp2018	[WHC+19]

Table 6.9: Overview of Freebase in various domains and their associated literature.

6.3.4 Wikidata

Since its discontinuation in 2016, Wikidata[13] has been the successor of Freebase, though its initial launch was already in 2012. The entities in Wikidata have abstract identifiers, which makes them language-independent ([SAB+24]) but also challenging for human-readability.

Domain	Dataset	References
Music	Rotten Tomatoes	[SA19]
Music	IMDb	[SA19]
News	MIND	[XCZ+22], [YHXH23]

Table 6.10: Overview of Wikidata in various domains and their associated literature.

The Wikidata licence allows to use the KB freely for personal or commercial purposes[14], making it the largest open KB [SAB+24].

[13]https://www.wikidata.org/, accessed 02.08.2024
[14]https://www.wikidata.org/wiki/Wikidata:Introduction, accessed 02.08.2024

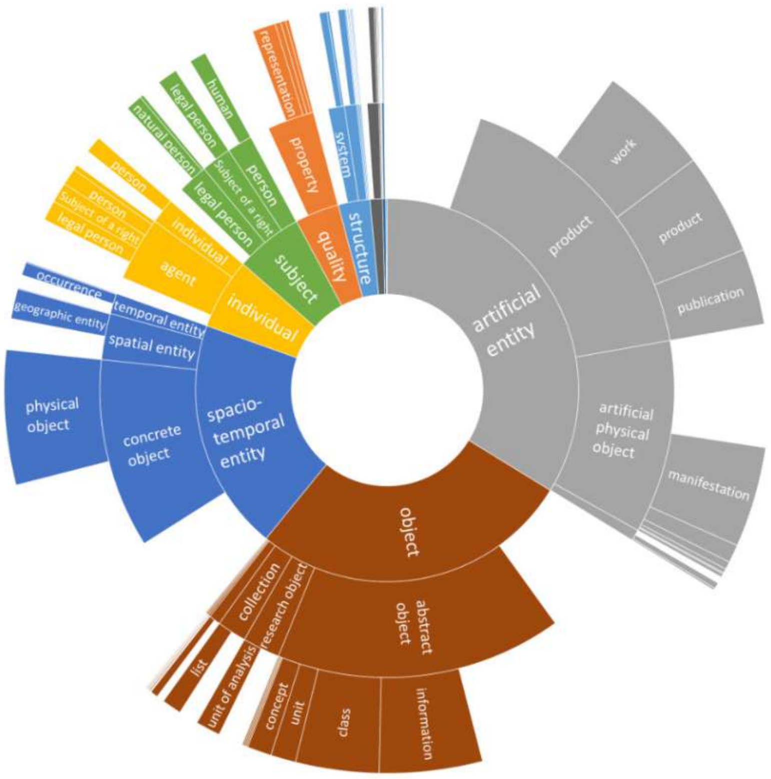


Figure 6.3: Instances in Wikidata (from [HHRP20]). It consists of around 1.4 billion facts and more than 100 million entities.

6.3.5 YAGO

YAGO (Yet Another Great Ontology)[15] was originally built by combining Wikipedia and the KB Wordnet, but since the release of YAGO 4, it consists of the facts of Wikidata and the ontology of schema.org[16] [PTWS20].

Domain	Dataset	References
Music	Last.FM	[WXZY20], [ZHY+19]
Books	Amazon Books	[WXZY20], [ZHY+19]

Table 6.11: Overview of YAGO in various domains and their associated literature

YAGO is also published under a licence allowing personal or commercial use.

[15]https://yago-knowledge.org/, accessed 02.08.2024
[16]https://schema.org/, accessed 02.08.2024

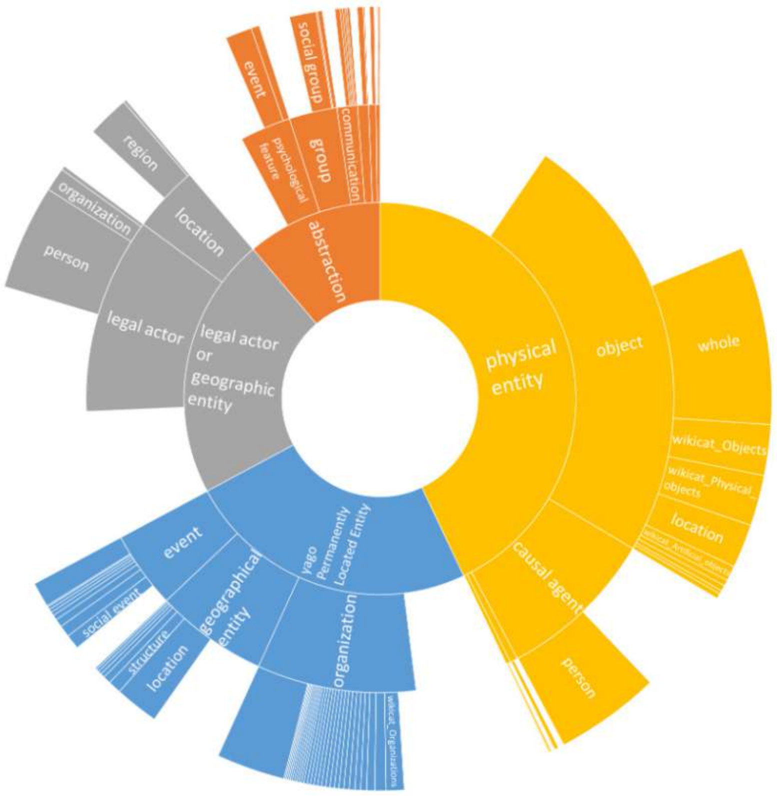


Figure 6.4: Instances in YAGO (from [HHRP20]). Currently YAGO 4.5 contains more than 50 million entities and 90 million facts (https://yago-knowledge.org/getting-started, accessed 02.08.2024).

6.3.6 Custom-Built and Other Knowledge Graphs

A variety of researchers built their own KGs without an external Knowledge Base (KB) or did not further specify the KB they used. The domains with the datasets and their references can be found in Table 6.12.

Domain	Dataset	References
Movies	MovieLens100k, IMDb	[EH22], [GNLC21], [KS21], [LMC22], [SSZ21], [WWX+19], [WLZL21], [XHZ+19]
Movies	MovieLens20M	[KLC23], [LXJ+20], [CLZ+22]
Movies	MovieLens1M	[WLF+21], [HS19], [DJM21]
Movies	Amazon Movies	[LMC22]
Books	Book Crossing	[EH22], [LXJ+20], [WLF+21], [DJM21]
Books	Anime	[GNLC21]
Books	Amazon Book	[WHW+21], [HHC21], [CWT22], [CLZ+22]
Music	Last.FM	[EH22], [WLF+21], [HHC21], [HS19], [CWT22], [DJM21]
Music	KKBox	[WWX+19], [XHZ+19]
Products	Alibaba-iFashion	[WHW+21], [YHXH23], [ZZG+19], [CWT22]
Products	Alibaba Taobao	[YWY+19]
POI & Tourism	Yelp	[FZZG22], [HTWX19], [STC+20], [SSZ21], [GSZT19], [WWH+19], [HHC21], [CLZ+22]
POI & Tourism	POI data from Hainan	[WZZH20]
POI & Tourism	CEM	[DTRP19]
POI & Tourism	Tripadvisor	[SOK19]
Health	Online Suicide Rescue Instruction	[WL20]
Health	National Health and Nutrition Examination Survey	[CQA+19b]
Health	Research studies	[CQA+19a]
Education	ScienceDirect	[LCPR19]
Software	Apple App Store	[DLD19]
Business	JGJISNF dataset	[BSY+20]
News*	Wikipedia, People's Daily newspaper	[ZLSW19]

Table 6.12: Overview of custom-built KGs in various domains and their associated literature. The KG marked with * uses HowNet as KB.

6.4 Analysis of the Reported Characteristics

All of the 67 research papers were also investigated for the type of characteristics they mention. Not every study reported characteristics of their KGs, the ones that did are summarized in Table 6.13.

6.5 Conclusion

To explore the second research question through a semi-systematic review, we used three different search engines, three search terms and a restriction of the publication year. Figure 6.1 illustrates the process, resulting in 67 research papers for analysis and summary.

As stated above, the publication years span from 2018 to 2023 due to the defined limitation, but without this constraint, the earliest publication found was from 2010 (see Figure 1.1.

The fact that even at the final stage of the selection process 67 research papers remained (127 in the step before that) shows, indicating a substantial interest in Recommender Systems (RS) with KGs. We found that five different external KBs were often used by research across seven different domains (with four additional domains if custom-built KGs are included). Most datasets were from the domains movies, books or music.

By analyzing the reported characteristics in these research studies, we can see that often similar ones are mentioned, with some of the publications not mentioning any at all. Comparing the information from Table 6.13 with the variety of characteristics identified in the first research question (see Chapter 5), we conclude that this research area still offers a range of opportunities for exploration.

