Leveraging Human-Computer Interaction and Crowdsourcing for Scholarly Knowledge Graph Creation

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des akademischen Grades

Doktor-Ingenieur
(Dr.-Ing.)

genehmigte Dissertation

von Herrn
Allard Oelen, M.Sc.
geboren am
01.09.1994
in
Amsterdam, Niederlande

2022

1. Referent: Prof. Dr. Sören Auer
2. Referent: Prof. Dr. Tobias Kuhn

Tag der Promotion: 15.11.2022

Abstract

The number of scholarly publications continues to grow each year, as well as the number of journals and active researchers. Therefore, methods and tools to organize scholarly knowledge are becoming increasingly important. Without such tools, it becomes increasingly difficult to conduct research in an efficient and effective manner. One of the fundamental issues scholarly communication is facing relates to the format in which the knowledge is shared. Scholarly communication relies primarily on narrative document-based formats that are specifically designed for human consumption. Machines cannot easily access and interpret such knowledge, leaving machines unable to provide powerful tools to organize scholarly knowledge effectively.

In this thesis, we propose to leverage knowledge graphs to represent, curate, and use scholarly knowledge. The systematic knowledge representation leads to machine-actionable knowledge, which enables machines to process scholarly knowledge with minimal human intervention. To generate and curate the knowledge graph, we propose a machine learning assisted crowdsourcing approach, in particular Natural Language Processing (NLP). Currently, NLP techniques are not able to satisfactorily extract high-quality scholarly knowledge in an autonomous manner. With our proposed approach, we intertwine human and machine intelligence, thus exploiting the strengths of both approaches.

First, we discuss structured scholarly knowledge, where we present the Open Research Knowledge Graph (ORKG). Specifically, we focus on the design and development of the ORKG user interface (i.e., the frontend). One of the key challenges is to provide an interface that is powerful enough to create rich knowledge descriptions yet intuitive enough for researchers without a technical background to create such descriptions. The ORKG serves as the technical foundation for the rest of the work. Second, we focus on comparable scholarly knowledge, where we introduce the concept of ORKG comparisons. ORKG comparisons provide machine-actionable overviews of related literature in a tabular form. Also, we present a methodology to leverage existing literature reviews to populate ORKG comparisons via a human-in-the-loop approach. Additionally, we show how ORKG comparisons can be used to form ORKG SmartReviews. The SmartReviews provide dynamic literature reviews in the form of living documents. They are an attempt address the main weaknesses of the current literature review practice and outline how the future of review publishing can look like. Third, we focus designing suitable tasks to generate scholarly knowledge in a crowdsourced setting. We present an intelligent user interface that enables researchers to

annotate key sentences in scholarly publications with a set of discourse classes. During this process, researchers are assisted by suggestions coming from NLP tools. In addition, we present an approach to validate NLP-generated statements using microtasks in a crowdsourced setting. With this approach, we lower the barrier to entering data in the ORKG and transform content consumers into content creators.

With the work presented, we strive to transform scholarly communication to improve machine-actionability of scholarly knowledge. The approaches and tools are deployed in a production environment. As a result, the majority of the presented approaches and tools are currently in active use by various research communities and already have an impact on scholarly communication.

Zusammenfassung

Die Zahl der wissenschaftlichen Veröffentlichungen nimmt jedes Jahr weiter zu, ebenso wie die Zahl der Zeitschriften und der aktiven Forscher. Daher werden Methoden und Werkzeuge zur Organisation von wissenschaftlichem Wissen immer wichtiger. Ohne solche Werkzeuge wird es immer schwieriger, Forschung effizient und effektiv zu betreiben. Eines der grundlegenden Probleme, mit denen die wissenschaftliche Kommunikation konfrontiert ist, betrifft das Format, in dem das Wissen publiziert wird. Die wissenschaftliche Kommunikation beruht in erster Linie auf narrativen, dokumentenbasierten Formaten, die speziell für Experten konzipiert sind. Maschinen können auf dieses Wissen nicht ohne weiteres zugreifen und es interpretieren, so dass Maschinen nicht in der Lage sind, leistungsfähige Werkzeuge zur effektiven Organisation von wissenschaftlichem Wissen bereitzustellen.

In dieser Arbeit schlagen wir vor, Wissensgraphen zu nutzen, um wissenschaftliches Wissen darzustellen, zu kuratieren und zu nutzen. Die systematische Wissensrepräsentation führt zu maschinenverarbeitbarem Wissen. Dieses ermöglicht es Maschinen wissenschaftliches Wissen mit minimalem menschlichen Eingriff zu verarbeiten. Um den Wissensgraphen zu generieren und zu kuratieren, schlagen wir einen Crowdsourcing-Ansatz vor, der durch maschinelles Lernen unterstützt wird, insbesondere durch natürliche Sprachverarbeitung (NLP). Derzeit sind NLP-Techniken nicht in der Lage, qualitativ hochwertiges wissenschaftliches Wissen auf autonome Weise zu extrahieren. Mit unserem vorgeschlagenen Ansatz verknüpfen wir menschliche und maschinelle Intelligenz und nutzen so die Stärken beider Ansätze.

Zunächst erörtern wir strukturiertes wissenschaftliches Wissen, wobei wir den Open Research Knowledge Graph (ORKG) vorstellen. Insbesondere konzentrieren wir uns auf das Design und die Entwicklung der ORKG-Benutzeroberfläche (das Frontend). Eine der größten Herausforderungen besteht darin, eine Schnittstelle bereitzustellen, die leistungsfähig genug ist, um umfangreiche Wissensbeschreibungen zu erstellen und gleichzeitig intuitiv genug ist für Forscher ohne technischen Hintergrund, um solche Beschreibungen zu erstellen. Der ORKG dient als technische Grundlage für die Arbeit.

Zweitens konzentrieren wir uns auf vergleichbares wissenschaftliches Wissen, wofür wir das Konzept der ORKG-Vergleiche einführen. ORKG-Vergleiche bieten maschinell verwertbare Übersichten über verwandtes wissenschaftliches Wissen in tabellarischer Form. Außerdem stellen wir eine Methode vor, mit der vorhandene Literaturübersichten genutzt werden

können, um ORKG-Vergleiche mit Hilfe eines Human-in-the-Loop-Ansatzes zu erstellen. Darüber hinaus zeigen wir, wie ORKG-Vergleiche verwendet werden können, um ORKG SmartReviews zu erstellen. Die SmartReviews bieten dynamische Literaturübersichten in Form von lebenden Dokumenten. Sie stellen einen Versuch dar, die Hauptschwächen der gegenwärtigen Praxis des Literaturreviews zu beheben und zu skizzieren, wie die Zukunft der Veröffentlichung von Reviews aussehen kann.

Drittens konzentrieren wir uns auf die Gestaltung geeigneter Aufgaben zur Generierung von wissenschaftlichem Wissen in einer Crowdsourced-Umgebung. Wir stellen eine intelligente Benutzeroberfläche vor, die es Forschern ermöglicht, Schlüsselsätze in wissenschaftlichen Publikationen mittles Diskursklassen zu annotieren. In diesem Prozess werden Forschende mit Vorschlägen von NLP-Tools unterstützt. Darüber hinaus stellen wir einen Ansatz zur Validierung von NLP-generierten Aussagen mit Hilfe von Mikroaufgaben in einer Crowdsourced-Umgebung vor. Mit diesem Ansatz senken wir die Hürde für die Eingabe von Daten in den ORKG und setzen Inhaltskonsumenten als Inhaltsersteller ein.

Mit der Arbeit streben wir eine Transformation der wissenschaftlichen Kommunikation an, um die maschinelle Verwertbarkeit von wissenschaftlichem Wissen zu verbessern. Die Ansätze und Werkzeuge werden in einer Produktionsumgebung eingesetzt. Daher werden die meisten der vorgestellten Ansätze und Werkzeuge derzeit von verschiedenen Forschungsgemeinschaften aktiv genutzt und haben bereits einen Einfluss auf die wissenschaftliche Kommunikation.

Preface

When I started with my Ph.D. in 2019, the world was a different place. For most people, Corona was merely a beer brand, not the shape of a virus that would cause a pandemic and worldwide lockdown. I was lucky enough to start my Ph.D. before the pandemic, so I was able to experience the true Ph.D.-life. This includes late nights at the office, conferences, project meetings, and summer schools. I specifically enjoyed in-person conferences, where you were able to discuss your work with fellow researchers. Such discussions really motivated and gave me many ideas for my own work. I want to take this opportunity to thank the people that helped me during this special journey.

First of all, I want to thank Prof. Dr. Sören Auer, who has been a great teacher along the way. Not only did he provide guidance when needed, but he also gave me the freedom and encouragement to explore my own ideas. With his positive attitude and persuasiveness, he was always able to motivate me and make me push even harder to accomplish my goals. Furthermore, I want to thank my daily supervisor Dr. Markus Stocker. The many discussions we had after work time were not only very valuable to me personally, but also often made me go home with a smile. His door was always open and he allowed me to discuss my latest (sometimes mad) ideas. No matter what we discussed, I could always count on constructive feedback. I also want to thank Mohamad Yaser Jaradeh, he was my first "office mate" when I arrived in Hannover in 2019 and still is sitting there to date. Normally, he was my go-to person to discuss (research) ideas. After the standard ironic "sounds like a bad idea" comment, I was sure to always get in-depth feedback that helped me to continue. I want to sincerely thank the rest of the ORKG team, Anna-Lena, Golsa, Hassan, Jennifer, Kheir Eddine, Lars, Manuel, Muhammad, Oliver, Salomon, Suhas, and Vitalis. When I started, our group was relatively small in size. Over the years, the team has been growing a lot which has been a great experience. Although the group is larger, we still have a lot of direct communication and a flat hierarchy. This makes working in the ORKG team a very pleasant experience.

Also, I want to thank Dr. Natalia Silvis-Cividjian. In my second year of the bachelor's program at Vrije Universiteit Amsterdam, she asked me to become a student assistant. Being a student assistant helped me to learn more than I ever imagined. Also because of this experience, I started appreciating education and research more. From that year on, I have assisted many other courses, which made me a "university dweller". These experiences also

made me realize that I wanted to continue in academia after my master's studies. I also want to thank Dr. Victor de Boer, my bachelor's and master's supervisor. He thought me how to do proper research and advised me many times about pursuing a Ph.D.

Finally, I want to thank my parents for their support during my studies. Already from a young age, they taught me to be curious and open-minded. When I was young, my mother often took me to the many beautiful museums in Amsterdam and persuaded me to come along with the promise of a visit to the gift shop. My father always supported learning, for example, he was always willing to pay for whatever educational book I wanted. These experiences made me appreciate science and are undoubtedly the reason why went to university in the first place. Thank you. I also want to thank my brother, Casper. He has been of great support during my studies.

Contents

CHAPTER 1

Introduction

Knowledge is power, a well-known proverb attributed to Sir Francis Bacon [1], stresses the importance of knowledge over other qualities such as physical strength, financial wealth, or even wisdom. The saying is arguably controversial as one might dispute whether the desire for power should be pursued at all. Less controversial but equally valid, if not more so, is the phrase: *Knowledge is empowering*. Knowledge empowers people in many aspects; to make better and informed decisions, to solve complex problems, and to understand the world around us. One of the key pillars of modern society is the ability to access and acquire knowledge in a virtually limitless fashion. Acquiring knowledge starts already at a young age through perception, memory, and practice. When growing older, education plays a crucial role in acquiring knowledge. The knowledge transferred through education generally has its foundation in scientific inquiry.

According to UNESCO, science is the largest global collective endeavor to acquire knowledge collaboratively [2]. Scientific research focuses on solving questions and problems which can potentially impact society as a whole. For example, Life Sciences, to fight diseases and increase health and life span. Earth Sciences where preservation of the planet is pursued by addressing global warming. Computer Sciences, which over the last decades changed the way we live and work. Science has an obligation to research global issues and provide solutions to those issues.

By definition, scientific knowledge is produced by following the scientific method. Although the scientific method materializes variably, it follows a basic general pattern. First, from specific observations, research questions and hypotheses are formed. Then, after experimenting, conclusions are formed about the hypothesis. Afterwards, the complete method, including the results, is reviewed by other scientists. Finally, the results are communicated and distributed among the respective scientific communities.

The communication of research falls under the umbrella term Scholarly Communication. It involves the creation, publication, dissemination, and discovery of scholarly research and is a crucial aspect of scientific research in general [3, 4]. Without the ability to effectively

communicate research results with other researchers, the work cannot be read and reused and will be lost over time. Especially using existing scientific findings enables researchers to build upon already conducted efforts and advance the state-of-the-art. After all, it is the collective effort that makes science this impactful. In the end, small contributions made by individual researchers, or research groups, contribute to solving big challenges.

In this thesis, we aim to improve the different steps and processes of Scholarly Communication. This includes the previously mentioned aspects: creation, publication, dissemination, and discovery of scholarly research. Our effort is guided by the technological advancements of the World Wide Web in the last few decades and, in particular, related to Semantic Web technologies. We advocate for scholarly knowledge that is represented and communicated in a machine-readable manner. We regard scholarly knowledge as the actual knowledge presented, primarily, within scholarly articles. This knowledge results from following the scientific method and the existence of this knowledge is the reason why a scholarly article was authored in the first place, i.e., to communicate the knowledge. The knowledge itself is the *contribution* to research communities, or more broadly to science in general. By embracing novel technologies to communicate scholarly knowledge, we improve the effectiveness and efficiency of this process and provide a means to enhance Scholarly Communication.

1.1 Motivation

The number of scholarly publications, journals and researchers continues to grow [5, 6]. With such growth, novel methods to more efficiently organize and communicate scholarly knowledge become increasingly urgent.

Without the ability to efficiently communicate the produced knowledge with other researchers, the research impact is limited, and knowledge can get lost along the way. We consider efficient scholarly communication as a process where retrieved results have both high recall and precision. This contributes to researcher's ability to oversee the state-of-the-art at a glance. Today, researchers do unknowingly address same or similar research challenges. Besides the risk of redundant work, this also hinders researchers to learn from already gathered experiences. This can have a significant negative impact on the usefulness of the conducted research. Furthermore, reading existing literature is a recurring and obligatory task that is part of any research endeavor. Researchers have to familiarize themselves with the state-of-the-art to be able to make significant contributions. Finally, scholarly knowledge discovery is a crucial part of the peer-review process. To review novel literature, one has to be familiar with the current work within the field. Otherwise, making an informed decision on acceptance or providing helpful feedback to the authors would be severely limited. We consider effective scholarly communication as a grand challenge for science. Given the growing number of published scholarly articles, the challenge has to be addressed sooner rather than later to conduct research efficiently.

Figure 1.1: Scholarly articles have not changed significantly since the beginning of science. The same narrative document-based communication methods are used.

With the current methods of communicating scholarly knowledge, researchers are hindered in their efforts to communicate and discover scholarly literature. To a large extent, this problem originates from the *document-based* methods used to communicate scholarly knowledge [7]. This becomes apparent when considering articles published within journals or conference proceedings. Journals and proceedings are collections of concatenated individual research articles and are generally distributed in this fashion. Looking at the evolution of scholarly knowledge sharing, we see that not much has changed throughout the centuries. *Journal des sçavans*, published in 1665, looks remarkably similar to journals published nowadays (depicted in Figure 1.1). Although published digitally nowadays, the same document-centered approach is used to communicate knowledge. This generally means a narrative approach is taken to explain the scientific process and results. To extract information from such document-based articles, one often has to read (parts of) an article to find the information of interest. To make matters worse, the PDF format is widely used [8]. This format is specifically designed for human consumption. This entails that the format is hard to process for machines as no semantics are encoded within the document [9]. This further complicates the information extraction of even high-level structural document elements, such as sections, tables, or figures.

When considering other domains that faced technological revolution, it is remarkable that scholarly communication has not undergone any significant transformation of its own. For example, where paper mail-order catalogs were the standard several decades ago, they are entirely replaced by a new method. Instead of a digital PDF version of the mail-order catalog, radical new approaches have emerged. Digital stores provide customers with product search with facets, personalized product recommendations, and even Augmented Reality (AR) solutions. Similar radically new approaches can be seen for route maps or books. Instead of providing a PDF version of a street map, an online map system offers new ways to interact with geospatial data. E-books are generally not published in PDF format but as EPUB files. This enables customized reader settings to provide a personalized reading experience. What can be learned from these examples is that novel technologies

Figure 1.2: The Google Scholar screenshot on the left shows the number of results for a search query related to COVID-19 R0 estimates. On the right is an excerpt from a paper listed in the results. The actual information the researcher is looking for is highlighted in yellow.

are leveraged not merely to digitize, but to digitalize, i.e. transform the respective domains, their information and information systems. One of the main features of PDF files is that they resemble a physical document. Specifically, PDF documents are always rendered in the same manner, no matter the tool or device used to display the file. Within user interface communities, this digital replication of physical objects is called *skeuomorphism* [10]. When integrated in user interfaces, skeuomorphism can contribute to usability, as novel interface components are introduced in the form of known concepts. We consider the use of PDF files to communicate knowledge also a form of skeuomorphism, this time however not contributing to, but limiting the digital transformation.

Several approaches have been developed to provide researchers with search tools to find scholarly literature. Most of the methods are based on keyword-based search terms. In such systems, the full text of articles is indexed. However, generally, the information need from a researcher cannot be represented with a set of keywords but is a more specific question instead. For example, a researcher is interested in COVID-19 R0 number estimates. To find relevant papers, the researcher must formulate this information need as natural text queries with the exact same terms used in the corpus. It is inevitable that search engines return irrelevant results (low precision) while relevant results are not retrieved (low recall). The problem is illustrated in Figure 1.2.

In this thesis, we aim to address the previously described challenges of scholarly communication. We propose to leverage Knowledge Graphs to describe scholarly knowledge semantically. This results in a scholarly knowledge graph containing a wide variety of scholarly knowledge. We build upon existing Semantic Web technologies to provide machine-actionable representations of the knowledge communicated within scholarly articles. Machine-actionability relates to the ability of machines to find and interpret data without the need of humans to program the machine for this purpose. Once scholarly knowledge is available in a machine-actionable format, new possibilities emerge for machines to more efficiently help researchers in their information need. Instead of providing *passive* help (as search engines do, by providing a list of potentially relevant documents), the machine can actively fulfill the researcher's information need. For our previous example

of COVID-19 R0 estimates, this means that a list of actual R0 estimates is provided by the machine (active help), instead of a list of documents that might contain that information (passive help). This can fundamentally transform the way research is conducted.

1.2 Challenges

The use of knowledge graphs to describe scholarly knowledge imposes multiple challenges. The challenges are specifically related to working with scholarly knowledge, but some aspects relate to working with knowledge graphs in general. For each challenge, we describe how they relate to thesis chapters. Furthermore, the challenges are recurring topics in each chapter and are considered when addressing the research questions and their solutions. Challenges 2, 3, and 4 present a trade-off demanding a rational balance. We now discuss the challenges in more detail.

Challenge 1: The transformation of unstructured to structured scholarly knowledge

To create a knowledge graph containing scholarly knowledge, firstly, the knowledge has to be structured so the machine can interpret it. This means that knowledge presented within scholarly articles has to be transformed to a machine-actionable representation. This transformation process is commonly referred to as information extraction. Generally, knowledge is communicated in the form of natural language sentences. As an example, *We estimate the basic reproduction number at 3.1* is a sentence published in [11]. A structured representation of this sentence may be as follows: [Study] -[Has R0 estimate]-> [3.1]. The process of converting the unstructured natural text sentence into structured data is a complex endeavor. This process can be conducted manually by humans. However, it is time-consuming to perform this task, even for a small amount of data. Given the large number of published articles every year, a fully manual approach is not feasible. Natural Language Processing (NLP) focuses on systems that interpret natural text content. In principle, NLP tools can be leveraged to create structured descriptions from scholarly articles automatically. However, they are currently not sufficiently accurate to generate high-quality results. This means that generated descriptions might contain wrong information or that they are incomplete. This stands in stark contrast with the fact that data quality is a crucial aspect for a scholarly knowledge graph to be a valuable research asset. NLP tools will likely further improve as computing power increases and further NLP research is conducted. However, the application of NLP on scholarly text will remain a challenging task also because of the high demands on domain knowledge and thus tools that are specifically trained to perform information extraction tasks on such content. We propose a hybrid approach that integrates both human and machine intelligence. The hybrid approach imposes several challenges, which we will discuss next.

Challenge 2: The trade-off between human- and machine-actionability

As becomes apparent from the description of the previous challenge, information that is understandable for humans is not necessarily understandable for machines, and vice versa. When creating structured paper descriptions, inherently the human readability decreases. After all, the description is specifically designed for machine consumption and not for human readability. While the previous example of the structured R0 estimate is still understandable for humans, structured descriptions quickly get too complex to understand, in particular for untrained users. This applies to consuming structured data, but even more so in creating such descriptions. Generally, they are created by Data Modeling experts, who are experienced with data modeling in general and with a certain research domain in particular. Naturally, one cannot expect arbitrary researchers to have data modeling skills. However, still, the data has to be consumed and partially generated by humans. From a machine-actionability perspective, it would be best if data is described consistently, reusing existing data structures (i.e., *ontologies*). This generally results in complex hierarchical graph structures. On the other hand, from a human perspective, modeling data would be more straightforward if a flat structure is used (e.g., in the form of key-value pairs). Hence, in general there is a trade-off between human readability and machine actionability. We address this challenge by providing user interfaces that hide the data complexity. For data consumption, this means interfaces that present data in a human-understandable way, while processing richly structured graph data in the background. We adopt a similar approach for data creation. Data entry is made simple for humans (by filling out forms), but complex graph structures are generated to support machine-actionability in the background.

Challenge 3: A research domain agnostic approach for scholarly knowledge graph creation

The knowledge graph should contain scholarly knowledge from various research domains and should not be tailored to a specific field. To find a vocabulary to describe arbitrary scholarly knowledge on a conceptual level is non-trivial. The more generic the terms are, the less informative, and thus the less value they offer in the knowledge graph. On the other hand, specific domain terms are only relevant within their respective context and are not applicable to other domains. The challenge relates to finding an approach that works for all disciplines while still providing value. The objective is not to find a single vocabulary that works for all domains but rather to focus on an generic approach. This domain agnostic approach has several implications from a user interface perspective and impacts the terminology used in the system. The term *knowledge graph* is relatively frequently used in Computer Science but a relatively unknown concept outside that domain. This means that the interface cannot make any assumptions from a terminology perspective. Several terms are commonly used when discussing knowledge graphs (triples, instances,

classes, ontologies, etc.). Although these concepts are required when creating a knowledge graph, these terms are not present in the user interface. To a large extend, this challenge relates to communicating concepts and finding relations to what an arbitrary researcher is familiar with. Finally, the user interfaces should focus on usability and should provide easy-to-use tools to create structured scholarly data.