Characteristic	References
#Items	[CWH$^+$19], [EH22], [FZZG22], [GSZT19], [HFQ$^+$19], [KLC23], [LAS23], [LTW$^+$19], [LXJ$^+$20], [LHL23], [MZC$^+$19], [QBZ$^+$19], [RRDN$^+$19], [SSZ21], [SCZ$^+$20], [TWYS19], [TCW$^+$21], [WWX$^+$19], [WHC$^+$19], [WZX$^+$19], [WZZ$^+$19], [WXZY20], [WLZL21], [WHW$^+$21], [WLZW22], [XFM$^+$19], [XHZ$^+$19], [XCZ$^+$22], [YHXH23], [ZZG$^+$19], [ZHY$^+$19], [ZDC$^+$20], [YWY$^+$19], [HHC21], [SOK19], [HS19], [CWT22], [DJM21], [CLZ$^+$22]
#Triples	[CWH$^+$19], [EH22], [FZZG22], [HFQ$^+$19], [KS21], [LAS23], [LTW$^+$19], [LXJ$^+$20], [LHL23], [QBZ$^+$19], [SCZ$^+$20], [TWYS19], [TCW$^+$21], [WWX$^+$19], [WZW$^+$19], [WHC$^+$19], [WZX$^+$19], [WZZ$^+$19], [WXZY20], [WLZL21], [WHW$^+$21], [WLZW22], [XHZ$^+$19], [YHXH23], [ZDC$^+$20], [HHC21], [SOK19], [CWT22], [DJM21], [CLZ$^+$22]
#Relations	[CWH$^+$19], [DLD19], [EH22], [FZZG22], [HFQ$^+$19], [KS21], [LAS23], [LTW$^+$19], [LXJ$^+$20], [LHL23], [MZC$^+$19], [QBZ$^+$19], [SSZ21], [SCZ$^+$20], [TWYS19], [TCW$^+$21], [WWX$^+$19], [WZW$^+$19], [WHC$^+$19], [WZX$^+$19], [WHW$^+$21], [WLZW22], [YHXH23], [ZZG$^+$19], [ZDC$^+$20], [HHC21], [CWT22], [DJM21], [CLZ$^+$22]
Sparsity	[CWH$^+$19], [DTRP19], [DLD19], [GSZT19], [KLC23], [QBZ$^+$19], [SSZ21], [WLZW22]
#Entities	[CWH$^+$19], [DTRP19], [EH22], [FZZG22], [HFQ$^+$19], [KS21], [LAS23], [LTW$^+$19], [LXJ$^+$20], [LHL23], [QBZ$^+$19], [SSZ21], [SCZ$^+$20], [TWYS19], [TCW$^+$21], [WWX$^+$19], [WZW$^+$19], [WHC$^+$19], [WZX$^+$19], [WHW$^+$21], [WLZW22], [YHXH23], [ZDC$^+$20], [YWY$^+$19], [HHC21], [SOK19], [CWT22], [DJM21], [CLZ$^+$22]
#k-hop triples	[EH22], [FZZG22], [QBZ$^+$19], [SCZ$^+$20], [WZW$^+$19], [ZDC$^+$20]
#Interactions	[DTRP19], [EH22], [FZZG22], [HFQ$^+$19], [LAS23], [LXJ$^+$20], [LHL23], [QBZ$^+$19], [SSZ21], [SCZ$^+$20], [TCW$^+$21], [WWX$^+$19], [WZW$^+$19], [WHC$^+$19], [WZX$^+$19], [WZZ$^+$19], [WXZY20], [WHW$^+$21], [WLZW22], [XHZ$^+$19], [YHXH23], [ZHY$^+$19], [ZDC$^+$20], [HHC21], [HS19], [CWT22], [DJM21], [CLZ$^+$22]
#Users	[CWH$^+$19], [EH22], [FZZG22], [GSZT19], [HFQ$^+$19], [KLC23], [LAS23], [LTW$^+$19], [LXJ$^+$20], [LHL23], [MZC$^+$19], [QBZ$^+$19], [SSZ21], [SCZ$^+$20], [TWYS19], [TCW$^+$21], [WWX$^+$19], [WZW$^+$19], [WHC$^+$19], [WZX$^+$19], [WZZ$^+$19], [WXZY20], [WLZL21], [WHW$^+$21], [WLZW22], [XFM$^+$19], [XHZ$^+$19], [XCZ$^+$22], [YHXH23], [ZZG$^+$19], [ZHY$^+$19], [ZDC$^+$20], [YWY$^+$19], [HHC21], [SOK19], [HS19], [CWT22], [DJM21], [CLZ$^+$22]
Average paths	[HFQ$^+$19], [LAS23], [WWX$^+$19], [XFM$^+$19]

Table 6.13: Summary of characteristics and their related papers in KG-based RS applications, that were reported by at least three different publications. #k-hop triples refers to the number of triples that are k hops away from a seed nodes [WZW$^+$18], the #Interactions are the user-item interactions (e.g. ratings)

CHAPTER 7

Experimental Evaluation

As mentioned in Chapter 4, the experimental design follows the guidelines by Wohlin et al. [WRH+12] consisting of the four phases experimental scoping, experiment planning, experiment operation and analysis and interpretation.

The following sections provide a detailed protocol of the experimentation process, aligning with the four phases mentioned before.

7.1 Experiment Scoping

The experiment aims to explore the third research question:

RQ2 *How do the RS algorithms perform on KGs with distinctive characteristics? What influence do certain KG characteristics have on the performance of a family of recommendation algorithms?*

7.1.1 Problem Statement

The utilization of a Knowledge Graph (KG) as additional information in Recommender Systems (RS) has gained interest over the span of the last years [CVD21] (see also Chapter 6). A KG can provide rich semantic side information to help overcome challenges such as data sparsity and cold-start problems as well as an increased explainability of recommendations [GZQ+20]. To our knowledge, the comparison of KG characteristics on the performances of RS in current research still leaves room for exploration (see also Chapter 3 for an overview of related research).

7.1.2 Objectives and Goals

The following objectives are defined for this experiment:

- **Evaluate algorithm performances:** assess the performances of a variety of RS algorithms on KGs with distinctive characteristics
- **Identify influential characteristics:** determine which characteristics influence the performances of the algorithms
- **Comparing algorithms:** analyze the performances of the algorithms on different KGs

The overall goal of this experiment stated in the template of Wohlin et al. [WRH$^+$12] is: *Analyze RS algorithms for the purpose of evaluating their performance with respect to their effectiveness on KGs from the point of view of the researcher in the context of KG characteristics.*

7.1.3 Methodology

The overall methodology is already outlined in Chapter 4, the key points are:

- Select a dataset from a popular domain in literature and KBs to serve as a side information
- Build the KGs
- Select a variety of algorithms and generate recommendations
- Evaluate the performances on selected metrics
- Compare the results using statistical tests to draw conclusions

7.2 Experiment Planning

The second phase consists of the experiment design to test the hypothesis regarding the influence of KG characteristics. This step contains the formulation of the hypothesis, the determination of the variables and tools and the selection of the dataset, KBs and characteristics to evaluate.

7.2.1 Hypothesis

The hypotheses for this experiment are the following:

- **Null Hypothesis (H_0):** The characteristics of KGs have no significant impact on the performance of RS algorithms.
- **Alternative Hypothesis (H_1):** The characteristics of KGs have a significant impact on the performance of RS algorithms.

7.2.2 Variable Selection

In the scope of this experiment, the independent variables (i.e. the controlled variables whose effect we want to study) are the characteristics of the KGs. We chose to evaluate different types of characteristics based on the finding of RQ1 in Chapter 5:

- Density (graph theory)
- k-hops (statistical distribution/characteristic)
- Number of blank nodes (quality characteristic)
- Number of entities (statistical characteristic)
- Number of triples (statistical characteristic)
- Number of instances (statistical characteristic)
- Number of classes (statistical characteristic)
- Number of predicates (statistical characteristic)

The dependent variables (i.e. the variables that potentially depend upon the independent variables) are the performance metrics of the RS.

The algorithms we used are those from Chapter 2.7. By combining semantic distance measures, information content metrics, and embeddings, we were aiming to achieve a nuanced understanding of the interplay of the KG characteristics in the performance of RS.

To evaluate the performances we used the standard metrics Precision@K, Recall@K and the F-measure@K [GS15]. Precision@K is the ratio of the number of relevant items to the total of retrieved items in the top K retrieved items, which indicates the accuracy of the positive predictions made by the recommender:

$$\text{Recall@K} = \frac{\text{Number of relevant items in K}}{K}. \tag{7.1}$$

Recall@K on the other hand, is the ratio of the number of relevant items to the total number of relevant items in the data amongst the top K relevant items, which indicates the ability to identify the the relevant items:

$$\text{Recall@K} = \frac{\text{Number of relevant items in K}}{\text{Total number of relevant items}}. \tag{7.2}$$

The F-measure is the harmonic mean between precision and recall:

$$F@K = 2 \times \frac{\text{Precision@K} \times \text{Recall}}{\text{Precision@K} + \text{Recall}}. \tag{7.3}$$

7.2.3 Dataset and Knowledge Bases

Since many researchers that we analyzed for RQ2 in Chapter 6 performed recommendation on one of the MovieLens datasets from the movie domain, we decided to use the small MovieLens dataset for education and development by GroupLens Research[1] [HK15].

The dataset contains 100,836 ratings given by 610 users on MovieLens[2] to 9,742 movies based on a 5-star system (1 being the worst and 5 being the best rating). For the experiment only the file containing the ID, name and genre of the movies are needed. To recommend movies to users, the file consisting of the ID of the user, the ID of the movie, the rating given by the user and the timestamp of the rating is necessary – only positive feedback (i.e. a rating greater than 3) was regarded to present users with movies they may also like.

The publications which were analyzed in Chapter 6 mostly used Microsoft Satori or Freebase as their KBs. As stated before, Microsoft Satori is not easily usable and Freebase is only available as data dump. To enable reproducability and also a timeliness of the experiment, we therefore decided on the use of DBpedia, Wikidata and YAGO as our external KBs.

7.2.4 Experiment Design

The experiment design is already explained in Chapter 4 and visualized in Figure 1.3. To enable consistent test conditions for each setting, the dataset preprocessing and parameter tuning stayed the same.

For each of the three KBs (DBpedia, Wikidata and YAGO) along with the dataset four KGs with different characteristics are built:

- **Version 1 (v1):** a base version of the KG.
- **Version 2 (v2):** if it exists, the release date of each movie is added via a blank node.
  This corresponds to a change of the quality characteristics of interpretability.
- **Version 3 (v3):** included `owl:sameAs` links.
  This corresponds to a change of the statistical characteristics of the number of triples, the quality characteristics of interlinking, and the graph characteristic of density.
- **Version 4 (v4):** included inverse relations.
  This corresponds to a change of the statistical distribution/statistic of the number of k-hop neighbors.

[1]https://grouplens.org/datasets/movielens/
[2]http://movielens.org

The underlying data in each KG stays the same across the versions, only the structures of the graphs are targets of the changes.

Then, all of the proposed algorithms (see Chapter 2.7) are implemented and used on each of the 12 KGs. Using multiple train/test splits of the movie data, recommendations are generated for each of them. The non-embedding algorithms produce distances or similarities, so for the embedding-based methods we used the cosine similarity to calculate the closeness of the entities to obtain the recommendations.

Definition 7.2.1 (Cosine similarity)**.** The cosine similarity between two vectors $\mathbf{A}$ and $\mathbf{B}$ measures the cosine of the angle between them using the dot product $\mathbf{A} \cdot \mathbf{B}$ and the euclidean norms (magnitude) $\|\mathbf{A}\|$ and $\|\mathbf{B}\|$:

$$cos(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{7.4}$$

After gathering the top-15 recommendations for each user, the results are aggregated for KG and algorithm and collected for statistical analysis.

7.3 Experiment Execution

The experiments were performed on a computer operating on the Linux distribution Ubuntu 22.04.4 LTS (Jammy Jellyfish)[3]. The hardware is equipped with an Intel Core i7-7700HQ processor, 16GB of RAM, and a 256GB SSD. The computer has a NVIDIA GeForce 940MX graphics card with NVIDIA CUDA version 12.2[4] installed.

The programming parts were written in Python using VS Code 1.90[5] and Python 3.9.13[6]. Storing, querying and managing the RDF data was performed with the SPARQL server Apache Jena Fuseki 4.6.1[7] and Apache Jena TDB[8].

The relevant python libraries for the experiment were the following:

- NumPy 1.21.5[9] [HMvdW+20] for vectorization and mathematical functions
- pandas 1.4.4[10] [pdt20], [WM10] for data analysis and manipulation
- scikit-learn 1.0.2[11] [PVG+11] for normalization and cosine similarity
- SciPy 1.9.1[12] [VGO+20] for statistical analysis

[3]https://releases.ubuntu.com/jammy/
[4]https://docs.nvidia.com/cuda/archive/12.2.1/cuda-toolkit-release-notes/index.html
[5]https://code.visualstudio.com/
[6]https://www.python.org/
[7]https://jena.apache.org/documentation/fuseki2/
[8]https://jena.apache.org/documentation/tdb/
[9]https://numpy.org/, accessed 05.08.
[10]https://pandas.pydata.org/, accessed 05.08.2024
[11]https://scikit-learn.org/stable/, accessed 05.08.2024
[12]https://scipy.org/, accessed 05.08.2024

- Matplotlib 3.5.2[13] [Hun07] for plotting of results
- seaborn 0.11.2[14] [Was21] for statistical data visualization
- RDFLib 6.2.0[15] for working with RDF data
- SPARQLWrapper 2.0.0[16] for running SPARQL queries on graphs in memory or via an endpoint
- PyKEEN 1.10.2.dev0[17] [ABH+21] for the embedding algorithm TransE and DistMult with GPU
- pyRDF2Vec 0.2.3[18] [SVAO23] for the embedding algorithm RDF2Vec

7.3.1 Knowledge Graph Creation

The KGs are created by accessing the selected KBs through their SPARQL endpoints and matching the movie entities to the movie titles of the dataset and the year of the release with regular expressions. This is performed using the DBpedia endpoint[19], the extracted entities are then used to build a KG.

For Wikidata and YAGO, the matched movie entities from DBpedia are used to extract the corresponding entities of these two KBs using `owl:sameAs` links (Figure 3.1 shows the linkage between these KBs). The endpoints of Wikidata[20] and YAGO[21] were then accessed to build the graphs.

After the entities have been extracted, a KG was created using `CONSTRUCT` queries which was the same for all four versions and served as their base. The different versions were then built by adding to the initial KG.

After the creation of the KGs, their characteristics were evaluated and the graphs loaded into a local Apache Jena TDB store.

DBpedia

Initial (used for all versions v1–v4):
Consists of all subjects and predicates of the movie entity, which is positioned as the object (subject → predicate → **movie**).
Excluded are the following predicates:

- `dbo:wikiPageRedirects`

[13]https://matplotlib.org/, accessed 05.08.2024
[14]https://seaborn.pydata.org/, accessed 05.08.2024
[15]https://rdflib.readthedocs.io/en/stable/, accessed 05.08.2024
[16]https://sparqlwrapper.readthedocs.io/en/latest/, accessed 05.08.2024
[17]https://pykeen.readthedocs.io/en/stable/, accessed 05.08.2024
[18]https://pyrdf2vec.readthedocs.io/en/latest/readme.html, accessed 05.08.2024
[19]https://dbpedia.org/sparql, accessed 05.08.2024
[20]https://query.wikidata.org/, accessed 05.08.2024
[21]https://yago-knowledge.org/sparql, accessed 05.08.2024

- dbo:wikiPageDisambiguates
- owl:sameAs

Version 1 (v1):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
Included are the following predicates:

- rdf:type
- dcterms:subject
- dbo:starring
- dbo:director
- dbo:distributor
- dbo:musicComposer
- dbo:genre
- dbo:releaseDate (optional, only if it exists)

The owl:sameAs predicate is again excluded.

Version 2 (v2):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and excluded predicates are the same as for v1, with the only difference that the dbo:releaseDate is not directly connected to the movie, but with a blank node between them.

Version 3 (v3):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and predicates are the same as for v1, with the only difference that the predicate owl:sameAs is not excluded.

Version 4 (v4):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and excluded predicates are the same as for v1, with the only difference that all of the included predicates with the prefix dbo are also inversed (e.g. movie → dbo:starring → object is also added as its inverse: subject → dbo:starring_inverse → movie).

The characteristics of the versions created from DBpedia are summarized in Table 7.1.

Characteristic	v1	v2	v3	v4
Density	5.5588×10^{-5}	4.8731×10^{-5}	8.9401×10^{-6}	6.3971×10^{-5}
1-Hop	57.9935	57.9270	87.3747	57.9935
2-Hop	0.0001	1.0564	0.0001	220.6974
3-Hop	0.0049	0.0049	0.0077	4625.6626
Number of Blank Nodes	0	6,758	0	0
Number of Entities	35,757	42,515	35,757	55,144
Number of Triples	426,858	433,547	625,416	491,226
Number of Instances	175,046	175,046	181,621	175,046
Number of Classes	5,212	5,212	5,216	5,212
Number of Predicates	34	35	35	40

Table 7.1: Characteristics of the KGs derived from DBpedia.

Wikidata

Initial (used for all versions v1–v4):
Consists of all subjects and predicates of the movie entity, which is positioned as the object (subject → predicate → **movie**).
Excluded are the following predicates:

- `wdt:P1365` (replaces)
- `wdt:P1889` (different from)
- `owl:sameAs`

Version 1 (v1):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
Included are the following predicates:

- `rdf:type`
- `wdt:P31` (instance of)
- `wdt:P161` (cast member)
- `wdt:P57` (director)
- `wdt:P750` (distributed by)
- `wdt:P162` (producer)
- `wdt:P86` (composer)
- `wdt:P58` (screenwriter)
- `wdt:P136` (genre)

- `wdt:P577` (release date, optional, only if it exists)

The `owl:sameAs` predicate is again excluded.

Version 2 (v2):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and excluded predicates are the same as for v1, with the only difference that the `wdt:P577` is not directly connected to the movie, but with a blank node between them.

Version 3 (v3):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and predicates are the same as for v1, with the only difference that the predicate `owl:sameAs` is not excluded.

Version 4 (v4):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and excluded predicates are the same as for v1, with the only difference that all of the included predicates with the prefix `wdt` are also inverse (e.g. movie → `wdt:P31` → object is also added as its inverse: subject → `wdt:P31_inverse` → movie).

The characteristics of the versions created from Wikidata are summarized in Table 7.2.

Characteristic	v1	v2	v3	v4
Density	4.6579×10^{-6}	4.5232×10^{-6}	5.2361×10^{-6}	6.0574×10^{-6}
1-Hop	19.0784	16.5359	28.7205	19.0825
2-Hop	3.9566	7.0745	9.2823	349.5981
3-Hop	32.2547	27.5112	185.8381	5223.8678
Number of Blank Nodes	0	6,744	0	0
Number of Entities	18,314	18,314	18,336	53,778
Number of Triples	361,817	368,566	429,331	470,530
Number of Instances	190	190	6,956	190
Number of Classes	33	33	66	33
Number of Predicates	285	286	286	293

Table 7.2: Characteristics of the KGs derived from Wikidata.

YAGO

Initial (used for all versions v1–v4):
Consists of all subjects and predicates of the movie entity, which is positioned as the object (subject → predicate → **movie**).
Excluded are the following predicates:

- `schema:sameAs`
- `owl:sameAs`

Version 1 (v1):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
Included are the following predicates:

- `rdf:type`
- `schema:actor`
- `schema:countryOfOrigin`
- `schema:director`
- `schema:productionCompany`
- `schema:musicBy`
- `schema:about`
- `schema:datePublished` (optional, only if it exists)

The `owl:sameAs` and `schema:sameAs` predicates are again excluded.

Version 2 (v2):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and excluded predicates are the same as for v1, with the only difference that the `schema:datePublished` is not directly connected to the movie, but with a blank node between them.

Version 3 (v3):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and predicates are the same as for v1, with the only difference that the predicates `owl:sameAs` and `schema:sameAs` are not excluded.

Version 4 (v4):
Consists of the initial KG and predicates and objects connected to the movie entities (**movie** → predicate → object).
The included and excluded predicates are the same as for v1, with the only difference that all of the included predicates with the prefix `schema` are also inversed (e.g. movie → `schema:actor` → object is also added as its inverse: subject → `schema:actor_inverse` → movie).

The characteristics of the versions created from YAGO are summarized in Table 7.3.

Characteristic	v1	v2	v3	v4
Density	8.0619×10^{-5}	6.9799×10^{-5}	7.3498×10^{-5}	1.5037×10^{-4}
1-Hop	17.3978	17.3978	19.3953	17.4015
2-Hop	0.7786	1.7786	0.7794	66.9769
3-Hop	15.0863	15.0872	15.8472	1188.4354
Number of Blank Nodes	0	2,421	0	0
Number of Entities	3,116	5,537	3,116	20,762
Number of Triples	43,503	45,924	48,339	81,139
Number of Instances	2,518	2,518	4,939	2,518
Number of Classes	47	47	48	47
Number of Predicates	10	11	11	15

Table 7.3: Characteristics of the KGs derived from YAGO.

7.3.2 Algorithms and Recommendation

The non-embedding based algorithms were implemented according to their definitions as described in Chapter 2.7. The embedding-based algorithms were built using the corresponding python libraries as laid out in Section 7.3.

The distance-based and information content-based algorithms already produce distances or similarities, so those were used to generate the recommendations. The embedding-based methods create low-dimensional vector representations of the entities, for those we used the cosine-similarities to compute the relatedness between items.

For RDF2Vec the embeddings were trained with random walks and maximum depth of 2 and maximum walks of 200 and 400 and a maximum depth of 4 with 200 maximum walks.

The embedding dimension for TransE and Distmult was set to 100 with learning rates of 0.001 and 0.005.

The movie data was split into multiple train/test splits, which were kept equal for each KB, with the test set containing 10 movies per user. For each of the users, recommendations are generated by building a user profile, by using a weighted average of the already seen movies based on their ratings and returning the 15 most similar movies.

7.4 Analysis and Interpretation

After the recommending process was finished, we collected the achieved performance metrics and analyzed the results. The next section contains some descriptive statistics for the KGs built with DBpedia, Wikidata and YAGO. After that, statistical tests were performed to test the formulated hypothesis.

7.4.1 Descriptive Statistics

Before performing hypothesis testing, we take a first look at the achieved results for DBpedia, Wikidata and YAGO separately.