Challenge 4: Ensure task simplicity for crowdsourcing and still generate valuable and high-quality data

To ensure the quality of the structured scholarly data, humans are involved in creating this data. As the knowledge graph targets scholarly information, primarily researchers will be using the data. To reach good coverage, even in selected research communities, a large number of researchers have to be included in this process. This takes place in a crowdsourced setting, where researchers can create structured scholarly knowledge collaboratively. To motivate researchers, the barrier to contribute should be as low as possible. This means, tasks have to be well defined, quick and easy to perform. When tasks are too complex or time-consuming, the willingness of researchers to contribute decreases. On the other hand, the task result should be valuable and machine-actionable knowledge. This challenge is addressed by allowing different levels of user contributions. A contribution can be as simple as fixing a textual error or annotating a sentence within a publication, or as complex as creating templates that specify how data should be modeled.

Challenge 5: Address the issues both retrospectively and prospectively

Challenge 1 describes the transformation from unstructured data to structured data. This transformation is necessary because the knowledge is already published in a traditional document-based form. This relates to *retrospectively* addressing the issues with scholarly communication, i.e., for published articles. It includes the information extraction from legacy articles, such as PDF documents. However, if the knowledge was made available in a machine-readable form in the first place, this knowledge transformation is not necessary. Mons said “Text mining? Why bury it first and then mine it again?” [12], criticizing the need for text mining tools to retrospectively extract knowledge, while this should be done at the publishing phase instead. Therefore, instead of merely focusing on structuring scholarly knowledge for existing documents, there should also be a focus on changing the methods and workflows for future publishing of new scholarly articles. This means that the issues are addressed *prospectively*, i.e., before the articles are published. Prospectively making changes to the knowledge-sharing methods imposes technical as well as social challenges. The social challenges include changes in behavior and workflows for researchers. Arguably, these social changes can be more challenging to accomplish than the technical aspects.

1.3 Research Questions

Based on the described challenges, we now formulate three research questions. The research questions are orthogonal to the challenges and are related to three main themes of this thesis, namely scholarly knowledge that is 1) machine-actionable, 2) comparable, and 3) crowdsourced.

RQ1: How to organize scholarly knowledge using a manually curated scholarly knowledge graph?

The first research question investigates how knowledge graphs can be leveraged to organize scholarly knowledge. In particular, it focuses on the manual human curation aspect. This includes the use of human labor to populate and validate knowledge within the graph. To answer this question, we develop graphical user interfaces to support researchers in consuming and creating scholarly knowledge. The designed interfaces and tools are deployed as a publicly available service called the Open Research Knowledge Graph (ORKG).

RQ2: How to generate machine-actionable and comparable overviews of related literature?

Research articles that address the same research topics are generally described using similar graph patterns. This research question focuses on the comparability of research addressing similar research problems. Such information can be represented in tabular overviews that highlight the similarities and differences between the articles. The tabular overviews can be used to list state-of-the-art research for a specific topic, to rank research based on their results (i.e., leaderboards), or to show how a research domain has evolved over time. We answer this research question by investigating how to represent these literature overviews in a machine-actionable way. The overviews are generated in a semi-automatic manner, leveraging structured data from the graph. The evaluation results indicate that the comparisons are able to represent the same information as tabular literature overviews found in traditional articles. Additionally, the evaluation showed that comparisons are providing literature overviews in machine-actionable form. Furthermore, methods are discussed on how to populate the literature comparisons based on existing survey article tables. As a use case, the presented methodology is used to import tables from over a hundred review articles. Finally, the comparisons are used as a basis to generate dynamic literature review articles.

RQ3: How to intertwine machine and human intelligence for populating and curating a scholarly knowledge graph?

To effectively populate the scholarly knowledge graph, we propose to intertwine human and machine intelligence to exploit the advantages of both approaches. This research question focuses on approaches where either humans are assisted by machines or machines by humans. To address this research question, we propose to use NLP tools to process scholarly knowledge and present the resulting data to humans who can decide whether the results are helpful or not. Specifically, we develop intelligent user interfaces where machine intelligence forms an integrated part of the interface. A user evaluation of this interface focuses on whether users appreciate the intelligent components, even if the recommendations are not always accurate. Results indicate that as long as the intelligent components provide dismissable suggestions, users do indeed appreciate the intelligent tools.

1.4 Thesis Overview

The thesis structure uses the overarching themes from the research questions: 1) machine-actionable, 2) comparable, and 3) crowdsourced scholarly knowledge. Now, we first discuss the contributions to each of the research questions. Afterwards, we list the publications that serve as the foundation for this thesis.

1.4.1 Contributions

Contributions for RQ1

- A scholarly knowledge graph called the Open Research Knowledge Graph (ORKG).
- A toolset that enables researchers to generate structured scholarly knowledge.

To organize scholarly knowledge with a manually curated knowledge graph, we introduce a scholarly infrastructure called the Open Research Knowledge Graph (ORKG). The contributions of this thesis are in particular related to the user interface and interaction side of the ORKG. The main contribution is the ORKG frontend which is available as an online service.[1] The frontend consists of different interfaces, which allow users with varying levels of expertise to interact with the ORKG. It includes the *Add paper wizard*, enabling users to add papers to the ORKG following a three-step wizard. Furthermore, a CSV import functionality is provided that allows users to import existing data. Imported data is automatically converted into an ORKG compatible graph format. A tabular contribution editor supports users in adding and editing multiple papers in parallel. This editor makes it easier to describe articles that address the same research problems and thus use the same

[1] `https://www.orkg.org`

graph structure to describe the data. The ORKG is evaluated with various user studies and is currently in active use by researchers. It forms the foundation of all other research contributions of this thesis. In most cases, the ORKG infrastructure is used to implement the proposed methodologies. Furthermore, the platform is used to validate various aspects of the proposed approaches.

Contributions for RQ2

- Comparisons – a methodology to generate machine-actionable literature reviews.
- A method to semi-automatically populate comparisons from existing survey tables.
- SmartReviews – a tool to author community-maintained semantic literature reviews in the form of living documents.

We present a methodology to generate dynamic literature comparisons. This methodology includes several steps that result in machine-actionable literature overviews. The methodology is used to implement comparisons within the ORKG. Comparisons form one of the key features of the ORKG, as it allows researchers to discover and compare existing information more easily. Comparisons can be assigned a DOI that forms a persistent link to the current state of the respective comparisons. This makes comparisons citable in research articles. Additionally, ORKG comparisons are leveraged to provide comprehensive literature overviews. Comparisons are community-maintained and can be created collaboratively. Another contribution is related to populating comparisons with data from existing literature review articles. We present a methodology to extract information from tables within the PDF version of these articles. The extracted data is used to populate ORKG comparisons via a semi-automatic workflow. Furthermore, we present an approach, called *SmartReviews*, to author review articles in the form of living documents. SmartReviews are implemented within the ORKG, and they include ORKG comparisons as a primary artefact.

Contributions for RQ3

- An crowdsourcing approach to annotate sentences within scholarly articles with discourse classes.
- A microtask approach to validate NLP extracted knowledge applied in a crowdsourced setting.

As previously described, we focus on intertwining human and machine intelligence. To this end, we integrated intelligent system components throughout the ORKG interface. We designed an interface that is specifically tailored towards a crowdsourced setting.

This interface is designed to be implemented within the paper submission process. Paper authors are requested to annotate their papers with discourse classes. During this process, authors are supported by intelligent components in various ways. This includes automatic highlighting of potentially interesting sentences within the article. Furthermore, selected sentences are supplemented with a list of potentially relevant annotation classes. The interface is evaluated with researchers. Evaluation results indicate that smart components enhance the annotation process and provide valuable suggestions for the participants. Furthermore, the results showed that paper authors are willing to annotate their papers during the submission process. This paper annotation interface is implemented within the ORKG. Annotated sentences and their discourse classes are stored within the graph. Finally, a microtask approach is introduced to validate scholarly knowledge statements generated by NLP tools. By decomposing the complex task of knowledge graph validation into smaller microtasks, we are able to include regular users into the validation process. Such users would normally merely consume content and not contribute to the knowledge graph itself. This enables processing and validating scholarly knowledge at scale.

1.4.2 List of Publications

Most of the work presented in this thesis has been published at conferences. Peer-reviewed publications relevant to this thesis are listed below in chronological order. Each thesis chapter is based on one or multiple publications. For the sake of clarity, we explicitly mention the respective publications in the introduction section of each chapter. A complete list of all work published during this thesis, including work not directly related to the thesis, can be found in Appendix A.

1. Jaradeh, M. Y., **Oelen, A.**, Prinz, M., Stocker, M., & Auer, S. (2019, August). *Open research knowledge graph: a system walkthrough.* In International Conference on Theory and Practice of Digital Libraries (pp. 348-351). Springer, Cham. (demo paper)

2. Jaradeh, M. Y., **Oelen, A.**, Farfar, K. E., Prinz, M., D'Souza, J., Kismihók, G., Stocker, M., & Auer, S. (2019, September). *Open research knowledge graph: next generation infrastructure for semantic scholarly knowledge.* In Proceedings of the 10th International Conference on Knowledge Capture (pp. 243-246). (short paper)

3. **Oelen, A.**, Jaradeh, M. Y., Farfar, K. E., Stocker, M., & Auer, S. (2019, November). *Comparing research contributions in a scholarly knowledge graph.* In CEUR Workshop Proceedings 2526 (Vol. 2526, pp. 21-26). (workshop paper)

4. **Oelen, A.**, Jaradeh, M. Y., Stocker, M., & Auer, S. (2020, August). *Generate FAIR Literature Surveys with Scholarly Knowledge Graphs.* In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020 (pp. 97-106). (full paper)

5. **Oelen, A.**, Stocker, M., & Auer, S. (2020, November). *Creating a Scholarly Knowledge Graph from Survey Article Tables.* In International Conference on Asian Digital Libraries (pp. 373-389). Springer, Cham. (full paper)

6. Auer, S., **Oelen, A.**, Haris, M., Stocker, M., D'Souza, J., Farfar, K. E., Vogt, L., Prinz, M., Wiens, V., & Jaradeh, M. Y. (2020, December). *Improving Access to Scientific Literature with Knowledge Graphs.* Bibliothek Forschung und Praxis, 44(3), 516-529. (journal paper)

7. **Oelen, A.**, Stocker, M., & Auer, S. (2021, April). *Crowdsourcing Scholarly Discourse Annotations.* In 26th International Conference on Intelligent User Interfaces (pp. 464-474). (full paper)

8. **Oelen, A.**, Stocker, M., & Auer, S. (2021, September). *SmartReviews: Towards Human- and Machine-actionable Survey Articles.* In International Conference on Theory and Practice of Digital Libraries. (pp. 181-186). Springer, Cham. (demo paper)

9. **Oelen, A.**, Jaradeh, M., Stocker, M., & Auer, S. (2021, Oktober) *Organizing Scholarly Knowledge leveraging Crowdsourcing, Expert Curation and Automated Techniques.* Book: Linking Knowledge: Linked Open Data for Knowledge Organization. (book chapter)

10. **Oelen, A.**, Stocker, M., & Auer, S. (2021, December). *Towards Human- and Machine-actionable Representation of Survey Articles.* In International Conference on Asian Digital Libraries. Springer, Cham. (short paper)

11. **Oelen, A.**, Stocker, M., & Auer, S. (2022, June). *TinyGenius: Intertwining Natural Language Processing with Microtask Crowdsourcing for Scholarly Knowledge Graph Creation.* In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2022. (late breaking paper)

1.5 Thesis Structure

The chapters within the thesis depend on each other. Specific chapters are used as a foundation for successive chapters. A summarized overview of the thesis chapters and their relations is depicted in Figure 1.3. The overview figure also highlights the three overarching themes of the thesis. The rest of the thesis is structured as follows: Chapter 2 introduces background concepts related to the thesis concepts. This includes an introduction to structured data, knowledge graphs, and the semantic web. Furthermore, the meaning of main-actionability and usefulness of FAIR data are discussed. Finally, different gradations of crowdsourcing are discussed. Chapter 3 discusses the related work. This includes

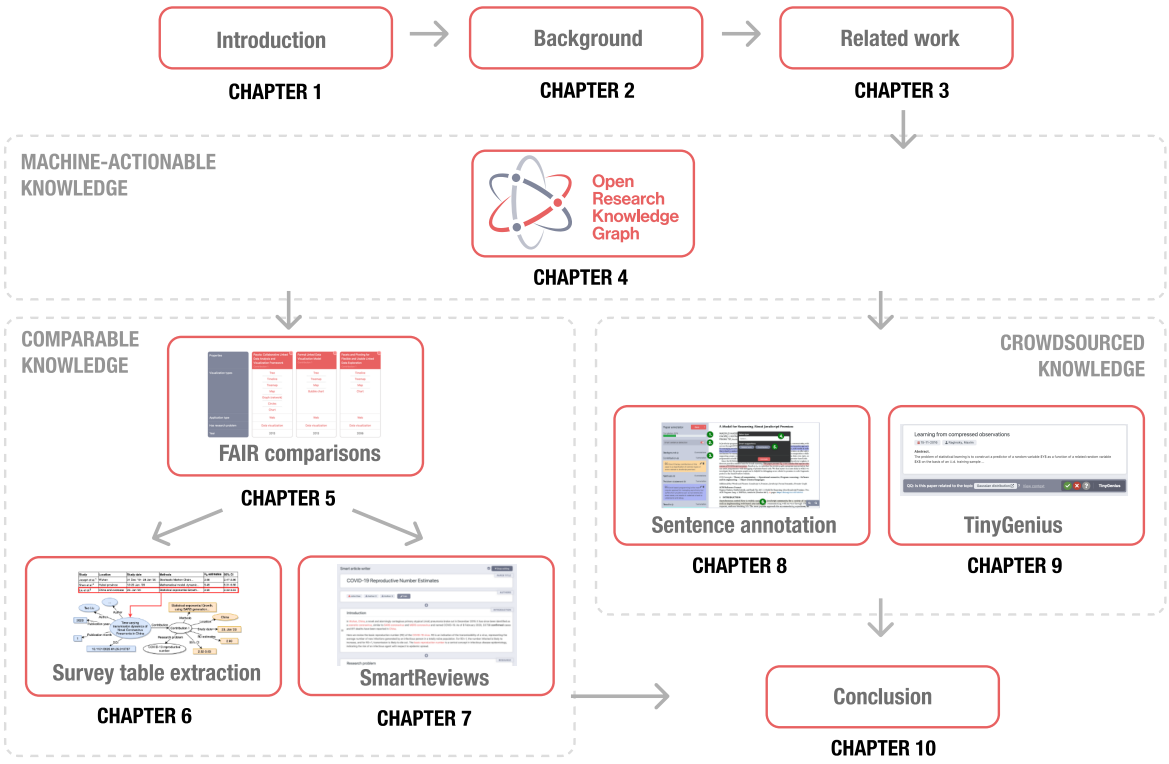

Figure 1.3: Overview of the contributions and chapters of this thesis. Arrows indicate relations between chapters.

work related to semantic scholarly knowledge and information extraction from existing documents. Chapter 4 introduces the Open Research Knowledge Graph (ORKG). We discuss the foundations of the ORKG infrastructure. This includes an explanation of the data model and the technical foundation of the service. Furthermore, we discuss the different user interfaces which can be used to create and curate structured scholarly knowledge. This includes the *Add paper wizard*, *Contribution editor*, and *CSV import* functionalities. Finally, the different actors and their interactions within the ORKG system are discussed. The chapter forms the foundations for the following chapters. Chapter 5 introduces the concept of *Comparisons*. We present a methodology that is used to generate comparisons. Furthermore, we explain how this methodology is used to implement the comparison concept within the ORKG. Finally, we evaluate the implementation to determine whether comparisons are able to replace existing methods of publishing literature overviews. Additionally, we empirically evaluate the performance of the implementation and evaluate the FAIR aspects of the methodology. Chapter 6 presents a methodology to semi-automatically generate the comparisons introduced in the previous chapter. This methodology relies on table extraction from PDF versions of review articles. We evaluate the approach through a use case where we extract data to generate several hundred comparisons. Chapter 7 introduces

the concept of *SmartReviews*, an approach to collaboratively author review articles in the form of living documents. This chapter discusses several significant weaknesses of the current review approach. The SmartReview approach addresses the weaknesses. Afterwards, the implementation within the ORKG is discussed. ORKG comparisons form the core of SmartReviews and provide a means to generate machine-actionable review articles. Chapter 8 presents an approach to create structured scholarly knowledge by integrating a paper annotation interface within the paper submission process. The method leverages artificial intelligence supported components to improve the user experience during this task. A user study is conducted to determine the usability of the system and the attitudes of researchers towards the task in general. Chapter 9 presents an approach to validate automatically extracted scholarly statements via crowdsourcing. The statements are extracted with a set of five NLP tools and validation happens via context-free microtasks. With microtasks, we lower the barrier for users to become content contributors without the need of specific domain knowledge. Chapter 10 concludes the thesis and answers the research questions. Finally, future research work directions are discussed in this chapter.

CHAPTER 2

Background

In this chapter, we discuss the background concepts and technologies used in this thesis. Semantic web technologies are a central topic in this thesis. We first briefly introduce the history of the web in general, where we relate the development of the web to the (lack of) development for scholarly communication. Afterwards, we discuss semantic web technologies in more detail.

2.1 Introduction to the Web

Document-centric scholarly article dissemination is one of the key issues scholarly communication currently faces. The first implementation of the World Wide Web 1.0 (or simply the Web 1.0) had a similar document-oriented setup. The first version of the web consists of interlinked documents forming a spider-web-like structure of hyperlinks, hence the naming of the web. This version of the web was mainly targeted at content delivery purposes, and web pages are therefore largely static documents [13]. Documents within the web are uniquely identified via Uniform Resource Identifiers (URIs). The web is accessible over the internet, a global system of interconnected networks. Web pages are transferred over the Hypertext Transfer Protocol (HTTP). Web pages are hypertext documents (i.e., documents that support linking of words to other documents) formatted in Hypertext Markup Language (HTML). Web pages are sent from a computer, called the web server, to another computer, called the client. On the client, HTML web pages are displayed in a web browser. HTML documents can be accompanied by Cascading Style Sheets (CSS) to provide specific visual styling to documents. Furthermore, HTML documents can embed the scripting language JavaScript to provide interactive behavior of web pages. The interactive behavior results from dynamic changes to the Document Object Model (DOM) rendered by the web browser. The DOM provides a hierarchical interface to HTML documents, where each HTML element is a node in the hierarchical tree.

Table 2.1: Overview of the history of the web, comparing different characteristics of each version. Based on work from Choudhury [13].

Web 1.0	**Web 2.0**	**Web 3.0**
Hypertext web	Social web	Semantic web
1996-2004	2004-2016	2016+
Read-only web	Read-write web	Executable web
Document-based	Interactive document-based	Data-based
Human-oriented	Human-oriented	Machine-oriented
Static documents	Dynamic documents	Dynamic data

The successive versions of the web are not strictly defined by technology changes, by mainly by usage changes. The previously described technology stack of the first version of the web also applies to the successive versions. The second generation of the web, Web 2.0, is a term introduced by O'Reilly [14]. In contrast to the read-only Web 1.0, it describes a read-write web where users can interact with the website. This resulted in online social communities and collaborative online knowledge bases such as Wikipedia [15]. The third generation of the web, Web 3.0, can be considered the *executable web*[13]. A more commonly used term among academics for this version of the web is the Semantic Web. This term was introduced by Berners-Lee [16]. In contrast to the Web 1.0, which is a web of *documents*, the Web 3.0 is a web of *data*. The former is specifically designed for human consumption, while the latter is focused on machine consumption. This transition from human to machine-oriented knowledge representation is what we try to achieve for scholarly knowledge as well. Semantic web technologies can be used in various use cases and provide a means to share and reuse data in a predefined format. For example, search engines can leverage semantic web technologies to provide users with *active help* to fulfill their information need. Regular document-based search engines mainly provide *passive help*, in which they generally provide a list of potentially relevant documents where the answer can be found. Table 2.1 summarizes the different versions of the web.

2.2 Semantic Web Technologies

The semantic web stack is commonly illustrated as a layered cake, see Figure 2.1. The illustration indicates which technologies are used to create the semantic web and how these technologies depend on each other. We will now use this illustration to explain the semantic web in more detail.

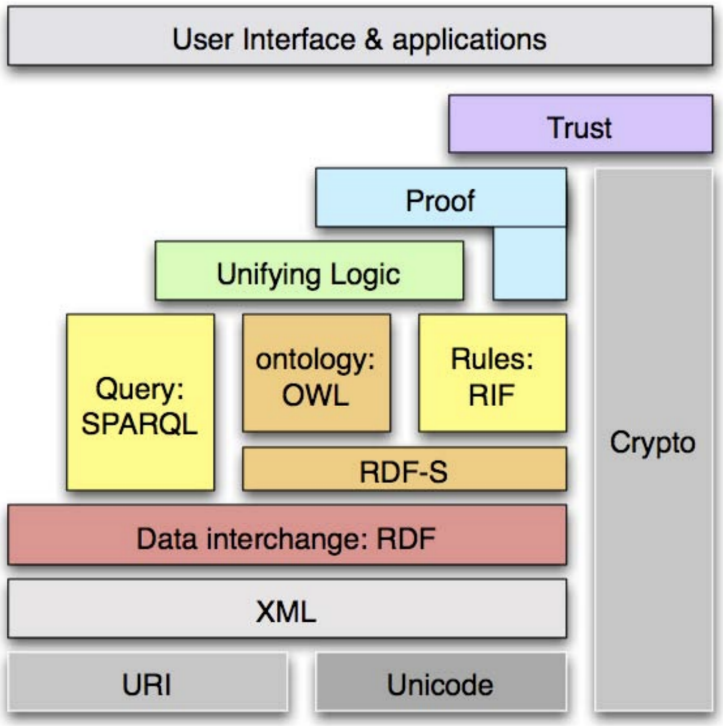

Figure 2.1: The semantic web *layer cake* [17].

2.2.1 Resource Identifiers

At the foundation, the semantic web uses URIs to uniquely identify resources. In contrast to the Web 1.0, which uses URIs to identify documents, the semantic web uses URIs to identify data. URIs containing information about the location where the resource can be found are called Uniform Resource Locators (URLs). URLs form a subset of URIs. Per specification, URIs may only contain characters from the American Standard Code for Information Interchange (ASCII) character encoding. To make URIs better accessible for people unfamiliar with the Latin alphabet, Internationalized Resource Identifier (IRIs) are introduced. IRIs extend the default character set from URIs by supporting Universal Coded Character Set (UCS, Unicode) characters. Among others, this provides support for Chinese and Japanese characters. IRIs form a superset of URIs, as they provide an extension to the URI specification.