DBpedia

The performance metrics from the KGs built with DBpedia were averaged and are summarized in Table 7.4. Figure 7.1 shows the boxplots of the achieved performance metrics. The data suggests that some algorithms are more affected by the characteristics than others:

- LDSD shows an improvement of v1 with the other versions, only v2 achieved a lower averaged performance. The boxplot highlights these findings and visualizes this increase in performance.
- Resim achieved the best results with v4, similar to LDSD.
- PICSS does not seem to be affected by the versions.
- RDF2Vec lead to the overall best results and shows only slight variations in the averages. The boxplot however shows more variation across the KGversions, which could suggest that the results are inconsistent.
- TransE looks similar to RDF2Vec, but shows the same results for v4 and v1, but the figure does not show the same variability.
- DistMult achieved the best results with v2.

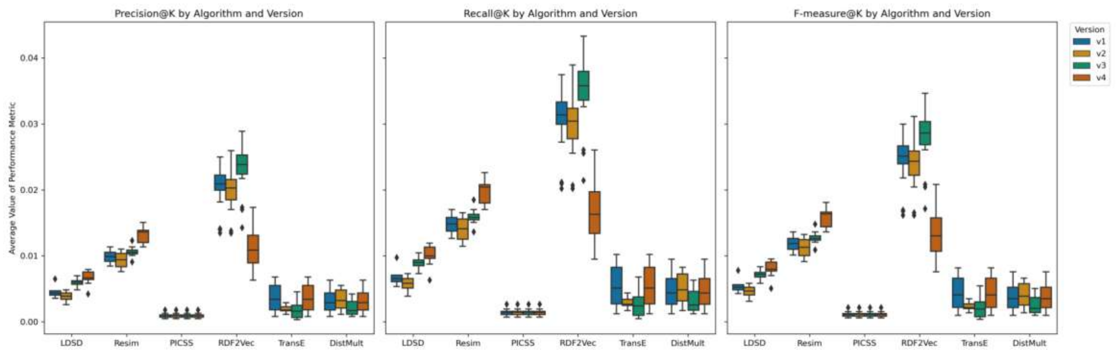


Figure 7.1: Boxplots of the performance metrics for each algorithm and version achieved with the KGs from DBpedia.

The distance-based algorithms achieved the best averaged performances with the KG v4, while the embedding-based algorithms only show slight variations across the versions. PICSS resulted in the lowest performances, which stay consistent for all versions.

		v1	v2	v3	v4
Average Precision@K	LDSD	0.004483	0.003834	0.005972	**0.006739**
	Resim	0.009865	0.009334	0.010632	**0.013109**
	PICSS	**0.000958**	**0.000958**	**0.000958**	**0.000958**
	RDF2Vec	0.020679	0.019942	**0.023879**	0.010956
	TransE	**0.003576**	0.001961	0.001644	**0.003576**
	DistMult	0.003178	**0.003333**	0.002146	0.003178
Average Recall@K	LDSD	0.006724	0.005751	0.008958	**0.010108**
	Resim	0.014798	0.014001	0.015948	**0.019664**
	PICSS	**0.001438**	**0.001438**	**0.001438**	**0.001438**
	RDF2Vec	0.031018	0.029912	**0.035818**	0.016434
	TransE	**0.005364**	0.002942	0.002466	**0.005364**
	DistMult	0.004767	**0.004999**	0.003218	0.004767
Average F-measure@K	LDSD	0.005379	0.004601	0.007167	**0.008087**
	Resim	0.011838	0.011201	0.012758	**0.015731**
	PICSS	**0.001150**	**0.001150**	**0.001150**	**0.001150**
	RDF2Vec	0.024815	0.023930	**0.028654**	0.013148
	TransE	**0.004291**	0.002353	0.001973	**0.004291**
	DistMult	0.003813	**0.003999**	0.002575	0.003813

Table 7.4: Average performance measured by Precision10, RecallK and the F-measure by algorithm and version achieved with the KGs from DBpedia. The highest average values are highlighted in bold.

Wikidata

The averaged performance metrics are displayed in Table 7.5 and the boxplots are shown in Figure 7.2. To summarize the observations:

- LDSD achieved the highest averages with v2, followed by v1. For v4 we can see an increase in the F-measure, indicating a balanced improvement of Precision and Recall.
- Resim achieved the best results with the base KG v1, but v2 only reduced the performance slightly.
- PICSS shows not consistent averages this time, but only slight variations.
- RDF2Vec visualized in the boxplot does not indicate the same level of variation as for the DBpedia KGs (see Figure 7.1). The table however shows a large increase for v4.
- TransE achieved notably the best results with v3.

- DistMult only indicates a dip with v3, but stays mostly consistent with the other versions.

		v1	v2	v3	v4
Average Precision@K	LDSD	0.037480	**0.038091**	0.013178	0.029008
	Resim	**0.041024**	0.040801	0.017988	0.030675
	PICSS	0.019909	0.019969	0.019969	**0.020043**
	RDF2Vec	0.003430	0.003643	0.004621	**0.006259**
	TransE	0.001377	0.001973	**0.002710**	0.001377
	DistMult	**0.002539**	0.002524	0.000908	**0.002539**
Average Recall@K	LDSD	0.056221	**0.057136**	0.019768	0.043511
	Resim	**0.061537**	0.061202	0.026982	0.046013
	PICSS	0.029864	0.029953	0.029953	**0.030065**
	RDF2Vec	0.005145	0.005465	0.006932	**0.009389**
	TransE	0.002066	0.002960	**0.004065**	0.002066
	DistMult	**0.003808**	0.003786	0.001363	**0.003808**
Average F-measure@K	LDSD	0.044977	**0.045709**	0.015814	0.034809
	Resim	**0.049229**	0.048961	0.021586	0.036810
	PICSS	0.023891	0.023962	0.023962	**0.024052**
	RDF2Vec	0.004116	0.004372	0.005545	**0.007511**
	TransE	0.001653	0.002368	**0.003252**	0.001653
	DistMult	**0.003047**	0.003029	0.001090	**0.003047**

Table 7.5: Average performance measured by Precision10, RecallK and the F-measure by algorithm and version achieved with the KGs from Wikidata. The highest average values are highlighted in bold.

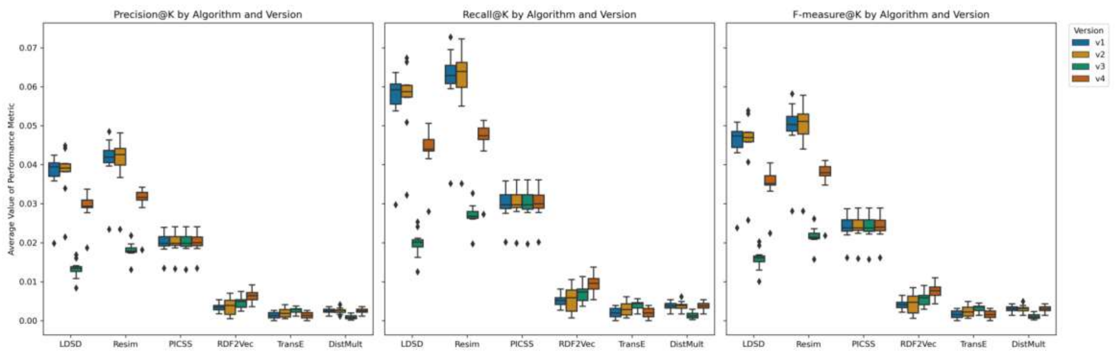


Figure 7.2: Boxplots of the performance metrics for each algorithm and version achieved with the KGs from Wikidata.

The distance-based methods showed again quite similar results with the highest achieved performance on v1 and v2. PICSS shows some minor variation in the performance metrics.

The embedding-based algorithms wary across the versions.

YAGO

The summarized and averaged performance metrics for the KGs built with YAGO are displayed in Table 7.6. Figure 7.3 shows the boxplots for each version and algorithm across the performance metrics. From those we can notice the following:

- LDSD performed quite consistent across the versions.
- Resim shows a peak in the plot for v3, which is also noticeable in the table, though smaller.
- PICSS stayed again consistent across the versions.
- RDF2Vec achieved a major increase in the performances for v4. This peak aligns with what the boxplot shows.
- TransE shows a peak in performance for v2 in the table as well as in the figure.
- DistMult spreads relatively tight across the versions, with a slight increase for v3.

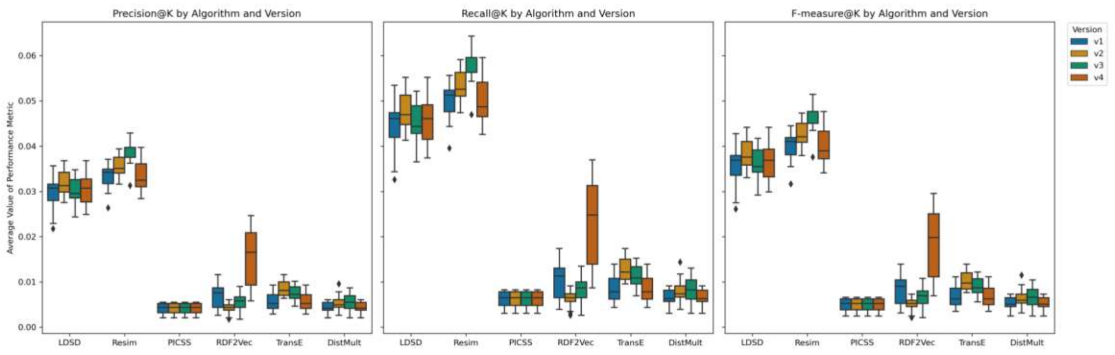


Figure 7.3: Boxplots of the performance metrics for each algorithm and version achieved with the KGs from YAGO.

The distance-based methods are seeming to be quite similar again, with the peak performances in v2 and v3. The performance of PICSS is not affected by the version changes. The embedding-based algorithms differ across the versions, though RDF2Vec indicates a large peak in performance for v4, while TransE aligns more with the characteristics fo v2.

7.4.2 Hypothesis Testing

After applying descriptive statistics, we applied statistical test to decide whether to reject or fail to reject the null-hypothesis with a significance level of 5%:

		v1	v2	v3	v4
Average Precision@K	LDSD	0.029407	**0.032016**	0.029802	0.030435
	Resim	0.033043	0.035599	**0.038472**	0.033544
	PICSS	**0.004111**	**0.004111**	**0.004111**	**0.004111**
	RDF2Vec	0.006983	0.004278	0.005639	**0.015793**
	TransE	0.005876	**0.008458**	0.007431	0.005876
	DistMult	0.004335	0.005204	**0.005560**	0.004335
Average Recall@K	LDSD	0.044111	**0.048024**	0.044704	0.045652
	Resim	0.049565	0.053399	**0.057708**	0.050316
	PICSS	**0.006166**	**0.006166**	**0.006166**	**0.006166**
	RDF2Vec	0.010474	0.006416	0.008458	**0.023689**
	TransE	0.008814	**0.012688**	0.011146	**0.012688**
	DistMult	0.006502	0.007806	**0.008340**	0.006502
Average F-measure@K	LDSD	0.035289	**0.038419**	0.035763	0.036522
	Resim	0.039652	0.042719	**0.046166**	0.040253
	PICSS	**0.004933**	**0.004933**	**0.004933**	**0.004933**
	RDF2Vec	0.008379	0.005133	0.006767	**0.018951**
	TransE	0.007051	**0.010150**	0.008917	0.007051
	DistMult	0.005202	0.006245	**0.006672**	0.005202

Table 7.6: Average performance measured by Precision@K, Recall@K and the F-measure@K by algorithm and version achieved with the KGs from YAGO. The highest average values are highlighted in bold.