2.2.2 Resource Description Framework (RDF)

The RDF model is used for data interchange. It provides a model to describe web resources, and it is standardized by the World Wide Web Consortium (W3C). At the core, the RDF

model specifies resources and the relationships between them. The specification defines data in a triple format, consisting of three entities. RDF reuses terminology from linguistics to describe the individual entities of the triple, namely a *subject*, *predicate*, and *object*. Additionally, data triples are called *statements* in RDF. Therefore, to create data, one makes a statement about a subject and how it relates (via the predicate) to the object. For example, it is possible to represent the following natural language sentence in RDF *The book is owned by Bob*. The RDF statement looks as follows: Subject: *the book*, predicate: *is owned by* and object *Bob*. The resources used in statements all have URIs. The URIs are those resources that can be used to make additional statements about the same resource. Apart from resources, statements can also contain literals and blank nodes. A literal in RDF cannot be identified via URIs. Literals have data types (i.e., data type resources with URIs). Furthermore, it is possible to specify the language of the text within a literal. Blank nodes can be considered resources without a URI, sometimes referred to as anonymous resources. Generally, they are used to group related information to prevent ambiguity between different sets of statements. Literals are used to describe values such as numbers, dates, or natural text. As per the RDF specification, the subject position of statements contains either resources or blank nodes. The predicate position can only contain resources. The object position can contain resources, literals, and blank nodes. In Figure 2.2, two RDF statements are depicted. Statement 1 describes the location of a particular study. It links the study resource to the Singapore resource. Statement 2 describes a result of the study, namely a basic reproduction number. Here, the same study resource is used to create a literal statement that links the basic reproduction number of 1.45 to the study. The literal has a data type of *xsd:decimal*. As can be seen, both the predicate and the object have external URIs. External URIs can also be used in the subject position, giving the possibility to make statements about external resources. The ability to make use of external resources highlights one of the cornerstones of the semantic web. No authority needs to give consent or validate the correctness of the created statements. This relates to the RDF concept “Anybody to say Anything about Any topic” (the AAA slogan) [18]. Because of this open character of the semantic web, the open-world assumption is adopted (absence does not imply falsehood) [19]. RDF prescribes how data should be represented (i.e., as triples), without enforcing specific data models. This means that data is can be machine-readable, even if the data models differ. To further enhance machine readability, more granular data models can be used to describe RDF data. For this purpose, ontologies (or vocabularies) are used. An ontology describes a model which consists of concepts and the relations between them. RDF provides a basic ontology to describe instance data. Instance statements are referred to as ABox statements. RDF Schema (RDFS) is an ontology specifically designed to create ontologies, resulting in TBox statements. It allows for defining classes, class hierarchies, and properties descriptions, such as the domain and range. The Web Ontology Language (OWL) extends both the RDF and RDFS ontologies to provide a richer and more expressive vocabulary. Furthermore, OWL provides the ability to define restrictions, such as the carnality for properties. OWL has become the standard for RDF ontology design.

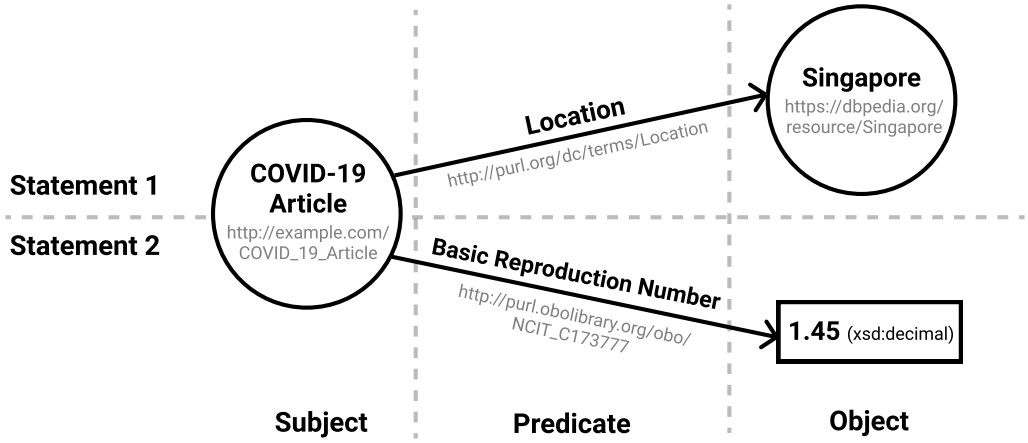

Figure 2.2: Illustration of a simplified version of an RDF graph. Two statements are depicted related to a resource that represents a COVID-19 publication. Both the study location and result (i.e., Basic Reproduction Number) are displayed. URIs of each resource are displayed to show how existing ontologies and data are reused to generate the statements.

The relations between the resources form a directed labeled graph. Directed because there is an asymmetric relation from the subject to the object. From our example, the reverse statement *Bob is owned by the book* is not valid. Furthermore, the graph is labeled by the predicates used to describe the relation. The graphs that result from describing data in RDF are generally referred to as Knowledge Graphs. The term knowledge graph is not strictly defined, and there are several popular definitions to describe the meaning [20]. For example, the Journal of Web Semantics defines knowledge graphs as "Knowledge graphs are large networks of entities, their semantic types, properties, and relationships between entities" [21]. The definition of Färber et al. explicitly mentions RDF to define knowledge graphs: "We define a Knowledge Graph as an RDF graph..." [22]. In this thesis, we use the term knowledge graph as defined by the Journal of Web Semantics [21]. This means a knowledge graph is defined by the network, the structure, and the semantics incorporated in the network and not by the model used to represent the data (e.g., RDF).

2.2.3 RDF Serialization Formats

Now, we discuss the document types (i.e., formats) in which data is communicated. Traditionally, the markup language Extensible Markup Language (XML) is used as data serialization format. More specifically, the RDF/XML syntax is used to represent RDF. Over time, more RDF serialization formats are introduced, for example, to provide better human readability (e.g., Notation3 and Turtle) or improved compatibility with other formats (e.g., JSON-LD). The different RDF serializations are not successors of previous formats; therefore, most formats are still widely used and chosen based on the use case. To highlight the differences

```
<?xml version="1.0"?>
<rdf:RDF xmlns:dc="http://purl.org/dc/terms/">
    <rdf:Description rdf:about="ex:COVID_19_Article">
        <dc:location rdf:resource="dbpedia:Singapore"/>
        <obo:NCIT_C173777 rdf:datatype="xsd:integer">1.45</obo:NCIT_C173777>
    </rdf:Description>
</rdf:RDF>
```

Serialization 2.1: RDF/XML serialization of an RDF graph.

```
@prefix dc: <http://purl.org/dc/terms/> .

<ex:COVID_19_Article>
    dc:location       <dbpedia:Singapore> ;
    obo:NCIT_C173777  1.45 .
```

Serialization 2.2: Notation3 and Turtle serialization of RDF graph

between different RDF serializations, we now serialize the same RDF graph from Figure 2.2 into various formats. Namespace prefixes are used to abbreviate data and increase the human readability of data. A namespace has a prefix and a corresponding URI. For brevity reasons, we only list a single prefix in the RDF serialization examples instead of the full list. The following list contains all prefixes that are used in the examples:

- **rdf:** http://www.w3.org/1999/02/22-rdf-syntax-ns#
- **xsd:** http://www.w3.org/2001/XMLSchema#
- **dc:** http://purl.org/dc/terms/
- **dbpedia:** http://dbpedia.org/resource/
- **obo:** http://purl.obolibrary.org/obo/
- **ex:** http://example.com/

The RDF/XML serialization is listed in Serialization 2.1. This was the first official W3C standard for RDF data serialization. The format is relatively hard to read for humans, as it can be cumbersome to identify the individual subjects, predicates, and objects. The format is mainly used as legacy serialization.

The Notation3 (N3) serialization is listed in Serialization 2.2. This is a non-XML format that presents a more human-readable way of representing RDF [23]. Compared to RDF/XML, data is more clearly identifiable as triples, making it easier for humans to understand. In addition, several features are included to abbreviate the syntax, such as the ability to specific objects without the need of repeating the subjects and predicates. Furthermore, prefixes

```
<http://example.com/COVID_19_Article> <http://purl.org/dc/terms/location> <http://
    dbpedia.org/resource/Singapore> .
<http://example.com/COVID_19_Article> <http://purl.obolibrary.org/obo/NCIT_C173777>
     "1.45"^^<http://www.w3.org/2001/XMLSchema#integer> .
```

Serialization 2.3: N-Triples serialization of RDF graph

```
[
    {
        "@id": "ex:COVID_19_Article",
        "http://purl.org/dc/terms/location": [{ "@id": "dbpedia:Singapore" }],
        "http://purl.obolibrary.org/obo/NCIT_C173777": [
            { "@value": "1.45", "@type": "xsd:integer" }
        ]
    }
]
```

Serialization 2.4: JSON-LD serialization of RDF graph

are supported to shorten the syntax more. A subset of N3 is Terse RDF Triple Language (Turtle). Turtle has fewer features than N3, but Turtle's expressive power is sufficient for most use cases. Therefore, Turtle is a frequently used RDF serialization format.

The N-Triple is displayed in Serialization 2.3. N-triples form a subset of Turtle. As seen in the example, prefixes or syntax to prevent data repetition are not supported. Data is simply listed as triples, where the full URI represents each element. This makes the serialization more lengthy compared to Turtle. Due to the lack of features, N-Triples are easier to parse as they only use a simple syntax.

The JSON Linked Data (JSON-LD) format is displayed in Serialization 2.4. This format is designed to integrate linked data components in the popular JSON format. It is frequently used by search engines when parsing website content. Websites can publish supplementary JSON-LD to enable search engines to more effectively extract data from the web pages.

Finally, RDF in Attributes (RDFa) is displayed in Serialization 2.5. RDFa is designed to integrate RDF in HTML pages [24]. The listed example shows an HTML snippet that embeds RDFa notation as attributes to the HTML tags. When a browser renders the HTML page, the RDFa annotations are not visible for users. However, machines can parse the annotations and extract structured data from the annotated HTML tags. RDFa can be relatively easily integrated into existing HTML pages to semantically enrich the documents.

2.2.4 Query RDF with SPARQL

RDF triples are generally stored in triple stores. A triple store is a database superficially designed to handle data in triple format (e.g., in RDF). While other databases can store RDF, such as a text file or a relational database, triple stores are optimized to process

```
<div resource="ex:COVID_19_Article" prefix="dc: http://purl.org/dc/terms/">
  This study is conducted in
  <span property="dc:location" resource="dbpedia:Singapore">Singapore</span>.

  We found a Basic Reproduction Estimate of
  <span property="obo:NCIT_C173777" datatype="xsd:decimal">1.45</span>
</div>
```

Serialization 2.5: RDFa serialization of RDF graph embedded in an HTML page. The parsed HTML is displayed as "*This study is conducted in Singapore. We found a Basic Reproduction Estimate of 1.45*".

```
PREFIX db: <http://purl.org/dc/terms/>
SELECT ?article ?r0
WHERE {
  ?article  dc:location       dbpedia:Singapore;
            obo:NCIT_C173777  ?r0.
}
LIMIT 100
```

Query 2.1: SPARQL query to select a maximum of 100 articles that present basic reproduction numbers (R0) for Singapore.

```
PREFIX db: <http://purl.org/dc/terms/>
INSERT {
  ?article  obo:NCIT_C173777  "2.45".
} WHERE {
  ?article  dc:location       dbpedia:China.
}
```

Query 2.2: SPARQL query to insert data for research studies conducted in China.

triples efficiently. Some popular triples stores include Virtuoso[1] and Jena-TDB[2] [25]. Triple stores that support RDF generally provide data access via the Protocol and RDF Query Language (SPARQL). SPARQL is officially recommended by the W3C to query RDF data. To retrieve information using SPARQL queries, graph pattern matching is applied. The syntax of the triple patterns is similar to the Turtle triple syntax as listed in Serialization 2.2. The following query types are available for information retrieval: SELECT, CONSTRUCT, ASK and DESCRIBE. The SELECT query selects data and returns it in tabular format. CONSTRUCT creates an RDF graph from the results. ASK returns true or false for a given pattern. Finally, DESCRIBE returns a description of resources, as specified by the query service.

[1] https://virtuoso.openlinksw.com/

[2] https://jena.apache.org/documentation/tdb/

An example of a SPARQL select query is displayed in Query 2.1. Two variables are selected. In the where clause, the graph pattern is displayed, which is used to find matching triples. In the graph pattern, the two variables are used to indicate wildcards for triple matching. The result returns a table with two columns, listing the article (?article) and the respective basic reproduction number (?r0). By default, all patterns specified in the where clause must match (conjunction). To support disjunct patterns, the UNION keyword is used. Finally, a limit is set to only return 100 articles at maximum. SPARQL also supports insertion and deletion of data, using the INSERT and DELETE query types, respectively. An example of an insert query is displayed in Query 2.2. As can be seen, the same query pattern matching can be used for insertion queries as well. This also applies to deletion queries, which deletes triples matching the provided pattern.

2.3 Linked Data

When data is described with semantics, the data can become knowledge that is part of a knowledge graph. Data has a part in the Data, Information, Knowledge, Wisdom (DIKW) pyramid. There is raw data at the bottom of the pyramid, for example, a set of COVID-19 infection measurements. The data in itself does not provide any information about the contagiousness of the virus. However, when the data is processed and plotted, the data provides information about the basic reproduction number. In turn, insights are gained, and lessons are learned from the information, thus providing knowledge. Finally, knowledge can become wisdom if the knowledge is applied intuitively to answer related questions. For example, in estimating the basic reproduction number of other viruses. This brief introduction to the DIKW pyramid highlights the importance of the incorporation of semantics in data. Otherwise, the data has limited impact and cannot be used to its full potential.

In Figure 2.3, we list four gradations of data with regards to the machine-actionability of the data. Machine-actionability is the ability of machines to interpret data without human intervention [26]. Occasionally, the term machine-readability is used interchangeably. However, we make a distinction between those terms. If a machine is able to read data (machine-readability), it does not entail that the machine is able to understand and further process the data (machine-actionability). Machine-readability is considered the first step towards machine-actionability. Regarding the listed gradations of data in Figure 2.3, machines cannot easily process Unstructured data. It is estimated that approximately 80% of the business data is unstructured [27]. Unstructured data includes natural language text, images, and videos. Therefore, this type of data gets significant attention from research communities. Machine learning techniques are used to extract information from these unstructured data sources. For example, Natural Language Processing (NLP) is used to extract information from natural text. Furthermore, machine learning based image processing is used to perform information retrieval and understanding from images. Optical

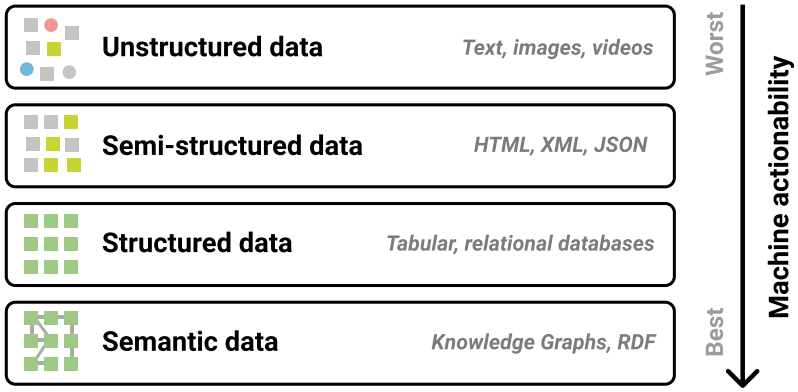

Figure 2.3: Different gradations of structured data. Unstructured data is the least machine-actionable. Semantic data is the most machine-actionable.

Character Recognition (OCR) is used to detect text in bitmap images to improve the machine readability of the data. Next, there is semi-structured data. This type of data consists of loosely structured data. For example, HTML, XML, and JSON documents. While machines can more easily read this data, it is still subject to machine interpretation, which remains a challenging task. In the literature, RDF is sometimes considered as semi-structured data as well. However, we make the distinction of data gradations based on machine-actionability. Therefore, we exclude semantically described RDF data from this category. Structured data consist of homogeneous tabular data, for example, found in relational databases. Compared to semi-structured data, structured data is easier to read for machines. However, still, there are no semantics encoded in the data. Meaning, that human intervention is generally required to process and reuse the data. Finally, there is semantic data. This data is generally described via graph structures and can be modeled via RDF. Because semantics are encoded in this data, it provides the most machine-actionable data from the listed data types.

Semantic data is linked data. The data links to other data sources and links to external vocabularies to incorporate semantics in the data. Berners-Lee formulated four design principles for linked data [28] in 2006:

1. Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information using the standards (RDF*, SPARQL).
4. Include links to other URIs so that they can discover more things.

The first principle related to one of the key pillars of the semantic web, resources should have URIs. This provides the possibility to uniquely identify, and link to, self-contained

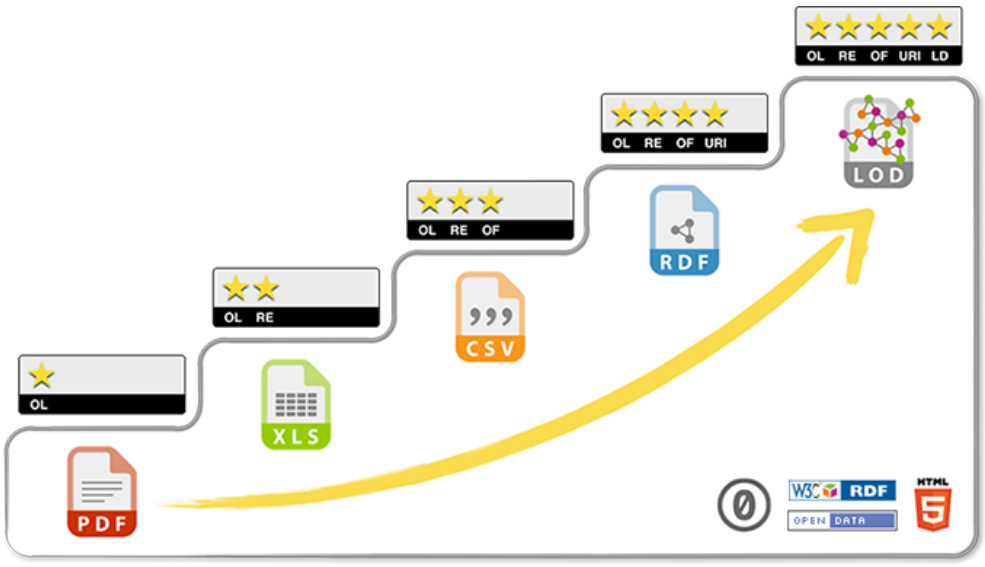

Figure 2.4: The Linked Open Data star rating system. Source: `https://5stardata.info/en/`.

pieces of data. The second principle describes that the URIs should be resolvable via HTTP. As mentioned in the proposal, there is a tendency to deviate from HTTP and to use alternative URI schemes [28]. This is discouraged as HTTP resolvable URIs provide a well-known mechanism to resolve them. The third principle relates to the format of the described resources. To enable reuse, standards such as RDF and SPARQL should be used. Finally, the fourth principle encourages linking to external data and ontologies. This creates an interconnected web of data, providing opportunities to discover new data.

Related to, but not to be confused with, the four principles are the Linked Open Data (LOD) stars. The star scoring system is proposed by Berners-Lee in 2010 [28]. It provides a scoring mechanism to evaluate publicly available datasets on how well they follow linked data principles. Originally, the system is designed to encourage governments to publish linked open data. The star rating scheme is depicted in Figure 2.4. As a minimal requirement, data has to be published with an Open License (OL). Otherwise, it cannot be considered as open data. This implies that data can be used and reused without costs. It does not impose any requirements for the format of the published data. Formats that are hard to process for machines (such as PDF) are acceptable to obtain a single star. The next star is given when data is available in machine-readable formats. This includes proprietary formats such as XLSX (created by Microsoft). Three stars are provided when non-proprietary formats are used to represent the machine-readable data, for example, CSV instead of XLSX. Four stars are given when data is assigned with URIs and when standards, such as RDF and SPARQL, are used. Finally, five stars can be obtained when data links to external data. Such datasets can be visualized as part of the Linked Open Data Cloud (LOD cloud). This is a

set of interlinked open datasets, forming a high interconnected graph or cloud. The LOD Cloud contains several highly interlinked datasets. For example, DBPedia [29] forms an interlinking hub between various other LOD cloud datasets.

2.4 FAIR data

To further improve the machine-actionability of data, Wilkinson et al. introduced the FAIR data principles in 2016 [26]. The FAIR acronym stands for Findable, Accessible, Interoperable, and Reusable. The objective of the FAIR data principles is to encourage machine-actionability and therefore enable the reuse of data. Each of the four principles is described using several sub-principles. The principles are summarized in Figure 2.5. The Linked Open Data (LOD) star rating system relates to the FAIR data principles. Both have the goal of improving data reuse. Some of the LOD star requirements can be mapped directly to the FAIR principles. Hasnain and Rebholz-Schuhmann researched the relation between the LOD stars and FAIR data principles [30]. However, the LOD star focuses explicitly on open data, while the FAIR data principles can also be applied with more restricted licenses. On the other hand, FAIR data makes statements about both data and the corresponding metadata. We now discuss the FAIR principles in more detail, and we highlight the relation to the LOD star schema, as described in [30].

2.4.1 FAIR Data Principles

Findable relates to the ability for machines to find both the data and metadata. This supports the automatic discovery of datasets by machines.

- *F1. (Meta)data are assigned a globally unique and persistent identifier.* This principle describes two conditions for data identifiers. They have to be unique and persistent. Examples of services that provide such identifiers are Digital Object Identifiers (DOIs), frequently used for scholarly articles and datasets. Furthermore, there is the Open Researcher and Contributor ID (ORCIDs), used to identify researchers. The LOD star scheme requires the use of persistent URIs from four stars and higher.

- *F2. Data are described with rich metadata (defined by R1 below).* Metadata should describe the contents of the data, the quality, and other relevant characteristics. This helps to make the interpretation of the data more straightforward, without the need for manual evaluation. Also, this encourages data reuse beyond the intended use-cases. There is no explicit mention of this in the LOD star scheme, however, the availability of metadata requires a machine-readable format (which is necessary from two stars and higher).