- **Null Hypothesis (H_0):** The characteristics of KGs have no significant impact on the performance of RS algorithms.

- **Alternative Hypothesis (H_1):** The characteristics of KGs have a significant impact on the performance of RS algorithms.

We conducted this experiment using the three KBs Dbpedia, Wikidata and YAGO. For each of those, we built a base KG (v1) and three other KGs with varying characteristics (v2, v3, v4) but the same underlying data. After obtaining the results from the recommendation algorithms, we applied statistical tests to see if there are statistically significant differences in the performances from the base KGs to the modifications.

First we tested for normality in the obtained results. Since we had to reject the normality assumption for some of the data, we used the non-parametric Mann-Whitney U Test, which does not require the same assumptions as the t-test [WRH+12].

DBpedia

The results from the statistical tests of the base KG v1 against the other versions are displayed in Table 7.7. The values highlighted in bold indicate a statistically significant difference in the performances.

Recommender	Metric	v2	v3	v4
LDSD	PrecisionK	0.0799	**0.0008**	**0.0009**
	RecallK	0.0799	**0.0008**	**0.0009**
	Fscore	0.0799	**0.0008**	**0.0009**
Resim	PrecisionK	0.2921	0.0749	**0.0001**
	RecallK	0.2921	0.0749	**9.17e-05**
	Fscore	0.2921	0.0749	**9.17e-05**
PICSS	PrecisionK	1.0	1.0	1.0
	RecallK	1.0	1.0	1.0
	Fscore	1.0	1.0	1.0
RDF2Vec	PrecisionK	0.0914	**1.93e-06**	**1.44e-11**
	RecallK	0.0914	**1.87e-06**	**1.44e-11**
	Fscore	0.0914	**1.87e-06**	**1.44e-11**
TransE	PrecisionK	**0.0121**	**0.0011**	1.0
	RecallK	**0.0121**	**0.0011**	1.0
	Fscore	**0.0121**	**0.0011**	1.0
DistMult	PrecisionK	0.7511	**0.0315**	1.0
	RecallK	0.7511	**0.0315**	1.0
	Fscore	0.7511	**0.0315**	1.0

Table 7.7: p-values obtained from statistical testing with the Mann-Whitney U Test, where the base KG v1 was compared to the other versions (v2, v3 and v4), all built with DBpedia. Statistically significant values are highlighted bold.

Significant differences were observed for LDSD for v3 and v4, Resim only with version v4. For PICSS there were no significant differences observed. For RDF2Vec the p-values are very small in v3 and v4, TransE indicates a statistical difference for v2 and v3 and with DistMult we can only observe a statistical difference with v3.

The results suggest that certain characteristics significantly impact the performances of the RS. Version v2, the addition of blank nodes, only affects TransE significantly, but all o the embedding-based algorithms are influenced by the addition of the `owl:sameAs` predicates. RDF2Vec seems to be particularly influenced by additional predicates (`owl:sameAs` and the inverse links). The addition of the inverse links does not influence TransE and DistMult.

Wikidata

The test results from Wikidata are displayed in Table 7.8. Again, the statistically significant differences to the base KG are highlighted with bold text.

Recommender	Metric	v2	v3	v4
LDSD	PrecisionK	0.9214	**8.04e-05**	**0.0010**
	RecallK	0.9214	**8.04e-05**	**0.0010**
	Fscore	0.9214	**8.04e-05**	**0.0010**
Resim	PrecisionK	0.9476	**8.04e-05**	**0.0010**
	RecallK	0.9476	**8.04e-05**	**0.0010**
	Fscore	0.9476	**8.04e-05**	**0.0010**
PICSS	PrecisionK	0.9212	0.8436	0.8693
	RecallK	0.9212	0.8436	0.8693
	Fscore	0.9212	0.8436	0.8693
RDF2Vec	PrecisionK	0.6032	**0.0004**	**1.52e-10**
	RecallK	0.6032	**0.0004**	**1.52e-10**
	Fscore	0.6032	**0.0004**	**1.52e-10**
TransE	PrecisionK	**0.0328**	**5.81e-06**	1.0
	RecallK	**0.0328**	**5.81e-06**	1.0
	Fscore	**0.0328**	**5.81e-06**	1.0
DistMult	PrecisionK	0.7414	**9.93e-08**	1.0
	RecallK	0.7414	**9.93e-08**	1.0
	Fscore	0.7414	**9.93e-08**	1.0

Table 7.8: p-values obtained from statistical testing with the Mann-Whitney U Test, where the base KG v1 was compared to the other versions (v2, v3 and v4), all built with Wikidata. Statistically significant values are highlighted bold.

From the results of the Wikidata KGs we can observe a similar picture than from DBpedia. We can obtain significant differences for LDSD and Resim with the versions v3 and v4. In the case of PICSS there were no significant differences observed. RDF2Vec, TransE and DistMult show differences for the same versions as with DBpedia.

Overall the p-values for v3 are smaller than before and except for PICSS, all algorithms are affected by version v3 (added `owl:sameAs` links). The addition of the inverse links does not influence TransE and DistMult.

YAGO

The results of the statistical tests for YAGO can be obtained from Table 7.9, where the statistically signicant values are displayed in bold.

Algorithm	Metric	v2	v3	v4
LDSD	PrecisionK	0.2116	0.9738	0.7670
	RecallK	0.2116	0.9738	0.7670
	Fscore	0.2116	0.9738	0.7670
Resim	PrecisionK	0.1073	**0.0014**	0.9215
	RecallK	0.1073	**0.0014**	0.9215
	Fscore	0.1073	**0.0014**	0.9215
PICSS	PrecisionK	1.0	1.0	1.0
	RecallK	1.0	1.0	1.0
	Fscore	1.0	1.0	1.0
RDF2Vec	PrecisionK	**7.84e-05**	**0.0219**	**2.48e-07**
	RecallK	**7.84e-05**	**0.0235**	**2.31e-07**
	Fscore	**7.84e-05**	**0.0219**	**2.48e-07**
TransE	PrecisionK	**1.69e-04**	**0.0084**	1.0
	RecallK	**1.77e-04**	**0.0084**	1.0
	Fscore	**1.69e-04**	**0.0084**	1.0
DistMult	PrecisionK	0.0774	**0.0224**	1.0
	RecallK	0.0774	**0.0224**	1.0
	Fscore	0.0774	**0.0224**	1.0

Table 7.9: p-values obtained from statistical testing with the Mann-Whitney U Test, where the base KG v1 was compared to the other versions (v2, v3 and v4), all built with YAGO. Statistically significant values are highlighted bold.

With the YAGO KGs we can no longer observe differences across the versions for LDSD. Resim only shows significant differences for version v3 and PICSS is again not affected by the characteristics. RDF2Vec has small p-values for all versions, indicating a statistical difference in all of them. TransE and DistMult indicate the same differences as with DBpedia and Wikidata.

From these tests we can obtain overall similar statistical difference as we had in the previous two sections, although in this case RDF2Vec is affected by all of the variations of characteristics.

7.4.3 Interpretation

In summary, we were able to observe differences in some algorithms across the various KG versions and verified some of these findings with statistical tests.

The recommendations obtained with LDSD achieved statistically relevant changes for the versions v3 (adding of `owl:sameAs` predicates) and v4 (adding of inverse predicates) on the DBpedia and Wikidata graphs. Both versions resulted in an increase in the performance metrics on DBpedia, while there was a performance decrease on these versions on Wikidata. No statistic significance was diagnosed on the graphs from YAGO. The reason could be, that the changes in the structure of the base KG in terms of additional predicates has not achieved the same clear changes in the evaluated characteristics than with DBpedia and Wikidata. The YAGO-based KGs were also smaller in terms of the number of triples, which could also be a reason for this observation.

Resim was not significantly influences by the changes of v2 (adding of blank nodes) on either graph. On the DBpedia graphs an increase in performance of v4 was observed in the data and validated using statistical testing. On the Wikidata graphs the differences decreased in the versions v3 and v4. The YAGO graphs showed a performance increase on version v3.

The algorithm PICSS was not affected by the changes of the characteristics, which we were able to observe in the descriptive analysis of the retrieved performances as well as through statistical testing.

In the results of the embedding-based algorithm RDF2Vec the most considerable sensitivity was detected. A statistically significant change in the performances was observed over all graphs for the versions v3 and v4. On the YAGO graphs, a significant difference was additionally observed for v2. In the analysis of the average performance values and the boxplots these changes were also visible. The addition of inverse graphs influenced the performances on Wikidata and YAGO for the better, while the performance decreased on DBpedia. Version v3 always performed slightly better than the base graphs on all metrics, whereas v2 only had an observable impact on YAGO, although for the worse.

TransE was similarly sensible to the changes presented by v3 and additionally to v2. On all of the graphs a statistical difference on these two version could be observed. Only with DBpedia the changes resulted in a decrease of the performance for both versions. There was no difference in the performance of v4 as compared to v1.

DistMult was only affected by the changes introduced by version v3 over all graphs. As with TransE, there was no effect on the performance with v4 observable.

Overall, the introduction of additional predicates in the graphs (v3 and v4) generally has an impact on the performance of the algorithms. One should keep in mind, that they do not improve the performance in all settings and also lead to an increase in size and therefore the computational complexity. The variance of the different outcomes supports the idea that KG characteristics do indeed have an impact on the performances.

Conclusion on the Null Hypothesis

The findings of this experiment and the statistical tests applied to the results suggest the rejections of the null hypothesis H_0 at the 0.05 level. Each of the versions of the graphs with distinctive characteristics showed statistically significant effects on the performance of the RS for specific settings, although this impact is not constant across all algorithms.

CHAPTER 8

Conclusion and Future Work

In this thesis, we explored the impact of Knowledge Graph (KG) characteristics on the performance of Recommender Systems (RS). The goal was to identify the types KG characteristics and to obtain a deeper understanding of the interplay between them and the performances.