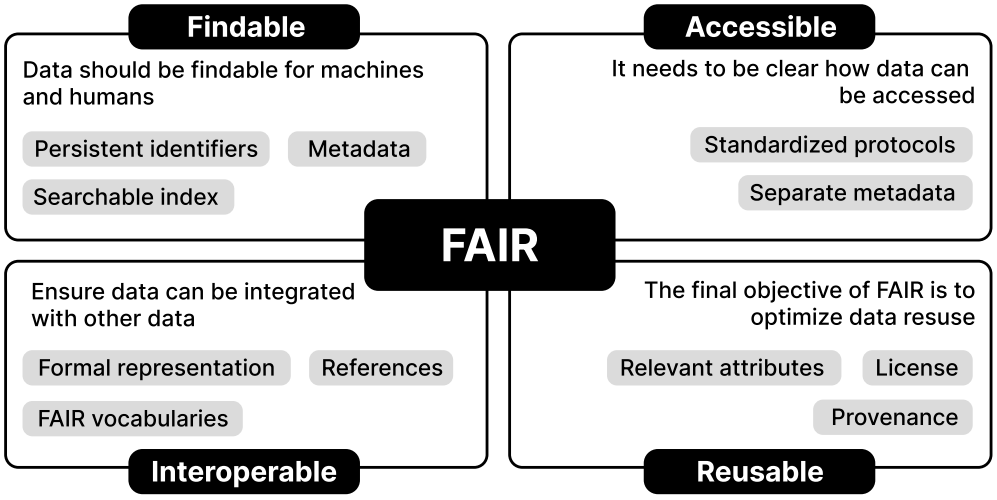

Figure 2.5: The FAIR data principles [26] summarized. Per principle the key sub-principles are displayed.

- *F3. Metadata clearly and explicitly include the identifier of the data they describe.* This simply requires an explicit link from the metadata to the actual dataset by means of the identifier from F1. As this also relies on URIs, this relates to four stars from the LOD star scheme.

- *F4. (Meta)data are registered or indexed in a searchable resource.* Without indexing the resources, the data remains unfindable, even if the previous principles are applied correctly. Therefore, the data should be registered in accessible search services. This relates to the first star of the LOD scheme, as it requires data to be accessible over the web.

Accessible relates to data access. It should be described how data can be accessed. This includes a description of the authentication and authorization processes when required by the service.

- *A1. (Meta)data are retrievable by their identifier using a standardized communications protocol.* This includes the frequently used HTTP protocol, as used for the web. Other protocols include the File Transfer Protocol (FTP), frequently used for file transfer, or Simple Mail Transfer Protocol (SMTP), used for email. As this relates to identifiers again, this is required from four LOD stars or higher.

- *A1.1 The protocol is open, free, and universally implementable.* The sub-principle prohibits the use of proprietary tools and protocols for data access. Also, this principle relates to four LOD stars because of the URIs.

- *A1.2 The protocol allows for an authentication and authorization procedure, where necessary.* This enables machines to automatically understand the requirements for authentication and authorization and are therefore able to autonomously use the data. This relates to a single LOD star, as it requires open access. However, FAIR principles do not relate to the openness of the data itself, as private data can also be FAIR. Therefore, one can argue that this principle is not part of the LOD stars.
- *A2. Metadata are accessible, even when the data are no longer available.* Data can get lost over time, especially large datasets that are relatively expensive to store. Therefore, this principle explains that the metadata of such datasets should remain available, also when the data itself is not anymore. This makes it easier to trace back deleted datasets and saves time in the search process.

  Interoperable relates to the ability to integrate data into other datasets and the possibility to be interoperable with different systems and workflows.

- *I1. (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.* Standardized data models should be used for data representation. This includes RDF, for example, using ontologies such as Dublin Core [31] to describe the (meta)data. This reduces the need for manual interpretation. It relates to three LOD stars, as a non-proprietary format should be used to represent the data.
- *I2. (Meta)data use vocabularies that follow FAIR principles.* The used ontologies should be well documented and identifiable via unique and persistent identifiers. This relates to five LOD stars, as semantically described data is needed.
- *I3. (Meta)data include qualified references to other (meta)data.* The references (i.e., links) to other datasets form a knowledge graph of interlinked data. This supports building upon existing datasets and reusing specific data entries from other datasets. As this also relates to semantically linked data, it corresponds to five LOD stars.

Reusable relates to optimizing the data for reuse by others.

- *R1. (Meta)data are richly described with a plurality of accurate and relevant attributes.* This principle is related to F2 but goes further than data discovery. It should provide the context under which the data was generated so that it is possible to decide whether the data is useful for a specific use case. Data includes the description of processes used to make measurements and experimental setups. This relates to three stars from the LOD schema, as it requires structured data.
- *R1.1. (Meta)data are released with a clear and accessible data usage license.* It should be clearly described which data license is attached to the data. Related, the LOD scheme requires an open license for the first star. However, as mentioned previously, for the FAIR principles, such an open license is not required.

- *R1.2. (Meta)data are associated with detailed provenance.* This includes provenance data such as the creator or collector, when it was created, and how it has been processed. This relates to five stars from the LOD scheme, as it requires contextual data.
- *R1.3. (Meta)data meet domain-relevant community standards.* When available, community standards should be used to represent data. In case the representation deviates from the standard, the reason should be described in the metadata. Such standards are also part of semantic descriptions and therefore related to five stars from the LOD scheme.

2.4.2 FAIR Research Data Initiatives

The FAIR data principles are not prescriptive [32], i.e., the principles outline what should be done but not how to do this. Therefore, there are several initiatives to incorporate the FAIR principals into more concrete workflows. In this section, we discuss two of such initiates in more detail. Firstly, we introduce the FAIR Digital Objects concept, based on the existing definition of Digital Objects. Secondly, we highlight RO-Crate, a lightweight approach to include metadata with research data.

FAIR Digital Objects

The concept of Digital Objects was described in 1995 (reprinted in 2006) by Kahn and Wilensky [33]. In this work, they outlined the key elements required for data management. The data itself is defined as a sequence of bits. The digital object is referenced by persistent identifiers and described by metadata. This describes the key components needed to make data abstractions, which provides a means to distinguish between the metadata and the data itself. Schwardmann highlights two aspects of data abstraction, encapsulation, and virtualization [34]. By encapsulating data, data is hidden that is not needed for a specific task. For example, metadata can be accessed without accessing the data itself. Virtualization enables linking between data and metadata. This enables the separate storage of data and the respective metadata.

The Research Data Alliance (RDA)[3] is a research organization that aims to improve sharing of research data by bridging both social and technical gaps. Among the founders are the European Commission and the United States National Science Foundation. RDA supports implementations of the FAIR data principles and uses them to develop data management strategies further. For example, a FAIR data maturity model is designed to access the FAIRness of research data [35]. Recently, the RDA developed the FAIR Digital Object Framework (FDOF), extending the original definition of digital objects [36]. The FDOF framework is currently under active development. This framework relates digital objects with the FAIR data principles. The two concepts are related as they both try to

[3] `https://www.rd-alliance.org/`

```json
{
  "@context": "https://w3id.org/ro/crate/1.1/context",
  "@graph": [
    {
      "@type": "CreativeWork",
      "@id": "ro-crate-metadata.json",
      "conformsTo": { "@id": "https://w3id.org/ro/crate/1.1" },
      "about": { "@id": "./" }
    },
    {
      "@id": "./",
      "identifier": "https://doi.org/10.5281/zenodo.5102832",
      "@type": "Dataset",
      "datePublished": "2021",
      "name": "Auxiliary evaluation data of SmartReviews",
      "description": "Contains two files related to the evaluation...",
      "license": { "@id": "https://creativecommons.org/licenses/by/4.0/" }
    }
  ]
}
```

Snippet 2.1: Simplified example of a *ro-crate-metadata.json* file for a dataset containing evaluation data.

improve data management. The FDOF defines the concept of FAIR Digital Objects (FDOs). FDOs are machine-actionable information units with a resolvable and persistent identifier, described by metadata and classified via the FDOF typing system. The metadata used to convey an FDO is an FDO on its own. An essential aspect of FDOs is that they need to have an information value. This excludes several types of atomic data to be FDOs on their own, but composed together, they can form an FDO.

RO-Crate

Research Object Crate (RO-Crate) [37] provides a lightweight approach to include metadata with their respective research data. Although the specification does not explicitly mention FAIR compliance, research data with an RO-Crate metadata specification increases the machine-actionability of the data. The specification includes the requirement of having machine-actionable metadata, identifiers, and the use of web protocols, contributing to the FAIRness of the data. RO-Crate aims to provide an approach for researchers from a variety of backgrounds. The RO-Crate specification is based on the schema.org[4] vocabulary. Schema.org provides a relatively easy-to-use ontology covering a broad range of topics, contributing to its popularity [38]. Therefore, the decision to use schema.org is beneficial for the adoption of RO-Crate. Furthermore, RO-Crate uses JSON-LD for metadata descriptions,

[4] `https://schema.org`

another well-established standard for linked data. Per the specification, the R0-Create metadata file *ro-crate-metadata.json* must be placed in the root directory of the dataset. Furthermore, optionally an HTML file of the metadata is included. This file represents the RO-Crate Website and represents a human-readable format of the metadata.

In Snippet 2.1, an example of a RO-Crate metadata specification in JSON-LD is displayed. In addition to boilerplate code to indicate the use of the Crate specification, it contains structured information about the data itself. This includes the identified (a DOI in this case), the type, publication year, name, description, and license. For brevity reasons, additional license information is omitted. More complex metadata specifications can also include paths for specific files within the dataset. These files can have their own name, description, license, and sub-parts.

CHAPTER 3

Related Work

The work presented in this thesis relates to, and builds upon, existing work, which we present in this chapter. First, we discuss existing literature related to semantic scholarly knowledge. This includes an overview of the FAIR data principles specifically focused on scholarly knowledge. Second, we discuss the role of literature reviews to organize scholarly articles. Last, information extraction methods are highlighted, specifically related to natural text extraction from PDF articles.

3.1 Semantic Scholarly Knowledge

Many of the applications to improve scholarly communication rely on the knowledge being represented in a structured way. Knowledge graphs can be employed to represent scientific contributions semantically, render scholarly knowledge more machine actionable. Prominent examples of openly available knowledge graphs include DBpedia [29], YAGO [39], and Wikidata [40]. With projects such as Semantic Scholar [41] and Microsoft Academic Graph [42], knowledge graphs are gaining popularity in the scholarly domain to structure scholarly knowledge. These graphs mainly capture metadata about research articles and do not describe the content of reported research work, including research contributions [43].

There is substantial related work on representing scholarly knowledge in structured form [44]. Building on the work of numerous philosophers of science, Hars [45] proposed a comprehensive scientific knowledge model that includes concepts such as theory, methodology, and statement. More recently, ontologies were engineered to describe different aspects of the scholarly communication process. Semantic Publishing and Referencing (SPAR)[1] is a collection of ontologies that can be used to describe scholarly publishing and referencing of documents [46–49]. Among others, SPAR includes ontologies to describe bibliographic metadata (FaBiO), to structure scholarly documents (DoCO), and to describe rhetorical

[1] http://purl.org/spar/{cito,c4o,fabio,biro,pro,pso,pwo,doco,deo}

discourse elements (DEO). Ruiz Iniesta and Corcho [44] reviewed the state-of-the-art ontologies to describe scholarly articles. Sateli and Witte [50] use some of these scholarly ontologies to add semantic representations of scholarly articles to the Linked Open Data cloud. For a literature survey comparing scholarly ontologies see Table 3.1. Most of these ontologies are designed to capture metadata about and structure of scholarly articles, not the content communicated in articles. We created an additional literature survey to compare approaches for semantically representing scholarly communication, see Table 3.2.

Publishing data as structured or semantic data is a well researched topic among various domains. For example, certain challenges related to publishing semantic open government data are similar to the challenges in scholarly communication. This includes extracting data from legacy documents, often in PDF format [56, 57]. Furthermore, in the literature use cases are described on publishing unstructured data as semantic data (e.g., [58–60]). These existing approaches cannot be adopted one to one for the scholarly domain since they generally aim to semantify a homogeneous set of documents. This enables them to create data specific ontologies. Instead, in our work, we are creating a domain agnostic approach that is applicable to a wide array of research fields.

3.1.1 FAIR Scholarly Knowledge

In light of the FAIR Data Principles [26], scholarly data should be Findable, Accessible, Interoperable and Reusable both for humans and machines. Due to the publication format, scholarly documents generally only weakly adhere to the FAIR guidelines. Scholarly data should be considered first-class objects [61] Rodríguez-Iglesias et al. [62] describe the difficulties of making data FAIR within the plant sciences. They argue that it is more complicated than reformatting data. On the other hand they suggest that most FAIR principles can be implemented relatively easily by using off-the-shelf technologies. Boeckhout, Zielhuis and Bredenoord [63] argue that the FAIR principles alone are not sufficient to lead to responsible data sharing. More applied principles are needed to ensure better scholarly data. This claim is supported by the findings of Mons et al. [32] who suggest that there are very diverse interpretations of the guidelines. In their work, they try to clarify what is FAIR and what is not.

3.1.2 Crowdsourced Scholarly Knowledge

In order to generate structured scholarly knowledge, crowdsourcing can be leveraged. Large complex tasks can be decomposed into a set of smaller, independent microtasks [64]. These microtasks are context-free, are more manageable, and are generating higher quality results [65]. The context-free setting relates to the absence of required prior task knowledge from a user perspective. While microtasks can be beneficial on an individual level, such as microwork [66], they are commonly performed in a crowdsourced setting by unskilled users [67]. In a crowdsourced setting, a large task, too big in scope for a single person, can

Table 3.1: Comparison of various scholarly ontologies. These ontologies are all related to scholarly communication, topics include scientific discourse, document description and citations. Comparison via the ORKG: `https://doi.org/10.48366/r8342`

Ontology	Full name	Description	Class count	Data property count	Object property count	Example class	IRI	Website
BIBO	Bibliographic Ontology	Describing citations and bibliographic references (including quotes, books and articles)	70	55	53	Agent, Collection, Document, Event	http://purl.org/ontology/bibo	http://bibliontology.com/
EXPO		Ontology for describing scientific experiments, including the design, methodology and results	325	0	78	Abstract, Entity, Physical, Proposition		http://expo.sourceforge.net
ORB	Ontology of Rhetorical Blocks	Coarse-grained rhetorical scholarly document structure	7	0	0	Discussion, Header, Introduction, Results	http://purl.org/orb	https://www.w3.org/TR/hcls-orb
ScholOnto		For scholarly interpretation and discourse. Describe debate and relations to the literature	29	6	44	PolarityType, RhetProperty, WeightType		http://projects.kmi.open.ac.uk/scholonto
PWO	Publishing Workflow Ontology	Used for describing the workflow steps for the publication of a document	65	31	102	Agent, Event, Situation, TempopralPosition, TimeZone	http://purl.org/spar/pwo	http://www.sparontologies.net/ontologies/pwo
DoCO	Document Components Ontology	For document structure (paragraph, chapter etc.) and rhetorical elements (introduction, discussion)	122	3	8	Abstract, Bibliography, Figure, Footnote, Paragraph	http://purl.org/spar/doco	http://www.sparontologies.net/ontologies/doco
PSO	Publishing Status Ontology	Can record the status for the publication of a document (draft, submitted etc.)	43	44	80	Agent, Document, Event, PublishingStatus	http://purl.org/spar/pso	http://www.sparontologies.net/ontologies/pso
PRO	Publishing Roles Ontology	For describing the different roles in the publication process (authors, editors, reviewers etc.)	39	44	82	Agent, PublishingRole, Role, RoleInTime	http://purl.org/spar/pro/	http://www.sparontologies.net/ontologies/pro
C4O	Citation Counting and Context Characterisation Ontology	Recording of in-text citations including their textual citation context	33	8	73	BibliographicInformation-Source, GlobalCitationCount, InTextReferencePointer	http://purl.org/spar/c4o	http://www.sparontologies.net/ontologies/c4o
BiRO	Bibliographic Reference Ontology	Define bibliographic records, bibliographic references	26	3	66	BibliographicCollection, BibliographicList, BibliographicRecord, BibliographicReference	http://purl.org/spar/biro	http://www.sparontologies.net/ontologies/biro
CiTO	Citation Typing Ontology	For describing citations, both factually and rhetorically	11	3	100	Citation, DistantCitation, JournalCartelCitation, SelfCitation	http://purl.org/spar/cito	http://www.sparontologies.net/ontologies/cito
FaBiO	FRBR-aligned Bibliographic Ontology	Describing entities that can be published (e.g., papers). Focus on bibliographic references	268	67	71	Item, Manifestation, StorageMedium, Work	http://purl.org/spar/fabio	http://www.sparontologies.net/ontologies/fabio

Table 3.2: Comparison of semantic representations of scholarly communication. Papers are included that focus on scholarly communication as a whole (not only on specific data, such a citations or authors). Comparison via the ORKG: `https://doi.org/10.48366/r8364`

Semantic representation	ORKG [51]	Nanopublications [52]	Micropublications [53]	RASH [54]	Dokie.li [55]
Acquisition	Authors, Crowdsourcing	Authors, Automatic extraction	Authors	Authors	Authors
Prospective / retrospective	Prospective, Retrospective	Prospective, Retrospective	Prospective	Prospective	Prospective
Data type	Free text	Free text	Free text	Quoted text	Quoted text
Scope	Summary	Statement level	Full paper	Full paper	Full paper
Discourse	Partially	Partially	✓	✗	✓
High level claims	✓	✓	✗	Partially	✗
Metadata	✓	✓	✓	✓	✓
Natural language statements	✓	✗	✓	✓	✓
Supports research data	Partially	✓	✓	✗	✗
Knowledge representation	Metadata, RDF	RDF	OWL, RDF	HTML, RDFa	HTML, RDFa

be completed collaboratively. Microtask crowdsourcing has been successfully employed for various tasks, for example, writing software programs [64], validating user interfaces [68], labeling machine learning datasets [69], ontology alignment [67], and knowledge graph population [70]. Microtasks can also play a key role in generating structured scholarly knowledge. This includes both creation and validation of such knowledge.

3.2 Literature Reviews

Survey articles provide well-structured overviews of the literature [71]. The terms "literature review" and "literature survey" are used interchangeably in the literature, often depending on the specific research domain. Review articles are a traditional and common means to organize literature. The typical structure of a review article consists of a comprehensive description of the research problem, textual analysis and explanation of the existing literature, and an outlook of future research directions. Additionally, tabular overviews of the literature are often included to provide a summary of the work. Such semi-structured tabular literature overviews can be leveraged to populate scholarly knowledge graphs. Review articles are generally considered valuable by research communities which frequently results in highly cited reviews articles [72]. Among other things, literature reviews are helpful in delimiting the research problem, avoiding fruitless approaches [73] and to discover new research directions [74]. Conducting a literature review is a complicated and time-consuming activity [71]. When literature reviews are not available for certain fields, its development could be weakened [75]. Because of the importance of literature surveys to scientific research, leveraging review article tables to build a graph results in a high-quality and relevant scholarly knowledge graph. Some existing work with respect to semantifying literature surveys exists [76–78]. However, those approaches are not (semi-)automated and are therefore not scaling well to larger amounts of survey articles.

3.2.1 Weaknesses of Current Review Approach

The current review authoring and publishing method faces numerous limitations and weaknesses [78]. Table 3.3 summarizes the weaknesses and includes a list of supporting related work. We consider two most pressing weaknesses the inability to update articles once published and to the machine-inactionability of the presented knowledge. Both these topics are extensively discussed in the literature.

Lacking updates. Once an article is published, it is generally not updated [79]. This is caused either by lacking incentives from the author's perspective or due to technical limitations. For most research articles, this is acceptable. After all, if new results are

Table 3.3: Summarized weaknesses of the current review approach and their respective related work.

Weakness	Definition	Related work
Lacking updates	Published articles are generally not updated due to technical limitations or lacking author incentives	[78–80]
Lacking collaboration	Only the viewpoint from the review authors is reflected and not from the community as a whole	[78, 81]
Limited coverage	Reviews are only conducted for popular fields and are lacking for less popular ones	[71, 75, 78]
Lacking machine-actionability	The most frequently used publishing format is PDF, which hinders machine-actionability	[8, 57, 78, 82–85]
Limited accessibility	The articles in PDF format are often inaccessible for readers with disabilities	[86–88]
Lacking overarching systematic representation	Web technologies are not used to their full potential because systematic representations are often lacking	[78, 89]

available, it provides an opportunity to publish a new article building upon previous work. However, specifically for review articles this implies that the articles are outdated soon after they are published.

Lacking collaboration. Reviews include research articles created by numerous authors. With the current review method, only the viewpoint of the review authors is considered and not from the community as a whole. This potentially imposes biases and hinders the objectiveness of the discussion of the reviewed work. Schmidt et al. found that a considerable amount of evaluated narrative review articles for the medical domain was severely biased [81].

Limited coverage. Authoring review articles is a resource intensive activity, which is generally more cumbersome than writing a research article [71]. Therefore, reviews are often only conducted for relatively popular domains and are lacking for less popular domains. Since review articles are an important factor for the development of research domains [75], the lack of review articles can potentially hinder the evolution of a domain.

Lacking machine-actionability. The most frequently used format for publishing scholarly articles is PDF, which is hard to process for machines [8]. PDF files focus on visual

presentation specifically designed for human consumption. Nowadays, machine consumption of PDF files relies on machine learning techniques and is often limited to parsing the article's metadata [82, 83].

Limited accessibility. Documents published in PDF format are often inaccessible to readers with disabilities [86]. PDF documents focus on the visual representation of documents instead of a structured representation, which hinders accessibility [87].

Lacking overarching systematic representation. Generally, there is no systematic representation of concepts used in articles, which means scholarly publishing does not use related web technologies to their full potential [89]. This has several implications and potentially causes redundancy and ambiguity across scholarly articles.

3.2.2 Semantic Literature Reviews

The essence of review articles comes down to comparing different articles based on a predefined set of properties. This includes comparisons based on approaches, methods, or results. Such comparisons can be created effortlessly when structured data about the literature exists. The task of comparing papers can be reviewed in light of the more general task of comparing resources (or entities) in a knowledge graph. While this is a well-known task in multiple domains (for instance in e-commerce systems [90]), not much work has focused on comparison in knowledge graphs, specifically. One of the few works with this focus is by Petrova et al. [91] who created a framework for comparing entities in RDF graphs using SPARQL queries. In order to compare contributions, they first have to be found. Finding is an information retrieval problem. As a well-known technique, TF-IDF [92] can be used for this task. More sophisticated techniques can be used to determine the structural similarity between graphs (e.g., [93]) and matching semantically similar predicates. This relates to dataset interlinking [94] or more generally ontology alignment [95]. For property alignment, techniques of interest include edit distance (e.g., Jaro-Winkler [96] or Levenshtein [97]) and vector distance. Gromann and Declerck [98] found that fastText [99] performs best for ontology alignment.

An initial attempt for semantifying review articles was done in [76]. The work comprises a relatively rigid ontology for describing contributions (mainly centered around research problems, approaches, implementations and evaluations) and a prototypical implementation using Semantic MediaWiki. We relax this constraint, since we are not limited by a rigid ontology schema but rather allow arbitrary domain-specific semantic structures for research contributions. The work by Vahdati et al. [77] focuses on semantic article representations for generating literature overviews. Their method is to use crowdsourcing to generate the overviews. Kohl et al. [100] present Cadima, a system that supports systematic literature reviews. The tool supports the formal process of performing a literature review but does, for example, not publish data in machine actionable form for reuse.

Shanahan advocates for "living documents" and to move away from the traditional and obsolete print model in which articles are sealed after publishing [101]. The living documents concept also provides opportunities for article retractions and corrections [102]. This gives the possibility to embrace the features the modern web has to offer, including semantic web technologies [89]. Berners-Lee et al. used to term Linked Data to describe the interlinking of resources (i.e., data) by means of global identifiers, which constitutes the semantic web [16]. The use of the Resource Description Framework (RDF) [103] and SPARQL query language [104] improves the machine-actionability of data and provides a means to make data FAIR [26]. Semantic web technologies also play a key role in the living documents concept presented by Garcia-Castro et al. [105]. This type of document supports tagging and interlinking of individual article components and embeds ontologies in the core of their approach.

Several approaches exist addressing the current issues of scholarly publishing. Dokie.li provides a decentralized article authoring tool supporting semantic data descriptions, making documents machine-readable and interoperable [55]. Specifically, Dokie.li leverages RDF for creating a knowledge graph. Stencila addresses the reproducibility crisis [106] and provides interactive articles with executable code [107]. A similar reproducible workflow can be achieved by using Jupyter Notebooks for publishing workflows [108], possibly extended with Jupyter Books to add additional publishing features [109]. Since these methods do not focus specifically on review articles, they fail to address the current limitations of literature reviews. Considerable research has been conducted on support tools for systematic literature reviews [100]. Tools include Cadima [100], Rayyan [110] and Covidence [111]. These tools successfully support researchers in conducting literature reviews more efficiently, but they do not address the challenges the review method is currently facing.

3.3 Scholarly Knowledge Extraction

We define scholarly knowledge extraction as the process of transforming unstructured data into structured data. This can be done manually by humans, automatically with Natural Language Processing (NLP), or a hybrid method. In this section, we discuss methods for scholarly knowledge extraction. Generally, this is extraction of natural text from Portable Document Format (PDF) articles. Due to the popularity of the PDF, PDF annotation has received considerable research attention (e.g., [112–114]). PDF documents are widely used among various domains, for example, in government data [57], legal documents [115], patents [116], and product datasheets [117]. Additionally, PDF is the most common format for scientific articles [8]. However, the PDF format hinders access and reuse of the data presented within the documents [118].

3.3.1 Table Extraction

Tables present knowledge in a semi-structured manner and are therefore particularly interesting for information extraction. Compared to extracting knowledge from natural text sentences, which requires NLP technologies, tabular extraction already provides a basic data model, by means of table headers. However, extracting tables from PDF documents is a cumbersome process since the tabular structure is not stored within the file itself [9]. This means that regular PDF extraction tools are only able to extract the text within a table, but loosing the tabular structure. Tools that specifically focus on table extraction from PDF files use segmentation techniques to estimate the position of rows and columns [119]. Corrêa and Zander did a literature survey on table extraction tools [57]. They concluded that Tabula[2] is the most suitable open-source tool. Although being a popular and high-performing tool, Tabula is criticized because of the lack of documentation [120]. Table 3.4 provides an overview of methods related to table representation and extraction.

In particular, survey table extraction provides a means to build a high-quality knowledge graph. One of the characteristics of survey tables is that a reference to the original work is provided. Apart from regular table extraction as described previously also references should be extracted to be able to link the tabular data to the respective paper. This is done by parsing the references that are used within a table. For this, the state-of-the-art PDF extraction tool GROBID [121] can be used. GROBID focuses specifically on extracting bibliographic data from scholarly articles [122]. Lipinski et al. compared GROBID to other PDF metadata extraction tools, and found out that GROBID performed best [83].

3.3.2 Sentence Extraction

Text annotation within documents can serve as a simple knowledge extraction method. For example, sentences in PDF articles are highlighted and associated with specific classes. These classes provide further information about the knowledge presented in the sentence. Specific to the scholarly domain, sentences can be annotated with classes such as related work, methods, results, or future work. Compared to the original text, the annotated text provides a more machine-actionable format. Text annotation tools are widely used in the NLP community to visualize automatically generated annotations by NLP tools, such as BRAT [128]. Additionally, some of these tools focus on corpus annotation and support the generation of complex corpora [129]. Such annotation tools have proven to be valuable for the collaborative creation of datasets. Eriksson [112] presents a tool to directly generate semantic descriptions from PDF documents. This tool requires annotators to have data modeling knowledge since the annotator is responsible for the modeling aspect. Shindo, Munesada and Matsumoto [113] integrates multiple linguistic technologies in the annotation tool. Takis et al. [114] presents a crowdsourcing approach for creating semantic annotations in scientific publications. They focus on entity annotation rather than

[2] `https://tabula.technology`

Table 3.4: Comparison of different approaches and steps involving table extraction. Some methods focus on specific parts of tabular extraction (e.g., extracting a schema) while other approaches provide a full pipeline for extracting tables. Comparison via the ORKG: `https://doi.org/10.48366/r36099`

Name	ORKG [123]	SemAnn [114]	Web Tables [124]	TableSeer [125]	TEXUS [126]	[127]
Method automation	Semi-Automatic	Semi-Automatic	Automatic	Automatic	Automatic	Automatic
Scope	Survey tables	Scholarly articles	Web tables	Scholarly articles	Documents (application agnostic)	Web tables
Knowledge graph creation	✓	✓	✗	✗	✗	✗
Supports reference extraction	✓	✗	✗	✗	✗	✗
User interface	✓(table viewer)	✓(annotation)	✗	✓(search)	✗	✗
Input format	PDF	PDF	HTML	PDF	PDF	HTML, Spreadsheet (xls)
Output format	JSON, RDF	RDF	JSON	Relational database	Abstract table representation	Relational schema
Summary	A semi-automatic approach of extracting survey tables for creating a scholarly knowledge graphs. Provides a user interface for displaying the tables	A manual approach of annotating scholarly PDF documents, including tables, to generate a semantic version of the paper	In this study, tables were extracted from the CommonCrawl corpus by using the WebDataCommons extraction framework.	Method to automatically detect tables in PDF documents and capture the table's metadata. Provides an interace to search for tables	End-to-end table processing of table in PDF documents. Includes tasks of document convertion, table locating and structural analysis of tables	Approach to automatically extract a schema for tabular data on the web.
Task	Automatic paper reference extraction, Automatically extract tables from PDFs, Knowledge graph creation from CSVs, Manual table formatting / quality control, Manually select survey papers , Manually select tables	Automatic RDF lookup , Convert PDF to HTML, Export table to CSV, Manual selection of tables, Manually annotate table text	Detecting key columns , Detecting table headers, Determining table orientations, Efficient extraction of tables from HTML pages, Extract timestamp and context information from HTML page, Table classification (layout or content)	Crawling of scientific documents, Extract tables from documents , Identify documents with tables, Indexing tables, Represent table with metadata file	Document conversion (PDF to text) , Functional analysis (identify role of each cell) , Segmenting: recognize table boundaries, Structural analysis (detect logical relationships between table cells), Table locating (find tables)	Classify rows, Detect row classes (i.e., understanding the contents of a row), Schema construction, Select row features

full sentence annotation. Furthermore, the integration of ontologies in their approach is prominently present. Dokie.li provides an interface that enables authors to create semantic annotations within the authoring tool itself [55]. Instead of annotating existing documents (retrospective), Dokie.li provides a method to annotate terms during the authoring process (prospective). Apart for making text annotations, annotating datasets is also a frequently recurring task in the literature. Such annotations can be used to create datasets, for example, to create a gold standard. Snow et al. [130] has demonstrated that crowdsourcing can be successfully employed to generate labeled datasets. Such crowdsourcing approaches rely on comprehensive task descriptions and guidelines to ensure high-quality results [69]. So when employing crowdsourcing in annotation tasks, one has to make a clear task description and leave no room for ambiguity.

3.3.3 Automatic Extraction

Machine learning tools are able to process data at scale without the need for human assistance. Therefore, such tools are especially suitable to handle large quantities of data, such as scholarly article corpora. The NLP domain focuses specifically on understanding natural language for machines [131]. We now list a set of five NLP tools that are particularly relevant to scholarly knowledge extraction. First, *Named Entity Recognition* (NER) is a task to identify entities within text belonging to a predefined class [132]. For example, the task of identifying the classes "materials" and "methods" within a scholarly article. Second, *Entity Linking* is the task of linking entities to their respective entry in a knowledge base [133]. This includes the task of entity disambiguation, to ensure entities are not only syntactically but also semantically the same. For example, the entity "Python" can be linked both to the animal and the programming language. The context determines which link is correct. Third, *Topic Modeling* is the task to identify and distinguish between common topics occurring in natural text [134]. This allows for classifying papers based on their mutual topics. Finally, *Text Summarization* is the task of compressing text into a shorter form, while preserving the key points from the original text [135].

CHAPTER 4

Introducing the Open Research Knowledge Graph

In this chapter, we introduce the technical infrastructure used in consecutive chapters. The infrastructure serves as the foundation for the work conducted in this thesis. It provides a means to organize structured scholarly knowledge with a focus on the manual curation aspect. In addition, the infrastructure includes a set of tools to support humans in manually creating and validating scholarly knowledge. Specifically, within this chapter, we address the following research question:

> **RQ1:** How to organize scholarly knowledge using a manually curated scholarly knowledge graph?

We address this research question by introducing the Open Research Knowledge Graph (ORKG). The ORKG is a scholarly knowledge infrastructure focusing on findable, structured, and FAIR scholarly knowledge. The infrastructure provides a toolset to support the consumption, population, and curation of this scholarly knowledge. From a software architecture point of view, we distinguish between the *frontend* and the *backend*. The frontend relates to the Graphical User Interface (GUI) parts of the system. This includes the ORKG website[1] and the tools made available there. The backend exposes APIs to access the underlying data, and those APIs are used by the frontend. Part of this thesis is the design and development of the ORKG frontend tools. Therefore, this chapter focuses on the frontend, explaining the tools from a user perspective. The provided toolset forms the foundation for several chapters and publications, which will be indicated in this chapter. The remaining tools are not directly associated with a chapter and are therefore explained in more detail in this chapter. The development efforts of the frontend tools focus on creating a sustainable service for researchers. Therefore, several recurring objectives are mentioned

[1] `https://www.orkg.org`

throughout this thesis. The service components and tools have to be well designed and well documented. This enables researchers to use the tools without needing human assistance. This also means that usability is a key aspect during the development. Furthermore, to create a sustainable service, the maintainability of the code base is essential. To accomplish this, the ORKG adopts popular software frameworks, including React and Spring Boot. In addition, technologies such as Git, an issue tracker, and code reviews are leveraged to ensure maintainable and high-quality code. Software developments conducted as part of this thesis also guide future development efforts of the ORKG.

This chapter is based on four joint publications [51, 136–138][2]. The remainder of this chapter is structured as follows: Section 4.1 discusses the technical infrastructures, system services, and implementation details. Section 4.2 explains the data model and the ORKG vocabulary. Section 4.3 lists the different system actors and how they are related to each other. Section 4.4 describes several frontend tools which form the main contribution of this chapter. Finally, Section 4.5 summarizes this chapter and answers the research question.

4.1 Technical Infrastructure and Implementation

The ORKG adopts a microservice architecture where different software services are responsible for their specific task. An excerpt of services is listed in Table 4.1. Only the services of interest for this chapter are listed. For each service, Docker files are provided to improve the local development experience. The frontend provides the graphical user interface, which is used most frequently by researchers to interact with the ORKG. For maintainability purposes, the frontend uses third-party libraries where appropriate. For example, React[3] is used in combination with Redux[4] for state management. Furthermore, libraries such as Bootstrap[5] are used for user interface components. The frontend code uses a *component-based architecture*, meaning that different user interface elements can be reused throughout the ORKG frontend. This benefits both code quality and development speed, as code can easily be reused or updated. Figure 4.1 visualizes the services grouped by layers.

At the core of the backend, we use a labeled property graph. Specifically, a Neo4j[6] graph database is leveraged, which stores the ORKG knowledge graph. The data model used within Neo4j is based on the Resource Description Framework (RDF). This provides flexibility to store data in arbitrary data triples and compatibility with the RDF notation. The data model is discussed in more detail in the next section. The backend is written in Kotlin and uses

[2] This chapter is based on joint publications with ORKG team members. Contributions part of this thesis are specifically related to the frontend user interfaces. A non-exhaustive list of those contributions is presented in Section 4.4.

[3] `https://reactjs.org/`

[4] `https://redux.js.org/`

[5] `https://getbootstrap.com/`

[6] `https://neo4j.com`

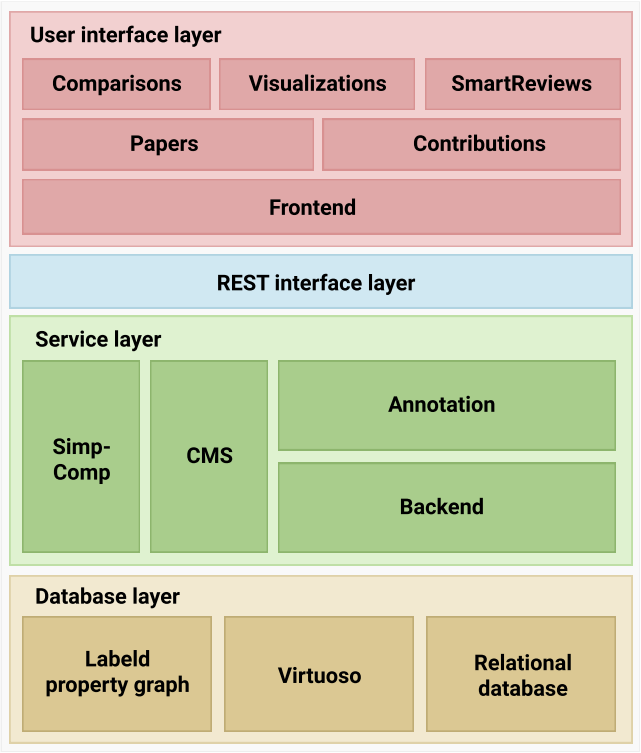

Figure 4.1: The ORKG infrastructure, grouped into a database layer, service layer, REST layer, and user interface layer.

the Spring Boot framework. It exposes a Representational State Transfer (REST) API, which provides access to the data within the Neo4j database. The documentation is available online.[7] The API enables users and clients to interact with the ORKG without the need of Neo4j's query language Cypher. One of the clients that uses the REST endpoints is the frontend. Due to the strict separation between front and backend, most application logic is handled in the backend. The separation of concerns is beneficial in multiple ways. Any ORKG user can directly use the available APIs to create their own programs and tools. For example, downstream Python applications can interact with the graph data directly. This provides the possibility to develop scripts for specific use cases, for example, a MyBinder script that visualizes data coming from the ORKG[8]. The ORKG data is made available as RDF data. A daily data dump is generated from the Neo4j data and converted to RDF. The dump is imported in a Virtuoso[9] server that is used to provide a read-only SPARQL endpoint to

[7] https://tibhannover.gitlab.io/orkg/orkg-backend/api-doc/index.html
[8] https://mybinder.org/v2/gl/TIBHannover%2Forkg%2Forkg-notebooks/master?urlpath=lab/tree/covid_19/R0/R0-estimates-plot.ipynb
[9] https://virtuoso.openlinksw.com/

Table 4.1: Excerpt of most important services part of the ORKG.

Name	Description	Language	Framework	Repository
Frontend	Contains the code of the graphical user interface which runs on `www.orkg.org`	JavaScript	React	`https://gitlab.com/TIBHannover/orkg/orkg-frontend`[11]
Backend	General backend code, it provides REST APIs to handle graph data	Kotlin	Spring Boot	`https://gitlab.com/TIBHannover/orkg/orkg-backend`
SimpComp	Microservice that handles generation and storage of comparison data	Python	Flask	`https://gitlab.com/TIBHannover/orkg/orkg-similarity`
Annotation	Microservice which provides API access to machine learning models	Python	Flask	`https://gitlab.com/TIBHannover/orkg/annotation`
CMS	Microservice for a headless Content Management System (CMS)	NodeJS	Strapi	`https://gitlab.com/TIBHannover/orkg/strapi`

the RDF data. Apart from the Neo4j database, a relational database manages users and groups (i.e., *observatories*). This database contains data that is either private or should be unchangeable for regular users.

Furthermore, two Python microservices provide additional functionalities related to specific ORKG features. Both services use Flask for running a web server. The SimpComp service is responsible for generating data used within ORKG Comparisons. The concept of ORKG Comparison is extensively discussed in Chapter 5. The service also enables the persistent storage of graph data. To accomplish persistency within the knowledge graph, a snapshot of the subgraph is serialized in JavaScript Object Notation (JSON) and stored within a separate document-based database. The Annotation service is used to run several Machine Learning models. The service exposes those models via REST APIs. Some of the models within the services are further discussed in Chapter 8. Finally, a headless CMS is used to display dynamic content in the ORKG frontend. This service mainly focuses on content that changes frequently. This includes news messages, about pages, and help center articles and categories[10]. The use of a CMS enables users within programming knowledge to also manage content within the platform.

[10] The ORKG help center uses the CMS: `https://www.orkg.org/orkg/help-center`

[11] A persistent version of the V0.65 release of the ORKG frontend is available via `https://doi.org/10.5281/zenodo.6347515`

4.2 Data Model and Vocabulary

The ORKG adopts and extends the RDF data model. We now introduce the ORKG data model and discuss its similarities and differences with the RDF data model. Afterwards, we describe the ORKG vocabulary, which contains the concepts and terminology used within the system.

4.2.1 Data Model

Knowledge is described in triple format, which consists of a *subject*, *predicate*, and *object*. These data triples are called *statements* in the ORKG, and they contain *entities*. The subject and object position can contain the entities: *resources*, *properties*, and *classes*. The predicate position contains *properties* and is used to denote the relation between the subject and the object. Additionally, the object position can contain literals. All entities and the statements have their own REST endpoint, provided by the backend. By REST convention, all entities have unique Identifiers (IDs). Different from RDF, literals, and statements also have IDs. IDs are automatically generated when creating entities. They follow a sequential pattern in the form of {*type*}{*number*}. The type represents a single-letter type indication, which is determined based on the entity type. The number represents the automatically assigned sequential number. For example, *P100* and *R250* represent a property (P) and a resource (R), respectively. The generated IDs are purposefully generated without containing the semantic meaning of the entity they represent. This ensures that URIs are stable, i.e., when the entity is updated, the URI remains the same. Additionally, multilingual knowledge is better supported. Resources can be associated with classes, which provides the possibility to instantiate classes. Most commonly, the subject position contains resources, and the object position contains resources or literals. This is used to describe instance data (i.e., *ABox*). The use of properties and classes in the subject position is mainly relevant when describing vocabularies (i.e., *TBox*). Each entity is associated with provenance data. This data is stored via properties on node level and can be accessed when reading an entity. This provides a more efficient provenance mechanism compared to RDF, where provenance data is stored by making separate provenance statements. By default, entity provenance data comprises creator, creation date, and extraction method (either manually or the name of the tool used to generate this entity). Furthermore, provenance data related to the organization and user observatory is recorded. Apart from the previously described data model, no other modeling restrictions are imposed. This results in a data model that is sufficiently flexible to express knowledge from a variety of research domains.

4.2.2 Vocabulary, Concepts, and Terminology

The ORKG is populated with multiple concepts. The most important concepts are summarized in Figure 4.2. At the core, *Research Contributions* form a self-contained structured

data description of a *Paper*. A *Paper* has a one-to-one relating to a published scholarly article. A *Research field* is associated with each paper. The research taxonomy from the US National Academy of Sciences [139] is adopted and extended with categories from the arXiv[12] taxonomy. Although no data model is imposed for the content of research contributions, a list of concepts is provided that serves as guidance when modeling data. This list consists of four classes: (i) *Research Problem* (ii) *Material* (iii) *Method* (iv) *Result*. Multiple contributions can be associated with a single paper. This implies that a paper can address multiple research problems with different methods and results. Now we discuss concepts that can be considered derivative products based on the contributions. *Comparisons* consist of a set of related contributions coming from different papers. The set of contributions is accompanied by a list of properties that are displayed within the comparison. *Visualizations* visualize numerical data from comparisons. They provide alternative methods to the tabular comparison view to display data. The previously discussed concepts (papers, comparisons, and visualizations) can be used to generate *SmartReviews*, dynamic literature review articles within the ORKG. *Observatories* provide a means to organize information related to a specific research topic. ORKG users can be part of observatories if they are knowledgeable about the respective research topic. Furthermore, observatories, and individual users, can be part of an organization. This represents typically institutes associated with the user (e.g., a university or research institute). Finally, *templates* structure information within research contributions. They enable users to reuse the same structure across different contributions, which improves the machine-actionability of the created content. Templates within the ORKG serve a similar purpose as SHACL-shapes[13] in RDF [140].