The results are summarized in the answers to the following research questions:

RQ1 *What are the defining characteristics of a KG that distinguishes it from another KG? How can these characteristics be quantified?*

With the guidelines provided by a semi-systematic review we analyzed 19 research papers and were able to establish four different types of characteristics: quality characteristics, statistical characteristics, statistical distributions and characteristics derived from graph theory. Apart from sub-categories of the quality characteristics all of them can be quantified using SPARQL-queries, mathematical operations, HTTP requests and other methods. The details are summarized in Chapter 5.

RQ2 *What KGs are commonly used to evaluate KG-based recommender algorithms? What are their characteristics?*

For the second research question we conducted another semi-systematic review, this time leading to a total of 127 publications. Due to the vast amount of research we decided to decrease the considered publication dates to a span of five years, resulting in 67 considered papers.

We identified 6 different KBs that were used in the literature, including Microsoft Satori, DBpedia, Freebase, YAGO, Wikidata and HowNet. Many of the researchers did not specify the used Knowledge Base (KB) or used custom-built KGs in their studies. From the used graphs, the only openly accessible and still maintained are DBpedia, YAGO and Wikidata, which are also cited in several papers. The datasets from these recommending experiment mostly are from the domains movies, books and music. The reported characteristics by the authors are mostly the same if there are any mentioned at all.

RQ3 *How do the RS algorithms perform on KGs with distinctive characteristics? What influence do certain KG characteristics have on the performance of a family of recommendation algorithms?*

The third research question was explored in the practical part of this thesis. We built KGs together with a dataset from the movie domain with distinctive characteristics, which were chosen based on the answers of the first two research questions. To generate recommendations we used different algorithms from the families of distance-based, information content-based and and embedding-based. The obtained results were analyzed using both descriptive statistics and testing of the following hypotheses:

- **Null Hypothesis (H_0):** The characteristics of KGs have no significant impact on the performance of RS algorithms.
- **Alternative Hypothesis (H_1):** The characteristics of KGs have a significant impact on the performance of RS algorithms.

The experimental results confirmed that characteristics can have an influence of the performance of RS, which led to the rejection of the null hypothesis. Specifically the addition of supplementary predicates significantly influenced the performances, only the information content-based method was not affected by any of the characteristics and demonstrated robustness to variations of the KGs. The embedding-based algorithm RDF2Vec appeared to have the highest sensitivity to these changes, which resulted in differences across almost all versions.

This research supports the hypothesis that RS are affected by the characteristics of a KG, aligning with the alternative hypothesis H_1. The observed nuances in the performances also provided deeper insights in the interplay of KG characteristics and RS performances and also into specific characteristics that are most influential.

8.1 Contributions of this Thesis

This thesis introduced a structured experimental framework to explore the effect that Knowledge Graph (KG) characteristics have on the performances of Recommender Systems (RS). Through performing semi-systematic reviews and a practical implementation, the thesis contributes to the understanding of KG characteristics and their influence on the performance of RS in the following points:

- **Characterization of KGs:** Through conducting a semi-systematic review we were able to collect and summarize four different types of KG characteristics: quality characteristics, statistical characteristics, statistical distributions and characteristics derived from graph theory. We also added examples on how to quantify them and cited the relevant literature for further information.

- **Review of used KGs:** With another semi-systematic research we identified the KGs that are often used by researchers. We also discovered, that the characteristics of the KGs are often not mentioned and aim to highlight the importance of this factor for future researchers.
- **Empirical evaluation:** By using the results from the first two research questions as inputs in our practical part, we made sure to use tools that are already in use by researchers. We built different subgraphs from consistent versions from the larger KGs DBpedia, Wikidata and YAGO to ensure results that are comparable amongst the different versions of the subgraphs. The statistical analysis of the final results highlights the importance of KG selection and configuration.
- **Guidance of best practices:** a first guidance for future researchers is to include the characteristics of their KGs, beyond basics such as the number of triples, to enable transparency and create a deeper understanding of their influence.
- **Discussion for future research:** This work can serve as a base for future discussion of the impact of characteristics on KG-based RS.

8.2 Limitations of the Thesis

This study was performed using only a limited amount of KG characteristics and RS algorithms. Future studies could extend this research to to allow for a deeper methods of application and generalization of the the proposed findings.

The methods to generate the recommendations using the similarities alone build a basis but could be enhanced to more sophisticated methods.

Additionally, the characteristics of the KGs were modified for the experiment. More advanced KGs might contain a higher complexity in their structures, which might not be entirely captured by the experimental setup.

8.3 Future Directives

Future directives include the conduction of further experiments to see if the captured trends can also be observed across other datasets or domains.

As already mentioned in the limitations, additional or more complex characteristics would also be a useful following direction. A deep dive into the characteristics that were identified to impact the algorithm most significantly (addition of further predicates) would also be a possible next step.

The algorithms could also be further explored by adding modifications or hyper parameter tuning which might enhance or decrease the sensitivity towards the characteristics.

Future research could also enrich emerging KG construction techniques towards a process that is more sensible towards the underlying characteristics.

There is also the potential to develop new algorithms or enhance existing ones to better utilize and capture the structures of real-world KGs.

8.4 Final Thoughts

As research on KGs and specifically KG-based RS continues to evolve and become even more sophisticated, a deeper focus on the interplay between the characteristics and performances will probably become more critical. The findings of this thesis aim to encourage further exploration in this topic and aspire to underpin the underlying potential to build a foundation for discussion of additional enhancement for personalized recommendations.

Übersicht verwendeter Hilfsmittel

Als Hilfsmittel für diese Arbeit wurde ChatGPT von OpenAI in folgenden Bereichen verwendet:

- Als Unterstützung beim Debuggen von eigenem Code und zur Hilfestellung bei Syntaxfragen.
- Umwandlung von selbst erstellten Tabellen in LaTeX-Format.
- Bei Übersetzungen ins deutsche/englische von kurzen Phrasen oder einzelner Vokabeln und zur Findung von Synonymen (im Sinne eines Wörterbuchs).

Es wurde keine von der KI erstellten Inhalte direkt übernommen.

List of Figures

List of Tables

Acronyms

CBF Content-based Filtering. 7, 8

CBOW Continuous-Bag-of-Words. 24

CF Collaborative Filtering. 7, 8, 91

IC Information Content. 20–22

IRI Internationalized Resource Identifier. 9

KB Knowledge Base. 7, 10, 11, 29, 37, 57–63, 66, 68, 70, 75, 80, 85, 94

KG Knowledge Graph. xiii, xiv, 1–7, 10–12, 14, 15, 20, 23, 24, 27–33, 36, 37, 39, 44, 45, 47–49, 51, 55, 56, 61–88, 91–94

KGE Knowledge Graph Embedding. 1, 2, 5, 7, 15, 23, 29–32, 91

LD Linked Data. 9, 21, 22, 27, 43, 44, 46, 57, 92

LDSD Linked Data Semantic Distance. xi, xiii, 16–18, 20, 32, 76, 77, 79, 81–83, 92

LOD Linked Open Data. 9, 10, 15, 16, 20, 21, 23, 29–31, 91

nDCG normalized Discounted Cumulative Gain. 31

OWL Web Ontology Language. 14

PCA Principal Component Analysis. 31

PIC Partitioned Information Content. 22

PICSS PIC-based Semantic Similarity. xi, xiii, 20, 22, 23, 33, 76–79, 81–84

PLDSD Propagated Linked Data Semantic Distance. 16, 17, 32

RDBMS Relational Database Management System. 11

RDF Resource Description Framework. 9, 11, 14, 24, 31, 46, 57, 70, 91

Resim Resource Similarity. xi, xiii, 16–20, 23, 32, 76, 77, 79, 81–84

RMSE root mean squared error. 31

RS Recommender Systems. xiii, xiv, 1–7, 15, 18, 27–33, 36, 37, 63–67, 80, 81, 84–88, 94

SPARQL SPARQL Protocol and RDF Query Language. 14, 15, 46, 47, 57, 58, 70, 85

URI Uniform Resource Identifier. 9, 20

W3C World Wide Web Consortium. 9, 10, 14

Bibliography

[ABH+21] Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings. *Journal of Machine Learning Research*, 22(82):1–6, 2021.

[ADNDS+21] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Antonio Ferrara, and Alberto Carlo Maria Mancino. Sparse feature factorization for recommender systems with knowledge graphs. In *Proceedings of the 15th ACM Conference on Recommender Systems*, pages 154–165, 2021.

[ALG17] Sultan Alfarhood, Kevin Labille, and Susan Gauch. Pldsd: propagated linked data semantic distance. In *2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pages 278–283. IEEE, 2017.

[BSY+20] Ye Bi, Liqiang Song, Mengqiu Yao, Zhenyu Wu, Jianming Wang, and Jing Xiao. Dcdir: A deep cross-domain recommendation system for cold start users in insurance domain. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pages 1661–1664, 2020.

[BUGD+13] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[CCCD19] Haihua Chen, Gaohui Cao, Jiangping Chen, and Junhua Ding. A practical framework for evaluating the quality of knowledge graph. In *Knowledge Graph and Semantic Computing: Knowledge Computing and Language Understanding: 4th China Conference, CCKS 2019, Hangzhou, China, August 24–27, 2019, Revised Selected Papers 4*, pages 111–122. Springer, 2019.

[CLZ+22] Huiyuan Chen, Xiaoting Li, Kaixiong Zhou, Xia Hu, Chin-Chia Michael Yeh, Yan Zheng, and Hao Yang. Tinykg: Memory-efficient training framework for knowledge graph neural recommender systems. In *Proceedings* *of the 16th ACM Conference on Recommender Systems*, pages 257–267, 2022.

[CMEC17] Rose Catherine, Kathryn Mazaitis, Maxine Eskenazi, and William Cohen. Explainable entity-based recommendations with knowledge graphs. *arXiv preprint arXiv:1707.05254*, 2017.

[CQA$^+$19a] Shruthi Chari, Miao Qi, Nkechinyere N Agu, Oshani Seneviratne, James P McCusker, Kristin P Bennett, Amar K Das, and Deborah L McGuinness. Making study populations visible through knowledge graphs. In *International Semantic Web Conference*, pages 53–68. Springer, 2019.

[CQA$^+$19b] Shruthi Chari, Miao Qi, Nkechinyere N Agu, Oshani Seneviratne, James P McCusker, Kristin P Bennett, Amar K Das, and Deborah L McGuinness. Ontology-enabled analysis of study populations. In *ISWC Satellites*, pages 117–120, 2019.

[CVD21] Janneth Chicaiza and Priscila Valdiviezo-Diaz. A comprehensive survey of knowledge graph-based recommender systems: Technologies, development, and contributions. *Information*, 12(6):232, 2021.

[CWH$^+$19] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The world wide web conference*, pages 151–161, 2019.