4.3 System Actors

We can distinguish between four different actors within the ORKG: (i) *content consumer* (ii) *content creator* (iii) *content curator* (iv) *AI-powered machine*. The first three actors are humans, while a computer represents the latter one. The actors and the interactions between them are displayed in Figure 4.3. The ORKG specifically focuses on tools to support collaboration between the different actors. First, the content consumers are users that visit the ORKG website to fulfill a particular information need. As the name suggests, those users merely consume content from the graph and make, therefore, no content contributions on their own. Such users are most likely interested in artifacts such as comparisons, visualizations, and SmartReviews. Those artifacts contain dense information relevant to human consumers. Generally, content consumers are researchers or users with an affinity to a specific research domain. Second, there are content creators. These users are researchers leveraging the ORKG infrastructure to structure scholarly knowledge. This can be a description of their own research articles or describing work of others. For the

[12] `https://arxiv.org`
[13] `https://www.w3.org/TR/shacl`

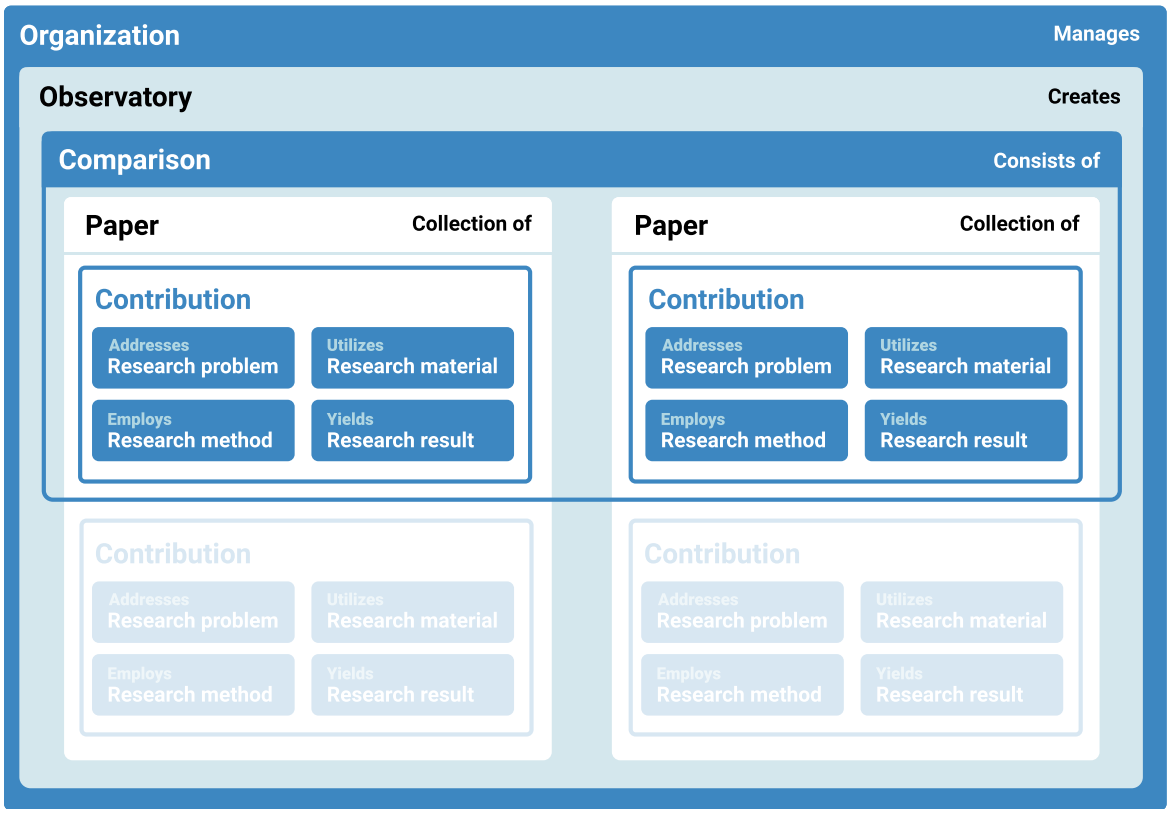

Figure 4.2: ORKG terminology.

latter case, an objective could be to make overviews of related work and state-of-the-art solutions to position their own contributions (as we did ourselves in Chapter 3). Compared to content consumers, only a fraction of the users is content creator. Content creators can be part of observatories and organizations. By joining an observatory, they indicate their knowledge and affinity with a specific research topic. Additionally, the organization also enables not only acknowledging the individual authors but also the organizations they represent. Organization logos are prominently displayed in the user interface to encourage collaboration and adoption of the ORKG between and within those organizations. Third, there are content curators. Those users are domain experts responsible for validating and organizing information within the graph. One of the tasks of the content curators is to define templates for their research domain. Once defined, the templates can be used by content creators to describe the data for their contributions. Finally, there is machine support through Artificial Intelligence (AI). Although AI provides support throughout the entire user interface, the primary interaction happens with content creators. As indicated in Figure 4.3, content creators are supported in the process of adding content to the graph. This includes the suggestion of potentially interesting terms and concepts used to describe articles. The interaction happens both in a *machine-in-the-loop* and a *human-in-the-loop*

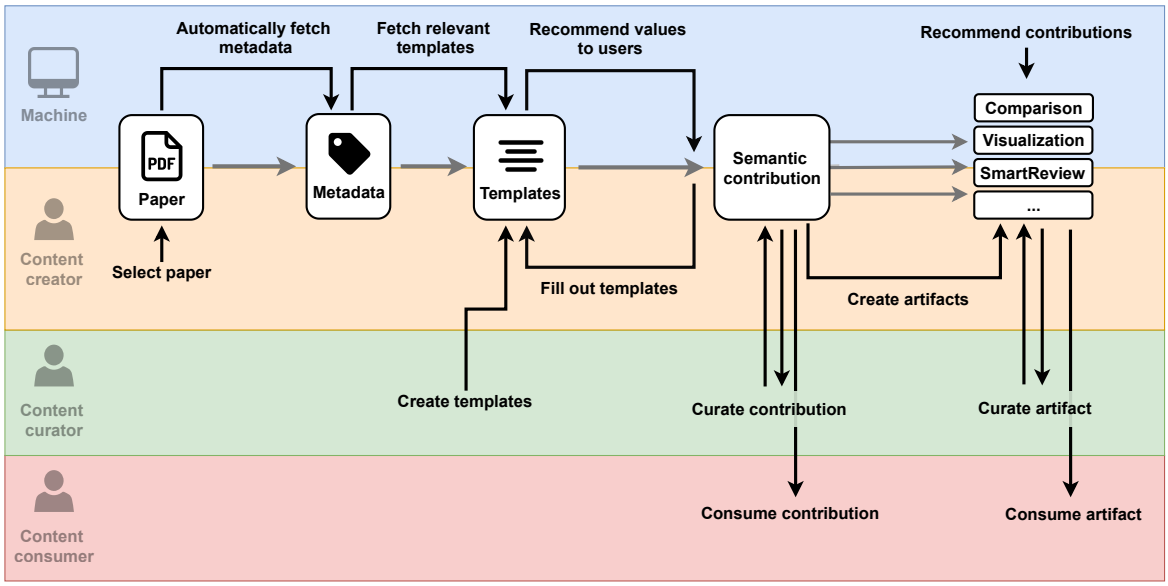

Figure 4.3: The interplay between crowdsourcing and automated approaches in ORKG.

approach. In the former case, the human is assisted by a machine while performing tasks. For the latter case, this is the other way around. Here the human assists the machine in this process. In the end, the collaboration between humans and machines combines the best of both approaches. It uses the intelligence of humans, specifically to determine what is correct and what is not. Furthermore, the scalability of machines is used. Scalability relates to the ability of machines to process large quantities of data without increasing the required time or resources significantly. The human-machine interaction is further discussed in Chapter 8.

4.4 Components and Tools

The ORKG infrastructure contains a set of frontend tools to provide graphical user interfaces for content creation and curation. As mentioned before, usability is a key aspect of these tools. In this section, we discuss several frontend tools in more detail. We specifically focus on tools that are not addressed in the remaining chapters.

4.4.1 Add Paper Wizard

The *Add Paper Wizard* is the entry point to the ORKG for most content creators. The tool presents a user-friendly three-step wizard which guides users through the necessary steps to enter a paper into the graph. The wizard focuses explicitly on inexperienced ORKG users. For example, a welcome tour is offered, which explains each user interface component in

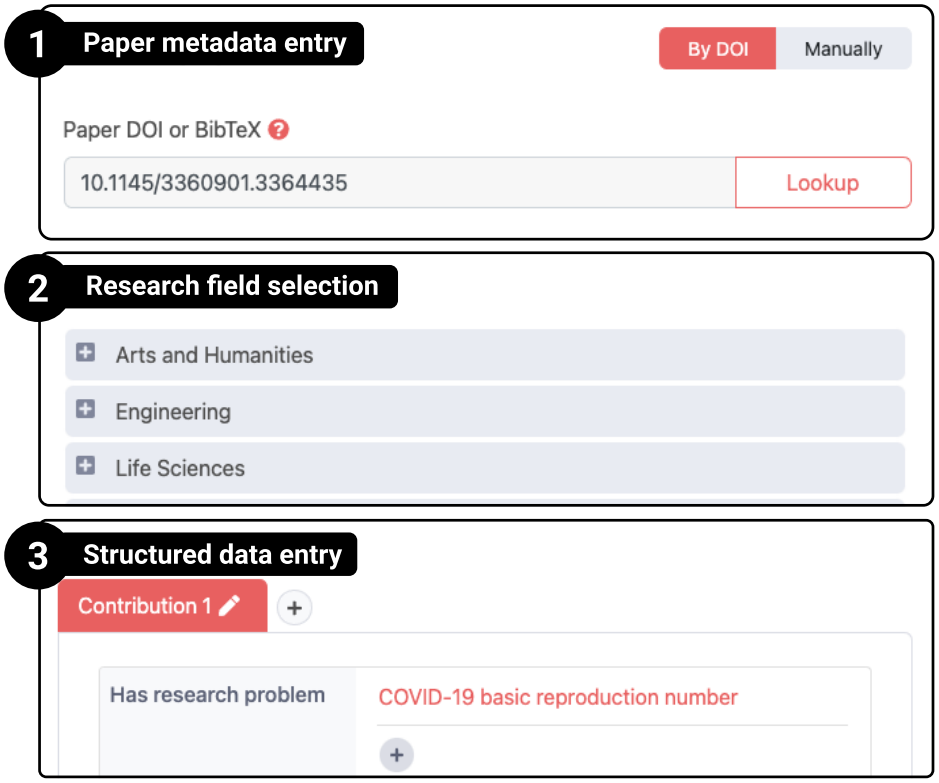

Figure 4.4: Screenshot of the Add Paper Wizard. Displayed are the three steps and their respective user interfaces within the ORKG.

detail. Additionally, tooltips and videos are available to provide further guidance to the user. A screenshot of the three steps is displayed in Figure 4.4. The first step of the wizard captures the metadata of a paper. This includes data such as the title, authors, publication date, and venue. The data can be entered in three ways. Firstly, a DOI can be entered. In this case, Crossref [141] is queried to automatically fetch the available metadata for the provided DOI. The use of DOIs is always the preferred method when entering a paper, as this is used to perform reliable paper disambiguation. Secondly, a BibTeX[14] entry can be provided. Lastly, the metadata can also be added manually. Authors can be entered in two ways, either via a textual author name or via their ORCID [142][15]. When available, ORCIDs are fetched from Crossref. OCRIDs provide a means to disambiguate authors and are used as primary author identifiers in the ORKG. For authors with an ORCID, there is a dedicated author page listing all their ORKG contributions. To lower the entry barrier, only a paper title is required. Before the user continues to the next step, a check is performed to ensure the added paper is not added already to the ORKG. When provided, the DOI is used for this. Otherwise, an exact match lookup on the paper title is performed. The second step of the

[14] A file format used to describe bibliographic references
[15] Open Researcher and Contributor ID

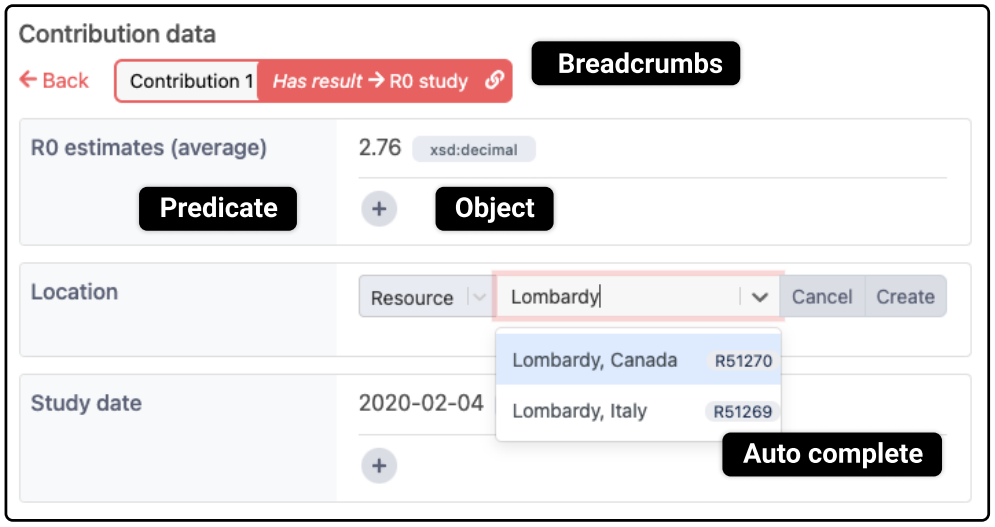

Figure 4.5: Screenshot of the statement browser. In the predicate position, three properties are listed. In the object position, two literals. The autocomplete shows matching resources for the search term "Lombardy". Breadcrumbs are used to indicate which resources are linked to the currently selected resource.

wizard presents the research field selector. Here, users can select a relevant field from the ORKG research field taxonomy (as discussed in subsection 4.2.2). Finally, the third step presents the interface to create structured data related to the paper. This interface is called the *Statement Browser*, and we will discuss this tool in more detail in the next section. An optional intelligent user interface component is integrated to provide automatic concept extraction from the paper abstract. For this, the user has to provide the abstract manually. A machine learning model is leveraged to perform the concept extraction [143].

4.4.2 Statement Browser

The *Statement Browser* is a tool used to handle statements. A screenshot of the statement browser is displayed in Figure 4.5. Any statement stored in the graph can be displayed in the statement browser. It provides a means to browse the graph and traverse its hierarchies. The statement browser supports all Create, Read, Update, Delete (CRUD) operations on statements. Statements are created both in free form and via predefined templates. As the statement browser forms a self-contained user interface component, it can be reused at multiple places in the interface. For example, step three of the *Add paper wizard* uses the statement browser to enable users to create structured contribution descriptions. Additionally, the statement browser is used on the *View paper* page, which lists all ORKG data related to a single scholarly article. Most of the frontend pages provide an option to view the underlying raw data in the statement browser. By default, the statement browser works on a predefined subject. The user can then select properties and objects to create a valid

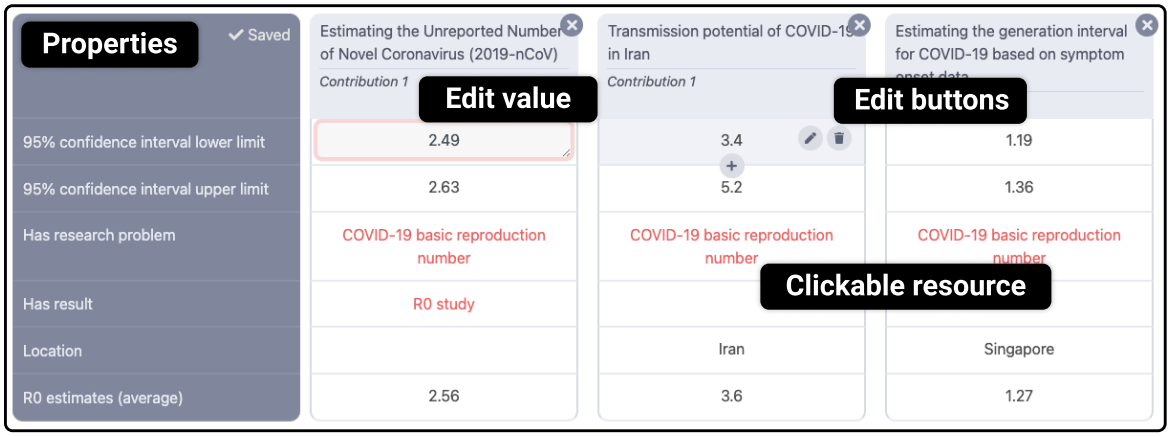

Figure 4.6: Screenshot of the contribution editor. Three contributions are edited in tabular form. Data can be added and changed inline.

data triple. This data can be selected from an autocomplete input field. The autocomplete component does a lookup by a label for both properties and resources to support the reuse of existing graph entities. For new properties created via the autocomplete component, a lookup by the label is performed to ensure no property with the same label exists. If an identical property has been found, a dismissable warning is displayed to inform users about this. The statement browser is a powerful tool that can be used to generate complex graph structures. Less experienced users can still define their data in the statement browser, albeit in a simpler, flatter structure. Additionally, the statement browser supports templates that are used to fill out predefined data structures. This disables the ability to define statements in free form, favoring a predictable and reusable structure. In the end, templates are beneficial for users because they are not responsible for data modeling but only for filling the respective data in the template. It also benefits machine-actionability and comparability of paper data when similar data models are used for knowledge representation.

4.4.3 Contribution Editor

The *Contribution Editor* provides a method to edit multiple paper contributions in the same interface. The tool presents a tabular interface, listing the contributions in the columns and their properties in the rows. The interface is displayed in Figure 4.6. Compared to the *Add paper wizard*, the contribution editor is more suitable for users familiar with the ORKG, as it presents a more powerful interface to edit data in bulk. The contribution editor is especially suitable as an entry point for new comparisons or can be used to edit existing comparisons. Similar to the add paper wizard, the contribution editor also provides the possibility to add new papers to the ORKG. Because of the tabular setup of the contribution editor, it is suitable to edit contributions that have a similar structure. Therefore, the contribution editor serves a similar purpose as the templates in the statement browser: To provide a

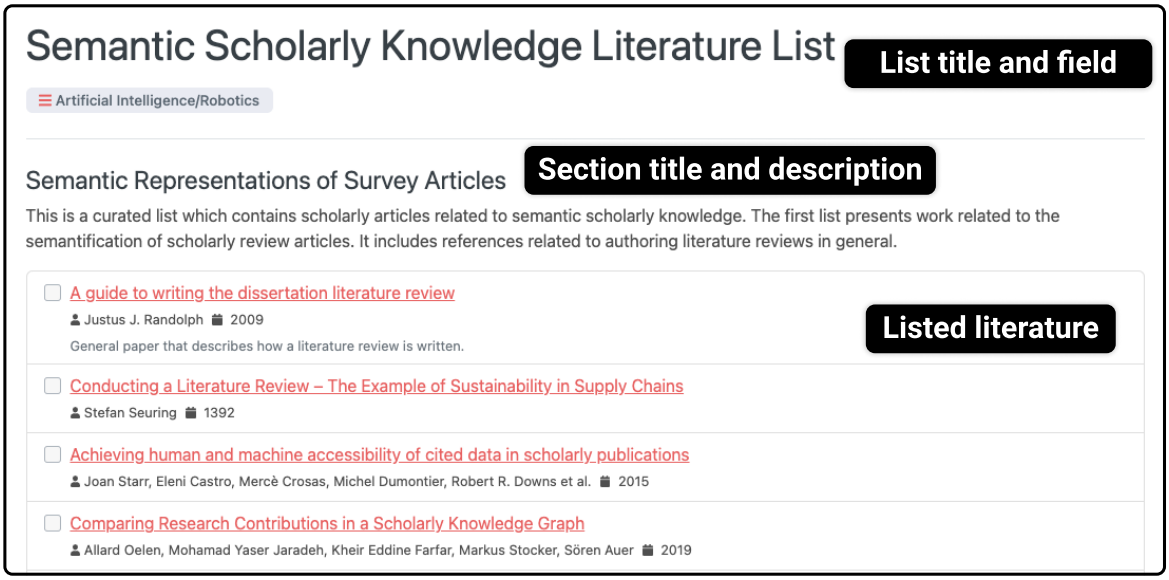

Figure 4.7: Screenshot of a literature list. A list contains sections with a title, description, and the actual literature list.

means to organize information with similar data structures. Data using the same properties across contributions are grouped in a single row. The tabular editor is multidimensional and supports the description of nested data. For this purpose, the statement browser is leveraged. Resources within the contribution editor can be further described using the statement browser.

4.4.4 Literature List

The *Literature List* is a tool that is used to organize literature references within a single list. A screenshot of the interface is displayed in Figure 4.7. Literature can be added to the list by DOI, BibTeX, and manual entry. The BibTeX import supports batch import of multiple references to provide compatibility with alternative literature organization tools. Imported items are represented as papers in the ORKG. By default, the paper metadata is stored, and an empty paper contribution is created. List entries can be grouped by sections, which consist of a title and a description. The literature list feature provides added value to users, for example, by simplifying the import process via DOI support. Furthermore, lists can be created in a collaborative setting, which enables other researchers to improve the lists by adding missing items. Lists can be published to ensure the entire state is preserved, and a published list version remains persistent over time. Literature lists form the starting point for ORKG comparisons. Once a list contains a substantial list of related literature, users can decide to create a comparison based on the listed literature. To support this process, a checkbox is displayed next to each list item. Clicking the checkbox adds the paper to

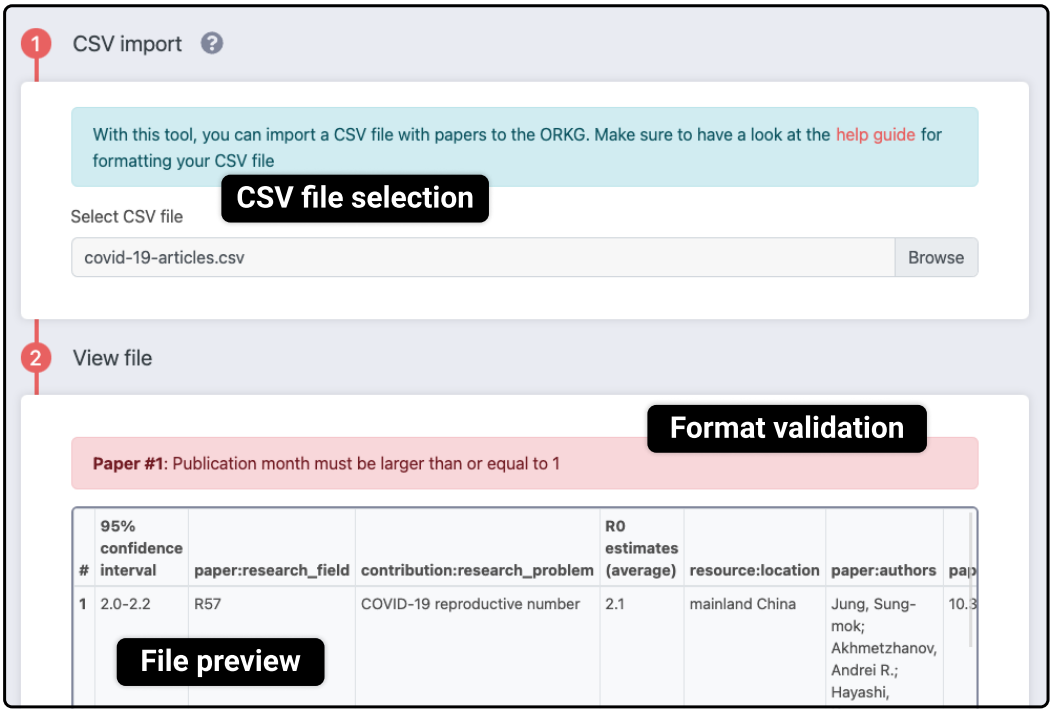

Figure 4.8: Screenshot of CSV import interface. After the file selection, the CSV contents are validated to ensure the correct file structure is used. Additionally, a preview of the parsed CSV data is displayed.

the *Comparison basked* which is a separate popup listing all selected papers. From within this popup, a new comparison can be created. Therefore, the literature list feature forms the most basic method in the ORKG to organize scholarly literature. It serves as an easy entry point to the system and provides the ability to create more comprehensive literature overviews in the form of comparisons.

4.4.5 CSV Importer

The *CSV Importer* focuses on importing existing data into the ORKG. A frequently used method to organize literature is via Comma-Separated Values (CSV) files or other spreadsheet formats (e.g., [144]). Specifically for such cases, the CSV import tool has been developed. To import a file, it has to be structured as follows: The first row is the header row, specifying the properties of the papers. The subsequent rows contain paper data, including metadata. Table 4.2 lists a set of predefined header labels, mainly used to describe the metadata of a paper. By default, all cell values for arbitrary header labels (i.e., not listed in the table) are imported as literals. To use a resource instead of a literal, the cell value can be prefixed with *resource:*. This will create a new resource with the provided label. Additionally, it is possible to reuse existing resources from the ORKG. In these cases, the

Table 4.2: List of the available header labels for the CSV import. Labels prefixed with *paper:* are used to describe metadata related to the paper.

Header label	**Description**
paper:title	Title of the paper. If the DOI of the paper is provided, the title, authors, publication month and year is fetched automatically
paper:authors	Paper authors, separated by a semicolon (e.g., Author 1; Author 2)
paper:publication_month	Numeric value of the publication month (e.g., 6 for June)
paper:publication_year	Numeric value of the publication year (e.g., 2020)
paper:published_in	The conference or journal name
paper:research_field	Research field ID (e.g., R11 for the most general field: "Science", only existing fields can be used)
paper:doi	The DOI of the paper (e.g. 10.1145/3360901.3364435)
paper:url	The URL of the paper (in case no DOI is provided, the URL is displayed instead)
contribution:research_problem	A research problem (e.g., Graph visualization)
Time	Create a literal for the property *Time*
resource:*Location*	Create a new resource with for the property *Location*

cell value has to be formatted as such: *orkg:{resource-id}*. The support for existing resources enables a multidimensional structure for imported data. After a CSV file is uploaded, a preview of the parsed file is displayed. This includes the mapping of imported papers to existing ORKG entries (in case they exist already in the graph) and the mapping of existing resources. Additionally, the CSV contents are validated to ensure it adheres to the required file structure. If formatting violations are detected, an error message is displayed informing that user that the errors should be fixed. By design, the CSV import tool only supports a simple two-dimensional file format. After importing the data, it is possible to edit the data directly in the *Contribution editor*. In this editor, a more flexible data structure can be applied to the imported data. Therefore, the CSV import mainly serves are an entry point for further data semantification.

4.4.6 Research Field Browser

Research fields form a key aspect of information organization for ORKG content. Research fields are assigned to papers, comparisons, visualizations, SmartReviews, literature lists, and research problems. Although most of those concepts directly link to their respective field, in some cases, the research field is automatically derived from the papers used within

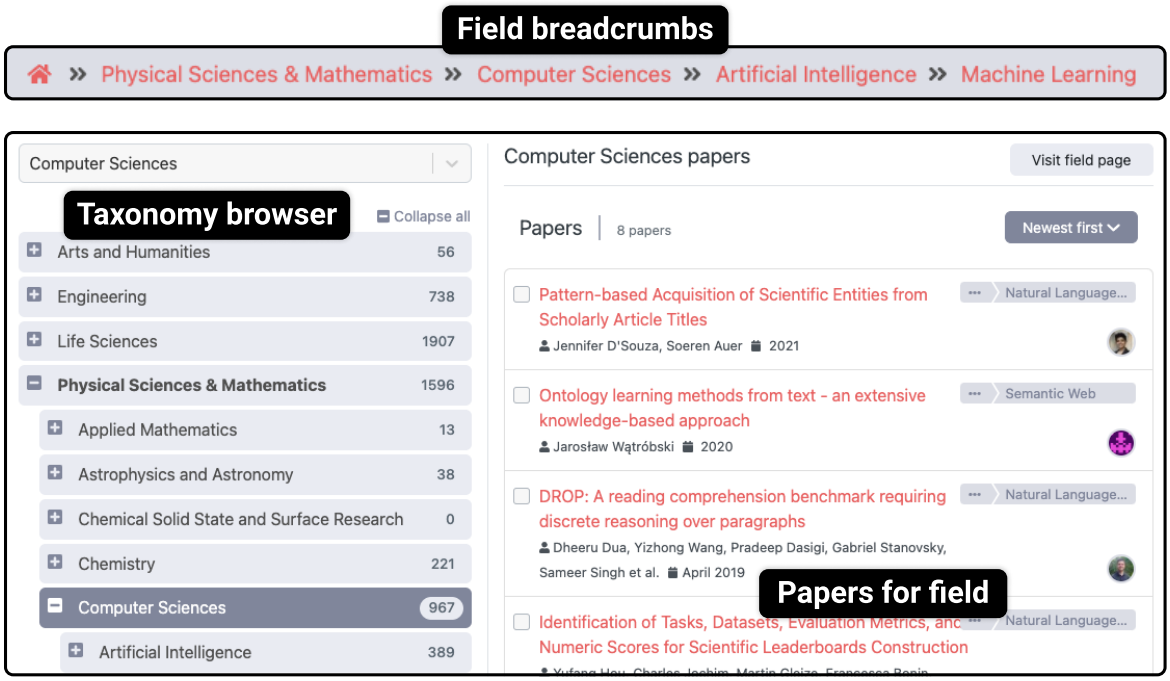

Figure 4.9: Screenshot of the research fields taxonomy page. Field breadcrumbs are displayed for all ORKG concepts related to research fields (e.g., papers, comparisons, SmartReviews).

this concept (for example, this applies to research problems). In Figure 4.9, the research field taxonomy browser is depicted. Also, the *Field breadcrumbs* are displayed. Those breadcrumbs are displayed at the top of each page that has a relation to the respective field. The field items within the breadcrumb link to the field page, which lists different content types for the selected field. The field taxonomy cannot be edited directly by regular ORKG users. In case a field is missing, they can contact the ORKG curation team to request the addition of a new field.

4.5 Summary

In this chapter, we introduced the Open Research Knowledge Graph (ORKG), an infrastructure to describe scholarly knowledge in a structured manner. The ORKG serves as the foundation for the remaining chapters. Each subsequent chapter uses the infrastructure presented in this chapter, either as an implementation platform for user interface components or as a data store for scholarly knowledge. The ORKG services are designed in such a way that a sustainable service is provided to researchers.

We first discussed the technical infrastructure and explained how microservices are used within the system. Afterwards, we explained the data model and introduced ORKG specific vocabulary. We highlighted the differences and similarities between our approach and

the RDF data model. Additionally, ORKG specific concepts and vocabulary are introduced. This includes the terms: contribution, paper, comparison, and SmartReview. Next, the system actors are discussed to explain who is part of the ORKG and to highlight their specific roles. System actors are the content consumer, content creator, content curator, and autonomous machine. Those actors work together to manually curate the content within the ORKG. Finally, we listed several tools as part of the ORKG user interface. The *Add paper wizard* is an easy-to-use three-step wizard to add a single paper to the ORKG. The *Statement Browser* is a component used throughout the interface to enable creation and editing of structured data in the form of triples. The *Contribution Editor* provides a tabular interface to edit multiple contributions concurrently. *Literature Lists* provide curated lists or related literature. They can be used as a starting point to generate comparisons. The *CSV Importer* is a tool to import existing overviews over related literature into the ORKG. Finally, the *Research Field Browser* has a central role in the ORKG to organize the different graph concepts in a predefined taxonomy.

By introducing the ORKG as scholarly knowledge infrastructure, we addressed **RQ1**. The infrastructure and a set of tools are provided to organize scholarly knowledge into a knowledge graph. Specifically, we focus on the human role in generating this structured data, which addresses the manual curation part of the research question. The tools provide different methods to accomplish the same goal: converting unstructured scholarly knowledge into structured scholarly knowledge. This relates to **Challenge 1**, which highlights the difficulty of this process. By providing different interfaces for the same process, we are able to serve users with different levels of expertise. While the *Add Paper Wizard* targets novel users, the *CSV import* provides a method for experienced users to get started with existing data quickly. This also relates to **Challenge 3**, which describes the difficulty to provide an approach that works for a wide variety of domains and users.

CHAPTER 5

Generate FAIR Literature Surveys with Scholarly Knowledge Graphs

When conducting scientific research, reviewing existing literature is an essential activity [145]. Familiarity with the state-of-the-art is required to effectively contribute to advancing it and do relevant research. Mainly because published scholarly knowledge is unstructured [146], it is currently very tedious to review existing literature. Relevant literature has to be found among hundreds and increasingly thousands of PDF articles. This activity is supported by library catalogs and online search engines, such as Scopus or Google Scholar [147]. Because the search is keyword-based, typically large numbers of articles are returned by search engines. Researchers have to manually identify the relevant papers. Having identified the relevant papers, the relevant pieces of information need to be extracted in order to obtain an overview of the literature. Overall, these are manual and time-consuming steps. We argue that a key issue is that the scholarly knowledge communicated in the literature does not meet the FAIR Data Principles [26]. While PDF articles can be found and accessed (assuming Open Access or an institutional subscription), the scholarly literature is insufficiently interoperable and reusable, especially for machines. For units more granular than the PDF article, such as a specific result, findability and accessibility score low even for humans.

In this chapter, we present a methodology and its implementation that can be used to generate and publish literature surveys in form of machine-actionable, comparable descriptions of research contributions. Machine-actionability of research contributions relates to the ability of machines to access and interpret the contribution data. The benefits for researchers of such an infrastructure are (at least) two-fold. Firstly, it supports researchers in creating state-of-the-art overviews for specific research problems efficiently. Secondly, it supports researchers in publishing literature surveys that adhere to the FAIR principles,

thus contributing substantially to reuse of state-of-the-art overviews and therein contained information, for both humans and machines. The methodology is integrated in the Open Research Knowledge Graph (ORKG) infrastructure, as presented in Chapter 4.

Literature reviews are articles that focus on analysing existing literature. Among other things, reviews can be used to gain an understanding of a research problem or to identify further research directions [73, 148]. Reviews can be used by authors to quickly obtain an overview of either emerging or mature research topics [149]. Review papers are important for research fields to develop. When review papers are lacking, the development of a research field is weakened [75]. Compiling literature review papers is a complicated task [71] and is often more time-consuming than performing original research [75]. The structure of such articles often consists of tables that compare published research contributions. We use the terms "literature review" and "literature survey" interchangeably. The state-of-the-art (SOTA) analysis is a special kind of literature review with the objective of comparing the latest and most relevant papers in a specific domain.

In this chapter we address the following research question:

> **RQ2:** How to generate machine-actionable and comparable overviews of related literature?

To answer this question, we divided the research question into three sub-questions:

(i) How to generate literature surveys using scholarly knowledge graphs?

(ii) How to ensure that published literature surveys comply with the FAIR principles?

(iii) How to effectively specify and visualize literature surveys in a user interface?

In support of the first sub-question, we present a methodology that describes the steps required to generate literature surveys. In support of the second sub-question, we describe how the FAIRness of the published literature review is ensured. Finally, in support of the third sub-question, we demonstrate how the methodology is implemented within the ORKG.

This chapter is based on the following publications: [78, 150]. The remainder of this chapter is structured as follows. Section 5.1 motivates the work with use cases. Section 5.2 presents the system design, the underlying methodology, and its implementation. Section 5.3 explains how the knowledge graph is populated with data. Section 5.4 presents the evaluation of the system, specifically system FAIRness, and performance. Section 5.5 discusses the results. Finally, Section 5.6 summarizes the work.

5.1 Use Cases

We motivate our work by means of two use cases that underscore the usefulness of a literature survey generation system. In the first use case, a researcher wants to obtain an overview of state-of-the-art research addressing a specific problem. The second use case describes how a researcher can publish a FAIR-compliant literature review with the ORKG.

Familiarize with the State-of-the-art

A state-of-the-art (SOTA) analysis reviews new and emerging research. They are useful for multiple reasons. Firstly, they provide a broad overview of a research problem and support understanding. Secondly, they juxtapose different approaches for a problem. Thirdly, they can support claims on why certain research is relevant by giving an overview of the breadth of research addressing a problem. The proposed approach enables automated generation of surveys to quickly obtain an overview of state-of-the-art research as well as sharing of surveys for others to reuse.

Publishing of Literature Reviews

Literature reviews typically consist of multiple (survey) tables in which different approaches from original papers are compared based on a set of properties. These tables can be seen as the main contribution and most informative part of the review paper since the tables juxtapose and compare existing work. Comparison tables are published in review papers as static content in PDF documents. This presentational format is generated from datasets that typically contain more (structured) information than what is presented in the published table. However, the additional information is not published. It is "dark data" which is not stored or indexed and likely lost over time [85]. Furthermore, published tables are not machine-actionable. Their overall low FAIRness hinders reusability of the published content. With the presented service, it is possible to publish a literature survey with high FAIRness, i.e. that is compliant with the FAIR principles to a high degree. subsection 3.1.1 discusses this aspect in more details.

Summary of Weaknesses of the Current Approach to Literature Review

The weaknesses of the current approach to literature review can be summarized as follows:

- *Static* – reviews are static, since once published as PDF they are rarely updated and there are no possibilities or incentives for creating new or updated reviews for considerable time.
- *Lack of machine assistance* – machine assistance is hardly possible, since the PDF representation of reviews is only human-readable, and relevant raw data is mostly not published along with the review.

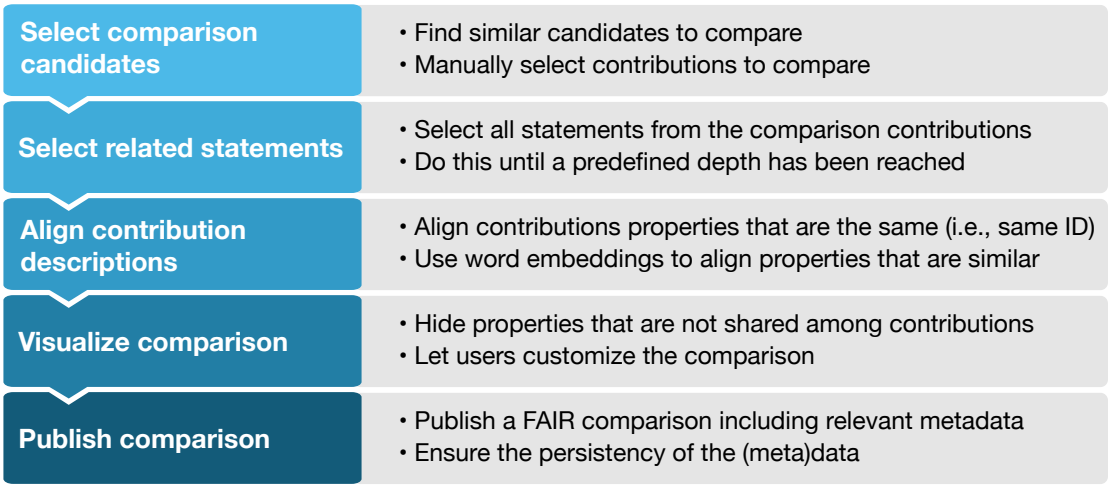

Figure 5.1: Research contribution comparison methodology.

- *Delay* – reviews are produced and published with significant delay (often years) after original research work was done.
- *Coverage* – due to the amount of work required, reviews are often only performed for relatively popular research topics and are stale or missing for less popular topics.
- *Lacking collaboration* – collaboration on reviews is not possible and reviews currently represent only the viewpoint of the few authors not the community.
- *Missing overarching systematic semantic representation* – the overlap between different reviews and related work sections in individual original research papers is not explicit and cannot be exploited.

We deem that these weaknesses of the current approach to scholarly literature review and synthesis significantly hinder scientific progress.

5.2 System Design

We now present the system design of the literature comparison service. It consists of a methodology that describes how to perform a comparison of research contributions. An early version of this methodology has been presented at the 3rd SciKnow workshop [150]. The methodology consists of five steps: 1) finding comparison candidates, 2) selecting related statements, 3) aligning contribution descriptions, 4) visualizing comparisons, and 5) publishing FAIR comparisons. The methodology is depicted in Figure 5.1. First, we discuss the data structure of the ORKG, which forms the foundation of the comparison. Then, each step of the methodology is described in more detail. Finally, we discuss the implementation.

5.2.1 ORKG Ontology

In ORKG, each paper is typed as *paper* class. A paper consists of at least one *research contribution*, which addresses at least one *research problem*. Research contributions consist of *contribution data* that describe the contribution. For instance, a paper in Computer Science might have descriptions for materials, methods, implementation, and results as contribution data. These predefined core concepts can be easily extended with domain-specific research problems, methods, etc. in ORKG curation using crowdsourcing or other curation approaches. The underlying data structure uses the notion of statements. Statements are triples that consist of a subject, a predicate (also called a property), and an object. The granularity of a comparison is at the research contribution, meaning that contributions are compared rather than papers. For simplicity, we use the terms "paper comparison" and "contribution comparison" interchangeably. Because a comparison happens on contribution level, it is possible to compare specific elements of a paper instead of the complete paper. The benefit of this is that a comparison does not contain data from irrelevant contributions. The ORKG OWL ontology is available online.[1] A more detailed description of the ORKG ontology is presented in subsection 4.2.2.

5.2.2 Select Comparison Candidates

To perform a comparison, a starting contribution is needed. This contribution is called *main contribution* and is always manually selected by a user. The main contribution is compared against other *comparison contributions*. There are two different approaches for selecting the comparison contributions. The first approach automatically selects comparison contributions based on similarity. The second approach lets users manually select contributions.

Find Similar Contributions

Comparing contributions makes only sense when contributions can sensibly be compared. For example, it does not make (much) sense to compare a biology paper to a history paper. We thus argue that it makes only sense to compare contributions that are similar. More specifically, contributions that share the same (or a similar set of) properties are good comparison candidates. For instance, a paper about question answering has the property *orkg:disambiguationTask*[2] and another paper is using the same property to describe what disambiguation tasks are performed. Since they share the same property it makes them likely candidates for comparison. Finding similar contributions is therefore based on finding contributions that share the same or similar informative description properties. To achieve this, each comparison contribution is converted into a string by concatenating all properties

[1] `https://gitlab.com/TIBHannover/orkg/orkg-ontology`

[2] *orkg:* denotes the ontology of the ORKG system described in subsection 5.2.1

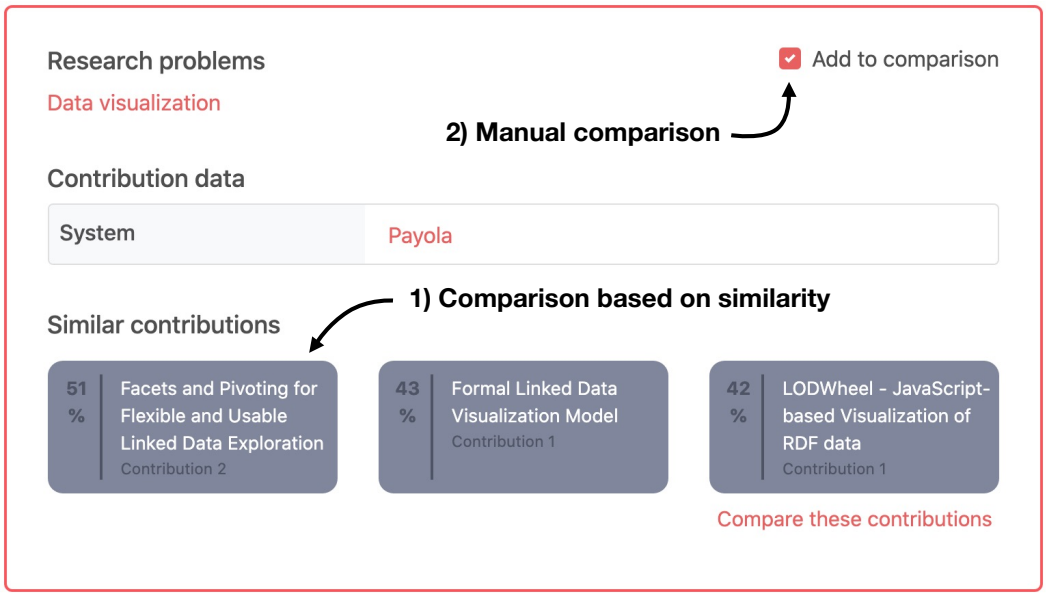

Figure 5.2: Implementation of the first step of the methodology: the selection of comparison candidates. Showing both the similarity-based and the manual selection approaches.

of the contribution. TF-IDF [92] is used to query these strings with the string of the main contribution as query. The search returns the most similar contributions by weighting the most informative properties higher due to TF-IDF. The top-k contributions are selected and form a set of contributions that are used in the next step.

Figure 5.2 displays how the similar contribution selection is implemented. As depicted, three similar contributions are suggested to the user (with the corresponding similarity percentage being displayed next to paper title). These suggested contributions can be directly compared.

Manual Selection

There are scenarios where comparison based on similarity computation is not suitable or desired. For example, a researcher wants to compare a specific set of implementations to see which performs best. Therefore, the manual selection method is implemented in a similar fashion to an e-commerce shopping cart. When the "Add to comparison" checkbox is checked, a box appears listing the selected contributions (Figure 5.3).

5.2.3 Select Related Statements

This step selects the statements from the graph related to the set of contributions selected in the previous step. Statements are selected transitively to match contributions in subject or object position. This search is performed until a predefined maximum transitive depth δ

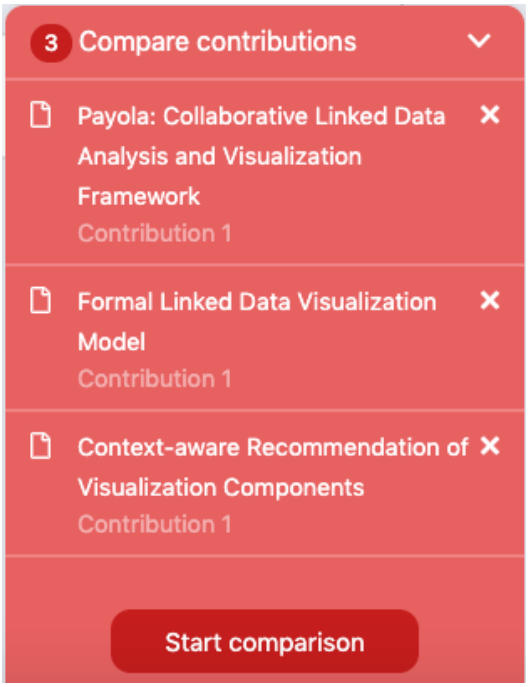

Figure 5.3: Box showing the manually selected contributions.

has been reached. The intuition is that the deeper a property is nested the less likely is its relevance for the comparison. The process of selecting statements is repeated until depth $\delta = 5$ is reached. This number is chosen empirically to include statements that are not directly related to the contribution, but to exclude statements that are less relevant because they are nested too deep.