[CWT22] Pengfei Chen, Qi Wang, and Yuan Tian. Exploring entity-level user preference on the knowledge graph for recommender system. In *Proceedings of the 2022 5th International Conference on Algorithms, Computing and Artificial Intelligence*, pages 1–7, 2022.

[CWX$^+$15] Wenliang Cheng, Chengyu Wang, Bing Xiao, Weining Qian, and Aoying Zhou. On statistical characteristics of real-life knowledge graphs. In *BPOE*, pages 37–49. Springer, 2015.

[DAL16] Jeremy Debattista, Sören Auer, and Christoph Lange. Luzzu—a methodology and framework for linked data quality assessment. *Journal of Data and Information Quality (JDIQ)*, 8(1):1–32, 2016.

[DJM21] Chunfang Dong, Xuchan Ju, and Yue Ma. Hrs: Hybrid recommendation system based on attention mechanism and knowledge graph embedding. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 406–413, 2021.

[DLD19] Xin Dong, Tong Li, and Zhiming Ding. An ontology enhanced user profiling algorithm based on application feedback. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, pages 316–325. IEEE, 2019.

[DTRP19] Amine Dadoun, Raphaël Troncy, Olivier Ratier, and Riccardo Petitti. Location embeddings for next trip recommendation. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 896–903, 2019.

[EH22] Ehsan Elahi and Zahid Halim. Graph attention-based collaborative filtering for user-specific recommender system using knowledge graph and deep neural networks. *Knowledge and Information Systems*, 64(9):2457–2480, 2022.

[EW16] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2, 2016.

[FBMR18] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129, 2018.

[FH11] Christian Fürber and Martin Hepp. Towards a vocabulary for data quality management in semantic web architectures. In *Proceedings of the 1st International Workshop on Linked Web Data Management*, pages 1–8, 2011.

[FLH$^+$23] Ziwei Fan, Zhiwei Liu, Shelby Heinecke, Jianguo Zhang, Huan Wang, Caiming Xiong, and Philip S Yu. Zero-shot item-based recommendation via multi-task product knowledge graph pre-training. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 483–493, 2023.

[FR18] Michael Färber and Achim Rettinger. Which knowledge graph is best for me? *arXiv preprint arXiv:1809.11099*, 2018.

[FT23] Anna Formica and Francesco Taglino. Semantic relatedness in dbpedia: A comparative and experimental assessment. *Information Sciences*, 621:474–505, 2023.

[FZZG22] Huilian Fan, Yuanchang Zhong, Guangpu Zeng, and Chenhao Ge. Improving recommender system via knowledge graph based exploring user preference. *Applied Intelligence*, pages 1–13, 2022.

[GLX$^+$19] Junyang Gao, Xian Li, Yifan Ethan Xu, Bunyamin Sisman, Xin Luna Dong, and Jun Yang. Efficient knowledge graph accuracy evaluation. *arXiv preprint arXiv:1907.09657*, 2019.

[GNLC21] Lu Gan, Diana Nurbakova, Léa Laporte, and Sylvie Calabretto. Emdkg: Improving accuracy-diversity trade-off in recommendation with em-based model and knowledge graph embedding. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 17–24, 2021.

[GS15] Asela Gunawardana and Guy Shani. Evaluating recommender systems. In *Recommender systems handbook*, pages 265–308. Springer, 2015.

[GSZT19] Qing Guo, Zhu Sun, Jie Zhang, and Yin-Leng Theng. Modeling heterogeneous influences for point-of-interest recommendation in location-based social networks. In *International Conference on Web Engineering*, pages 72–80. Springer, 2019.

[GZQ$^+$20] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[Had18] A Hadhiatma. Improving data quality in the linked open data: a survey, vol. 978, 2018.

[HBC$^+$21] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs. *ACM Computing Surveys (CSUR)*, 54(4):1–37, 2021.

[HFKP20] Armin Haller, Javier D Fernández, Maulik R Kamdar, and Axel Polleres. What are links in linked open data? a characterization and evaluation of links between knowledge graphs on the web. *Journal of Data and Information Quality (JDIQ)*, 12(2):1–34, 2020.

[HFQ$^+$19] Xiaowen Huang, Quan Fang, Shengsheng Qian, Jitao Sang, Yan Li, and Changsheng Xu. Explainable interaction-driven user modeling over knowledge graph for sequential recommendation. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 548–556, 2019.

[HHC21] Ruoran Huang, Chuanqi Han, and Li Cui. Entity-aware collaborative relation network with knowledge graph for recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3098–3102, 2021.

[HHL$^+$21] Bernd Heinrich, Marcus Hopf, Daniel Lohninger, Alexander Schiller, and Michael Szubartowicz. Data quality in recommender systems: the impact of completeness of item content data on prediction accuracy of recommender systems. *Electronic Markets*, 31(2):389–409, 2021.

[HHRP20] Nicolas Heist, Sven Hertling, Daniel Ringler, and Heiko Paulheim. Knowledge graphs on the web-an overview. *Knowledge Graphs for eXplainable Artificial Intelligence*, pages 3–22, 2020.

[HK15] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[HLR$^+$21] Lavdim Halilaj, Jürgen Lüttin, Susanne Rothermel, Santhosh Kumar Arumugam, and Ishan Dindorkar. Towards a knowledge graph-based approach for context-aware points-of-interest recommendations. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, pages 1846–1854, 2021.

[HM18] Monika Rani HG and Shakti Mishra. An investigative study on the quality aspects of linked open data. In *Proceedings of the 2018 International Conference on Cloud Computing and Internet of Things*, pages 33–39, 2018.

[HMvdW$^+$20] Charles R. Harris, K. Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020.

[HS19] Hao Hou and Chongyang Shi. Explainable sequential recommendation using knowledge graphs. In *Proceedings of the 5th International Conference on Frontiers of Educational Technologies*, pages 53–57, 2019.

[HTWX19] Sihang Hu, Zhiying Tu, Zhongjie Wang, and Xiaofei Xu. A poi-sensitive knowledge graph based service recommendation method. In *2019 IEEE International Conference on Services Computing (SCC)*, pages 197–201. IEEE, 2019.

[Hun07] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[JPC$^+$21] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514, 2021.

[KKS21] M. Kejriwal, C.A. Knoblock, and P. Szekely. *Knowledge Graphs: Fundamentals, Techniques, and Applications*. Adaptive Computation and Machine Learning series. MIT Press, 2021.

[KLC23] Gurinder Kaur, Fei Liu, and Yi-Ping Phoebe Chen. A deep learning knowledge graph neural network for recommender systems. *Machine Learning with Applications*, 14:100507, 2023.

[KS21] Miriyala Kartheek and GP Sajeev. Building semantic based recommender system using knowledge graph embedding. In *2021 sixth international* *conference on image information processing (ICIIP)*, volume 6, pages 25–29. IEEE, 2021.

[KW16] Markus Krötzsch and Gerhard Weikum. *Journal of Web Semantics*, 37-38:53–54, 2016.

[KWA+14] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. Test-driven evaluation of linked data quality. In *Proceedings of the 23rd international conference on World Wide Web*, pages 747–758, 2014.

[LAS23] Seungjoo Lee, Seokho Ahn, and YoungDuk Seo. Relation modeling on knowledge graph for interoperability in recommender systems. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, pages 751–758, 2023.

[LCPR19] Xinyi Li, Yifan Chen, Benjamin Pettit, and Maarten De Rijke. Personalised reranking of paper recommendations using paper content and user behavior. *ACM Transactions on Information Systems (TOIS)*, 37(3):1–23, 2019.

[LHL23] Yakun Li, Lei Hou, and Juanzi Li. Preference-aware graph attention networks for cross-domain recommendations with collaborative knowledge graph. *ACM Transactions on Information Systems*, 41(3):1–26, 2023.

[LIJ+15] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195, 2015.

[LMC22] Yun Liu, Jun Miyazaki, and Qiong Chang. Knowledge graph enhanced multi-task learning between reviews and ratings for movie recommendation. In *Proceedings of the 37th ACM/SIGAPP symposium on applied computing*, pages 1882–1889, 2022.

[LTW+19] Qianyu Li, Xiaoli Tang, Tengyun Wang, Haizhi Yang, and Hengjie Song. Unifying task-oriented knowledge graph learning and recommendation. *IEEE Access*, 7:115816–115828, 2019.

[LXJ+20] Zhaopeng Li, Qianqian Xu, Yangbangyan Jiang, Xiaochun Cao, and Qingming Huang. Quaternion-based knowledge graph network for recommendation. In *Proceedings of the 28th ACM international conference on multimedia*, pages 880–888, 2020.

[MD15] Rouzbeh Meymandpour and Joseph G Davis. Enhancing recommender systems using linked open data-based semantic analysis of items. In *AWC*, pages 11–17, 2015.

[MD16] Rouzbeh Meymandpour and Joseph G Davis. A semantic similarity measure for linked data: An information content-based approach. *Knowledge-Based Systems*, 109:276–293, 2016.

[MLU22] Cedric Möller, Jens Lehmann, and Ricardo Usbeck. Survey on english entity linking on wikidata: Datasets and approaches. *Semantic Web*, 13(6):925–966, 2022.

[MMB12] Pablo N Mendes, Hannes Mühleisen, and Christian Bizer. Sieve: linked data quality assessment and fusion. In *Proceedings of the 2012 joint EDBT/ICDT workshops*, pages 116–123, 2012.

[MMM21] Sudhir P Mudur, Serguei A Mokhov, and Yuhao Mao. A framework for enhancing deep learning based recommender systems with knowledge graphs. In *Proceedings of the 25th International Database Engineering & Applications Symposium*, pages 11–20, 2021.

[MT18] Michalis Mountantonakis and Yannis Tzitzikas. Scalable methods for measuring the connectivity and quality of large numbers of linked datasets. *Journal of Data and Information Quality (JDIQ)*, 9(3):1–49, 2018.

[MZC$^+$19] Weizhi Ma, Min Zhang, Yue Cao, Woojeong Jin, Chenyang Wang, Yiqun Liu, Shaoping Ma, and Xiang Ren. Jointly learning explainable rules for recommendation with knowledge graph. In *The World Wide Web Conference*, pages 1210–1221, 2019.

[NM$^+$01] Natalya F Noy, Deborah L McGuinness, et al. Ontology development 101: A guide to creating your first ontology, 2001.

[NPAB20] Vanshika Vikas Nigam, Shreya Paul, Arun Prakash Agrawal, and Rishabh Bansal. A review paper on the application of knowledge graph on various service providing platforms. In *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 716–720. IEEE, 2020.

[NTK$^+$11] Maximilian Nickel, Volker Tresp, Hans-Peter Kriegel, et al. A three-way model for collective learning on multi-relational data. In *Icml*, volume 11, pages 3104482–3104584, 2011.

[P$^+$13] Heiko Paulheim et al. Exploiting linked open data as background knowledge in data mining. *DMoLD*, 1082, 2013.

[PAB16] Guangyuan Piao, Safina Showkat Ara, and John G Breslin. Computing the semantic similarity of resources in dbpedia for recommendation purposes. In *Semantic Technology: 5th Joint International Conference, JIST 2015, Yichang, China, November 11-13, 2015, Revised Selected Papers 5*, pages 185–200. Springer, 2016.

[Pas10] Alexandre Passant. Measuring semantic distance on linking data and using it for resources recommendations. In *2010 AAAI Spring Symposium Series*, 2010.

[PB14] Heiko Paulheim and Christian Bizer. Improving the quality of linked data using statistical distributions. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):63–86, 2014.

[PB16] Guangyuan Piao and John G Breslin. Measuring semantic distance for linked open data-enabled recommender systems. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 315–320, 2016.

[PB18] Guangyuan Piao and John G Breslin. A study of the similarities of entity embeddings learned from different aspects of a knowledge base for item recommendations. In *European Semantic Web Conference*, pages 345–359. Springer, 2018.

[pdt20] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[PTWS20] Thomas Pellissier Tanon, Gerhard Weikum, and Fabian Suchanek. Yago 4: A reason-able knowledge base. In *The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17*, pages 583–596. Springer, 2020.

[PVG$^+$11] 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. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[QBZ$^+$19] Yanru Qu, Ting Bai, Weinan Zhang, Jianyun Nie, and Jian Tang. An end-to-end neighborhood-based interaction model for knowledge-enhanced recommendation. In *Proceedings of the 1st international workshop on deep learning practice for high-dimensional sparse data*, pages 1–9, 2019.

[Ris19] Petar Ristoski. *Exploiting semantic web knowledge graphs in data mining*, volume 38. IOS Press, 2019.

[RMGCGP18] Filip Radulovic, Nandana Mihindukulasooriya, Raúl García-Castro, and Asunción Gómez-Pérez. A comprehensive quality model for linked data. *Semantic Web*, 9(1):3–24, 2018.

[RP16a] Petar Ristoski and Heiko Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *The Semantic Web–ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part I 15*, pages 498–514. Springer, 2016.

[RP16b] Petar Ristoski and Heiko Paulheim. Semantic web in data mining and knowledge discovery: A comprehensive survey. *Journal of Web Semantics*, 36:1–22, 2016.

[RRDN+16] Jessica Rosati, Petar Ristoski, Tommaso Di Noia, Renato de Leone, and Heiko Paulheim. Rdf graph embeddings for content-based recommender systems. In *CEUR workshop proceedings*, volume 1673, pages 23–30. RWTH Aachen, 2016.

[RRDN+19] Petar Ristoski, Jessica Rosati, Tommaso Di Noia, Renato De Leone, and Heiko Paulheim. Rdf2vec: Rdf graph embeddings and their applications. *Semantic Web*, 10(4):721–752, 2019.

[SA19] Vineet Kumar Sejwal and Muhammad Abulaish. Context-based rating prediction using collaborative filtering and linked open data. In *Proceedings of the 9th International Conference on Web Intelligence, Mining and Semantics*, pages 1–9, 2019.

[SAB+24] Fabian M Suchanek, Mehwish Alam, Thomas Bonald, Lihu Chen, Pierre-Henri Paris, and Jules Soria. Yago 4.5: A large and clean knowledge base with a rich taxonomy. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 131–140, 2024.

[SCZ+20] Rui Sun, Xuezhi Cao, Yan Zhao, Junchen Wan, Kun Zhou, Fuzheng Zhang, Zhongyuan Wang, and Kai Zheng. Multi-modal knowledge graphs for recommender systems. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1405–1414, 2020.

[Sha48] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[Sny19] Hannah Snyder. Literature review as a research methodology: An overview and guidelines. *Journal of business research*, 104:333–339, 2019.

[SOK19] Takafumi Suzuki, Satoshi Oyama, and Masahito Kurihara. Explainable recommendation using review text and a knowledge graph. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 4638–4643. IEEE, 2019.

[SSZ21] Xiao Sha, Zhu Sun, and Jie Zhang. Hierarchical attentive knowledge graph embedding for personalized recommendation. *Electronic Commerce Research and Applications*, 48:101071, 2021.

[STC+20] C Santhoshi, V Thirupathi, KR Chythanya, S Aluvala, and G Sunil. A comprehensive study on efficient keyword-aware representative travel route recommendation. *Int. J. Adv. Sci. Technol*, 29:1800–1810, 2020.

[SVAO23] Bram Steenwinckel, Gilles Vandewiele, Terencio Agozzino, and Femke Ongenae. pyrdf2vec: A python implementation and extension of rdf2vec. In *European Semantic Web Conference*, pages 471–483. Springer Nature Switzerland, 2023.

[TCW+21] Ke Tu, Peng Cui, Daixin Wang, Zhiqiang Zhang, Jun Zhou, Yuan Qi, and Wenwu Zhu. Conditional graph attention networks for distilling and refining knowledge graphs in recommendation. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 1834–1843, 2021.

[TFTCDN17] Paolo Tomeo, Ignacio Fernández-Tobías, Iván Cantador, and Tommaso Di Noia. Addressing the cold start with positive-only feedback through semantic-based recommendations. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 25(Suppl. 2):57–78, 2017.

[Tve77] Amos Tversky. Features of similarity. *Psychological review*, 84(4):327, 1977.

[TWYS19] Xiaoli Tang, Tengyun Wang, Haizhi Yang, and Hengjie Song. Akupm: Attention-enhanced knowledge-aware user preference model for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1891–1899, 2019.

[VGO+20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[Was21] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.

[WHC+19] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 950–958, 2019.

[WHW+21] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the web conference 2021*, pages 878–887, 2021.

[WK17] Albert Weichselbraun and Philipp Kuntschik. Mitigating linked data quality issues in knowledge-intense information extraction methods. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics*, pages 1–12, 2017.

[WL20] Fulin Wang and Yun Li. An auto question answering system for tree hole rescue. In *International Conference on Health Information Science*, pages 15–24. Springer, 2020.

[WLF$^+$21] Yu Wang, Zhiwei Liu, Ziwei Fan, Lichao Sun, and Philip S Yu. Dskreg: Differentiable sampling on knowledge graph for recommendation with relational gnn. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3513–3517, 2021.

[WLZL21] Yani Wang, Aoran Li, Ji Zhang, and Bohan Li. Enhanced knowledge graph embedding for multi-task recommendation via integrating attribute information and high-order connectivity. In *Proceedings of the 10th International Joint Conference on Knowledge Graphs*, pages 140–144, 2021.

[WLZW22] Fei Wang, Yansheng Li, Yongjun Zhang, and Dong Wei. Klgcn: Knowledge graph-aware light graph convolutional network for recommender systems. *Expert Systems with Applications*, 195:116513, 2022.

[WM10] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[WRH$^+$12] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[WWH$^+$19] Haifang Wang, Zhongjie Wang, Sihang Hu, Xiaofei Xu, Shiping Chen, and Zhiying Tu. Duskg: A fine-grained knowledge graph for effective personalized service recommendation. *Future Generation Computer Systems*, 100:600–617, 2019.

[WWX$^+$19] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5329–5336, 2019.

[WXZY20] Qingqin Wang, Yun Xiong, Yangyong Zhu, and Philip S Yu. Kasr: knowledge-aware sequential recommendation. In *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*, pages 493–508. Springer, 2020.

[WZW+18] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 417–426, 2018.

[WZW+19] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Exploring high-order user preference on the knowledge graph for recommender systems. *ACM Transactions on Information Systems (TOIS)*, 37(3):1–26, 2019.

[WZX+19] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. Knowledge graph convolutional networks for recommender systems. corr abs/1904.12575 (2019). *arXiv preprint arXiv:1904.12575*, 2019.

[WZZ+19] Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Multi-task feature learning for knowledge graph enhanced recommendation. In *The World Wide Web Conference*, pages 2000–2010, 2019.

[WZZH20] Jie Wu, Xinning Zhu, Chunhong Zhang, and Zheng Hu. Event-centric tourism knowledge graph – a case study of hainan. In *International conference on knowledge science, engineering and management*, pages 3–15. Springer, 2020.

[XCZ+22] Yong Xu, Bao Chen, Jingru Zhen, Guoqing Ma, Gongbin Chen, Yan Liu, and Qun Fang. Nrkm: News recommendation based on knowledge graph with multi-view learning. In *Proceedings of the 2022 3rd international conference on control, robotics and intelligent system*, pages 123–127, 2022.

[XFM+19] Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 285–294, 2019.

[XHZ+19] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon Jose. Relational collaborative filtering: Modeling multiple item relations for recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 125–134, 2019.

[XZ22] Bingcong Xue and Lei Zou. Knowledge graph quality management: a comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4969–4988, 2022.

[YHXH23] Yuhao Yang, Chao Huang, Lianghao Xia, and Chunzhen Huang. Knowledge graph self-supervised rationalization for recommendation. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, pages 3046–3056, 2023.

[YHXL22] Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1434–1443, 2022.

[YWY+19] Yuting Ye, Xuwu Wang, Jiangchao Yao, Kunyang Jia, Jingren Zhou, Yanghua Xiao, and Hongxia Yang. Bayes embedding (bem) refining representation by integrating knowledge graphs and behavior-specific networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 679–688, 2019.

[YYH+14] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[ZDC+20] Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pages 179–188, 2020.

[ZHY+19] Wayne Xin Zhao, Gaole He, Kunlin Yang, Hongjian Dou, Jin Huang, Siqi Ouyang, and Ji-Rong Wen. Kb4rec: A data set for linking knowledge bases with recommender systems. *Data Intelligence*, 1(2):121–136, 2019.

[ZLSW19] Siqi Zhu, Yi Li, Yanqiu Shao, and Lihui Wang. Building semantic dependency knowledge graph based on hownet. In *Workshop on Chinese Lexical Semantics*, pages 525–534. Springer, 2019.

[ZRM+16] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Soeren Auer. Quality assessment for linked data: A survey. *Semantic Web*, 7(1):63–93, 2016.

[ZWL20] Yuhang Zhang, Jun Wang, and Jie Luo. Knowledge graph embedding based collaborative filtering. *IEEE Access*, 8:134553–134562, 2020.

[ZZB+20] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1006–1014, 2020.

[ZZG$^+$19] Jun Zhao, Zhou Zhou, Ziyu Guan, Wei Zhao, Wei Ning, Guang Qiu, and Xiaofei He. Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2347–2357, 2019.