5.2.4 Align Contribution Descriptions

As described in the first step, comparisons are built using shared or similar properties of contributions. In case the same property has been used between contributions, these properties are grouped and form one *comparison row*. However, often different properties are used to describe the same concept. This occurs for various reasons. The most obvious reason is when two different ontologies are used to describe the same property. For example, for describing the population of a city, DBpedia uses *dbo:populationTotal* while WikiData uses *WikiData:population* (actually the property identifier is P1082; for the purpose here we use the label). When comparing contributions, these properties should be considered as equivalent. Especially for community-created knowledge graphs, differently identified properties likely exist that are, in fact, equivalent.

To overcome this problem, we use pre-trained fastText [99] word embeddings to determine the similarity of properties. If the similarity is higher than a predetermined threshold τ,

the properties are considered equivalent and are grouped. This happens when the similarity threshold $\tau \geq 0.9$ (also empirically determined). In the end, each group of properties will be visualized as one row in the comparison table. The result of this step is a list of statements for each contribution, where similar properties are grouped. Based on this similarity matrix γ is generated

$$\gamma_{p_i} = \left[cos(\overrightarrow{p_i}, \overrightarrow{p_j}) \right] \tag{5.1}$$

with $cos(.)$ as the cosine similarity of vector embeddings for property pairs $(p_i, p_j) \in \mathcal{P}$, whereby $\mathcal{P}$ is the set of all contributions.

Furthermore, we create a mask matrix Φ that selects properties of contributions $c_i \in C$, whereby C is the set of contributions to be compared. Formally,

$$\Phi_{i,j} = \begin{cases} 1 & \text{if } p_j \in c_i \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

Next, for each selected property p we create the matrix φ that slices Φ to include only similar properties. Formally,

$$\varphi_{i,j} = (\Phi_{i,j}) \underset{p_j \in sim(p)}{\underset{c_i \in C}{}} \tag{5.3}$$

where $sim(p)$ is the set of properties with similarity values $\gamma[p] \geq \tau$ with property p. Finally, φ is used to efficiently compute the common set of properties [51]. This process is displayed in Algorithm 1.

Algorithm 1 Align contribution descriptions

```
1: procedure AlignProperties(properties, threshold)
2:     for each property p1 ∈ properties do
3:         for each property p2 ∈ properties do
4:             similarity ← cos(Embb(p1), Embb(p2))
5:             if similarity > threshold then
6:                 similarProps ← similarProps ∪ {p1, p2}
       return similarProps
```

5.2.5 Visualize Comparison

The next step of the workflow is to visualize the comparison and present the data in a human-understandable format. Tabular format is often appropriate for visualizing comparisons since tables provide a good overview of data. Another aspect of the visualization is determining which properties should be displayed and which ones should be hidden. A property is displayed when it is shared among a predetermined amount α of contributions,

where α mainly depends on comparison use and can be determined based on the total amount of contributions in the comparison. By default, only properties that are common to at least two contributions ($\alpha \geq 2$) are displayed.

Another aspect of comparison visualization is the possibility to customize the resulting table. This is needed because of the similarity-based matching of properties and the use of predetermined thresholds. For example, users should be able to enable or disable properties. They should also get feedback on property provenance (i.e., the property's path in the graph). Ultimately, this contributes to a better user experience, with the possibility to manually correct mistakes made by the system.

Figure 5.4 displays a comparison for research contributions related to visualization tools published in the literature. In this example, four properties are displayed. Literals are displayed as plain text while resources are displayed as links. When a resource link is selected, a popup is displayed showing the statements related to this resource. The UI implements some additional features that are particularly useful to compare research contributions.

Customization

Users can customize comparisons including transposing the table as well as hiding and rearranging the properties. Especially the option to hide properties is helpful when contributions with many statements are compared. Only properties considered relevant to the user can be selected to display. Customizing the comparison table can be useful before exporting or sharing the comparison.

Sharing and Persistence

Comparisons can be shared using a persistent link. Especially when sharing the comparison for research purposes, it is important to refer to the original comparison. Since contribution descriptions may change over time comparisons may also change. To support persistency, the whole state of the comparison is stored in a document-oriented database and retrieved when the permalink is invoked.

Export

It is possible to export comparisons in different output formats such as PDF, CSV, RDF and LaTeX. The LaTeX export is useful for direct integration in research papers. Together with the LaTeX table, a BibTeX file containing the bibliographic information of the papers used in the comparison is also generated. Also, a persistent link referring back to the comparison in ORKG is showed as table footnote.

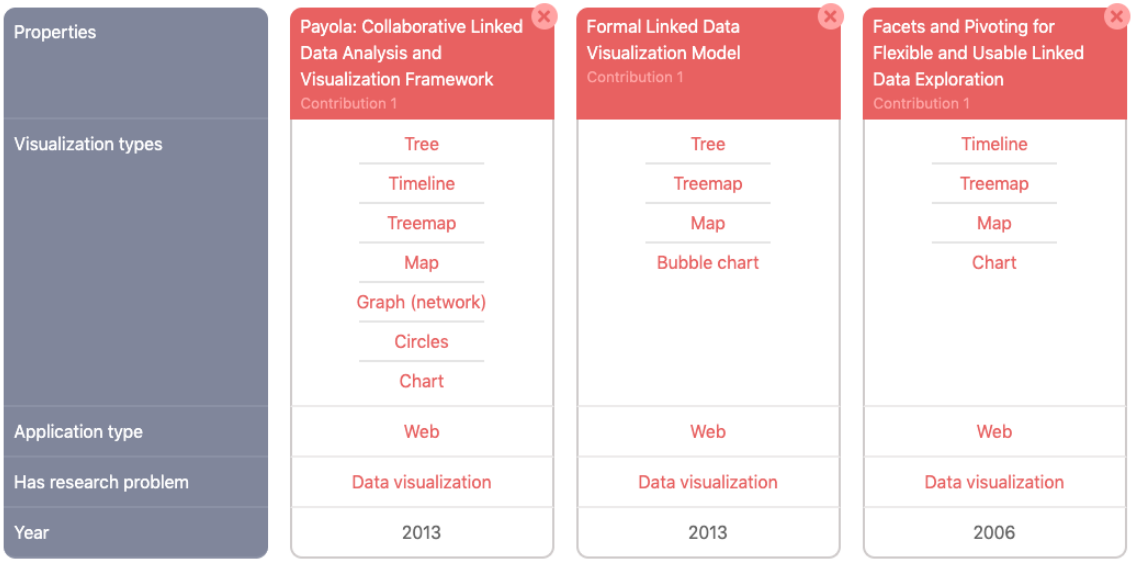

Figure 5.4: Comparison of research contributions related to visualization tools.

5.2.6 Publish Comparison

Visualized and customized comparison tables can be stored. Storing tables is part of the publishing process and therefore only needed when a generated table is going to be used in a paper. In order to regenerate the table the whole state of the comparison should be saved. The knowledge graph from which the comparison was generated changes over time and thus storing just the URIs of the respective papers would not suffice. While saving a comparison, the user can provide additional metadata to ensure findability, an aspect of the FAIR principles. Metadata include a comparison title, which would normally consist of a one-sentence description of the comparison. Additionally, a longer textual description can be provided. This metadata is extended with machine-generated data, such as the creation date and the creator of the comparison. The metadata is stored in the knowledge graph to support easy access and interoperability. In Figure 5.5, the structure of the metadata is displayed using the Dublin Core Metadata Terms[3]. The comparison data itself is stored in a document-oriented database. An RDF export of both the metadata and the comparison data can be generated. The comparison data is modeled with the RDF Data Cube Vocabulary[4]. A unique identifier is attached when the comparison is saved. This ID is used when the comparison is shared or when it is referenced in a paper. The literature comparison can also be performed without publishing. Although the workflow and the steps to create a comparison stay the same, the goal is different. Instead of creating a comparison that will be published and referenced in a paper, the comparison will be used by the researcher herself.

[3] `https://dublincore.org/specifications/dublin-core/dcmi-terms`
[4] `https://www.w3.org/TR/vocab-data-cube`

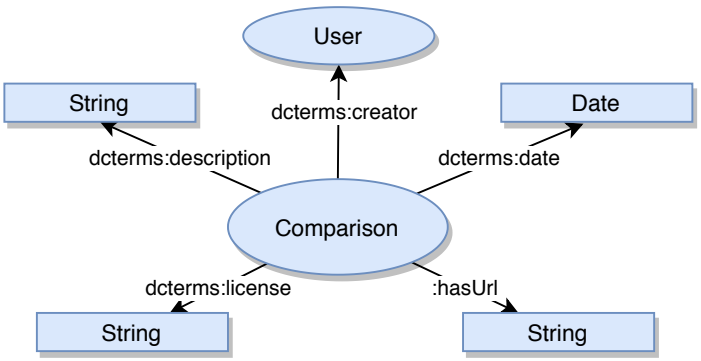

Figure 5.5: The graph structure of the metadata for a published comparison. The *dcterms:* prefix denotes the Dublin Core Metadata Terms ontology.

Table 5.1: List of imported survey tables in the ORKG. The paper and table reference can be used to identify the original table.

Paper reference	Table reference	Research problem	Papers	ORKG representation	Loss
[151]	Table 1	Generic visualizations	11	`https://orkg.org/orkg/c/pdLJDk`	No
[151]	Table 2	Graph visualizations	21	`https://orkg.org/orkg/c/Rx476Z`	No
[152]	Table 2	Question answering evaluations	33	`https://orkg.org/orkg/c/gaVisD`	No
[152]	Table 3,4,5,6	Question answering systems	26	`https://orkg.org/orkg/c/IuEWl2`	No
[153]	Table 4	Author name disambiguation	5	`https://orkg.org/orkg/c/vDxKdr`	No
[153]	Table 5	Author name disambiguation	6	`https://orkg.org/orkg/c/XXg8Wg`	No
[153]	Table 6	Author name disambiguation	9	`https://orkg.org/orkg/c/9rOwPV`	No
[153]	Table 7	Author name disambiguation	6	`https://orkg.org/orkg/c/mB7kIK`	No
[154]	Table 4	Text summarization	52	`https://orkg.org/orkg/c/OUqYB9`	No

5.2.7 Technical Details

The user interface of the comparison feature is seamlessly integrated with the ORKG front end, which is written in JavaScript. The back end of the comparison feature is a service separate from the ORKG back end written in Python and also available Open Source[5]. The comparison back end is responsible for step two and three of the comparison methodology. The input in step two is the set of contribution IDs. The API selects the related statements and aligns the properties and returns the data needed to visualize the comparison. This data includes the list of papers, list of all properties and the values per property.

5.3 Data Collection

In order to generate useful literature reviews it is crucial for the knowledge graph to contain sufficient and relevant papers. Populating the knowledge graph with high-quality paper

[5] `https://gitlab.com/TIBHannover/orkg/orkg-similarity`

descriptions it not straightforward. Structured descriptions of papers should be created in such a way that it is possible to compare papers based on shared properties. Both published papers and papers that will be published in the future should be added to the ORKG, retrospectively or prospectively. Although a comprehensive description on how to populate the ORKG is out-of-scope here, we now briefly describe how we envision populating the ORKG in a manner that would facilitate comparing contributions.

Prospectively, authors can become part of generating structured descriptions of their papers. This should be done in a crowdsourced manner and can become part of the paper submission process. Input templates that collect relevant properties can be used to ensure structured and comparable paper descriptions. Retrospectively, automated (machine learning) methods can be helpful ensure scalability of the process of adding a paper.

Leverage Legacy Review Paper Tables

To populate the ORKG with comparable paper descriptions, we leverage the data published in review papers. Review papers consist of high-quality, curated, and often structured data that is collected from a set of papers that address the same (or a similar) research problem. Hence, using reviews to populate a scholarly knowledge graph is a relatively straightforward approach to obtain high-quality structured paper descriptions. We now present a methodology to convert survey paper data into a knowledge graph structure. The steps are as follows:

1. **Survey paper selection.** The first step is the selection of survey papers that are suitable for building a knowledge graph. Firstly, the survey should compare peer-reviewed scientific articles. For instance, a comparison of different systems without a reference to peer-reviewed work is not suitable for the scholarly knowledge graph. Secondly, the review should compare the papers' content in a structured way and should not merely list work in a field. Especially reviews that present their results and literature comparisons in tabular format are suitable. The result of this step is a list of papers that will be added to the ORKG.

2. **Table selection.** Given the selected survey papers, tables have to be selected. Some surveys contain only one table while in others multiple tables are presented. In some cases, a collection of tables can be joined into one larger table.

3. **Data modeling.** Given the selected tables, a suitable graph structure has to be determined. The data structure has to be modeled. For instance, when implemented systems are compared, a suitable structure could be: `[has implementation] -> System name`. The referenced system can be described with a list of properties to be compared. Additionally, a research problem has to be defined, which is typically the same for all papers that are part of the table.

4. **Metadata collection.** Next, the metadata for the papers that are referenced in the survey table is collected. In case a referenced paper has a DOI[6], the metadata can be automatically retrieved via a lookup service (e.g. Crossref[7]). Otherwise, at least the title, authors, and publication date have to be collected.

5. **Data ingestion.** Finally, the paper data is ingestion into the knowledge graph. The paper data consists of both the paper's metadata and the extracted data from the comparison table. This does not result in a single description of the survey paper. Each paper referenced in the survey table is ingested individually. In order to speed up the process of adding papers, we developed a Python package[8] that has a function to add a paper to the knowledge graph.

This methodology has been used to populate the ORKG with comparable paper data. The data is used to evaluate the presented literature review tool. The imported paper data is not only useful for the evaluation but does also provide significant value to the ORKG itself.

In total, four review papers were selected for importing into the ORKG. The Python script for importing the table data is available online.[9] From those papers, 12 different tables were imported. Together, 169 papers were reviewed in those four survey papers. This resulted in a total amount of 3 750 statements being added to the knowledge graph. Table 5.1 lists the imported review papers and tables. The survey papers address different research problems. Figure 5.6 depicts an excerpt of the resulting graph for one particular paper. A set of comparison tables made with the imported data is available online.[10] This list includes some alternative comparison tables that were generated with the same data.

5.4 Evaluation

In this section, we present an evaluation of multiple aspects of the presented comparison methodology and implementation. Firstly, we evaluate information representation. Then, we evaluate the FAIRness of published reviews. Finally, we present a performance evaluation that tests the scalability.

5.4.1 Information Representation

This part of the evaluation focuses on the aspect of information representation. We use the data from the imported review papers, as described in Section 5.3. In order to build and publish useful and correct literature reviews, at a minimum, our service should display

[6] Digital Object Identifier
[7] `https://www.crossref.org`
[8] `https://gitlab.com/TIBHannover/orkg/orkg-pypi`
[9] `https://gitlab.com/TIBHannover/orkg/orkg-papers`
[10] `https://orkg.org/orkg/featured-comparisons`

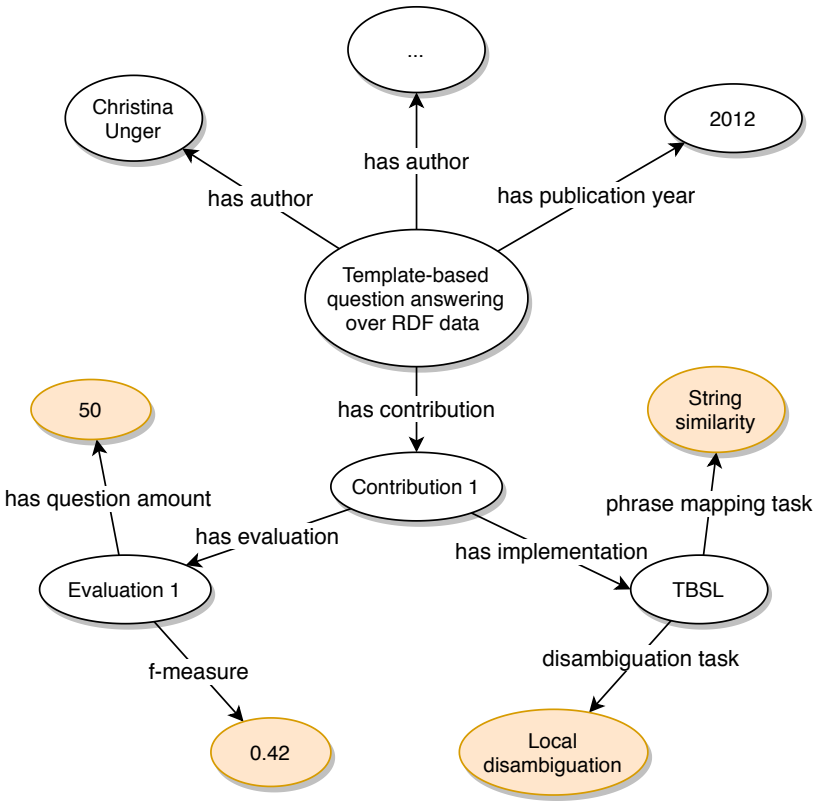

Figure 5.6: Partial graph structure of an imported paper. Orange-colored resources indicate potentially interesting values for a paper comparison.

the same information that was originally presented in the review tables. This means that there should not be information loss when review tables are published using our service. If there is no information loss, it means our service can be used as an alternative to the current way of publishing review tables. Apart from generating the same table, the added value comes from the ability to aggregate new (tabular) views using the same data as well as the increased FAIRness of the data published via our service. For each of the imported review tables, listed in Table 5.1, we can evaluate whether the same table can be generated with our service. For this, we have compared the table from the review paper to the table generated by the ORKG comparison service. A collection of 169 papers with 9 distinct literature views/tables is part of this evaluation. These tables can be viewed online, the links are listed in the "ORKG representation" column. The results of this evaluation are displayed in the same table, in column "Information loss". As the results show, using our service it is possible to recreate the same tabular views as originally published in the review papers.

5.4.2 FAIR Data Evaluation

As described before, with the presented service it is possible to publish a generated comparison that adheres to the FAIR principles. Because the service leverages a knowledge graph to generate and save comparisons, complying with the FAIR principles is more obvious for the ORKG comparison service than for tables in published PDF articles. In order to evaluate the FAIRness of a published comparison, we evaluate each of the four FAIR principles in detail. Wilkinson et al. [26] described each principle by assigning sub-principles.[11] We discuss the relevant sub-principles and explain how they are met. We use the term (meta)data to refer to both the actual comparison data (i.e., the data that is used to create the comparison table) and the associated metadata (e.g., the title, description, and creator of a comparison). Table 5.2 presents an overview for the evaluation of the FAIR principles.

Findable

To make data findable for both humans and machines (i.e., agents), a unique and persistent identifier should be attached to the data (F1). Additionally, metadata should describe the data (F2). In the metadata, the unique identifier of the data should be mentioned (F3). Also, a search interface should be available to find the data (F4). To ensure the findability of comparisons, users can title and describe them. Furthermore, machine-generated metadata is attached to a comparison (e.g., the number of papers and the creation date). A unique identifier is generated and attached to the data and included in the metadata. Optionally, users can assign a DOI to their comparison. Finally, the ORKG search interface allows users to search the whole graph and has a dedicated filter to specifically find comparisons. Additionally, comparisons can be indexed and found by third-party search engines (such as Google or Bing).

Accessible

Having found data, agents need to know how to access it. This principle is primarily about using accessible standardized communication protocols (A1). Additionally, metadata should be available even when the data is not (A2). The metadata is part of the knowledge graph, which can be accessed via the HTTP protocol. The data can be accessed without authentication. To support A2, the metadata and the actual comparison data are stored separately. Therefore, it is possible to access only metadata when the original data is not available anymore (for example when data is retracted by the author).

Interoperable

To ensure the interoperability of data, it should use a formal language for knowledge representation (I1) and should use vocabularies that are FAIR (I2). Finally, references or

[11] For a more detailed definition of the FAIR principles, see: `https://go-fair.org/fair-principles`

links to other (meta)data should be made (I3). As argued before, thanks to highly-structured data and the integration of shared vocabularies, interoperability is an inherent feature of knowledge graphs. Data is (partially) described using the ORKG core ontology and other ontologies we use to canonicalize the representation of relevant information content types. Links to other data are present in the knowledge graph. For example, if a comparison uses the "Web" resource to specify the domain of an application, this resource is generic, can be shared among paper descriptions and comparisons, and can be described in more detail, independently of a particular comparison.

Reusable

Finally, data should be reusable. This can be accomplished by adding relevant (meta)data (R1). Required are an accessible data license (R1.1) and detailed provenance (R1.2) data. Finally, (meta)data should use community standards to describe data (R1.3). It is possible to add additional metadata to a comparison, e.g. metadata about the scope of the comparison, which could be a reference to the paper in which the comparison is being used. The metadata is complemented with the metadata that is already part of the Findability principle, e.g. provenance data about the creator of the comparison. The data license of the graph data is CC BY-SA[12] (Attribution-ShareAlike), which allows reuse of the data. There is currently no community standard to describe the comparison data. However, standard ontologies are used to describe metadata (e.g., Dublin Core).
The evaluation of the FAIR principles shows that comparisons published with our service rank high in FAIRness, which can be even further increased with some effort from users. Users are mainly responsible for adding the correct information to the comparison and reuse vocabularies. Otherwise, findability, accessibility, and to some extent also interoperability are largely handled by the service.

5.4.3 Performance Evaluation

In order to evaluate the performance of the overall comparison, we compared the implemented ORKG approach to a naive approach for comparing multiple resources. The naive approach compares each property against all other properties to perform the property alignment. Table 5.3 shows the time needed to generate comparisons, for both the naive and the ORKG approach. In total, eight papers are compared with on average ten properties per paper. In the naive approach, the "Align contribution descriptions" step is not scaling well, since each property is compared against all others. If multiple contributions are selected, the number of property similarity checks grows exponentially. Table 5.3 shows that the ORKG approach outperforms the naive approach. The total number of papers used for the evaluation is limited to eight because the naive approach does not scale to larger sets.

[12] `https://creativecommons.org/licenses/by-sa/2.0`

Table 5.2: Overview of FAIR principles compliance.

Principle	Level	Explanation
Findable		
F1	1	Unique IDs exist, DOI assignment possible
F2	2	Machine and user generated metadata is attached
F3	1	Properties used to link data to metadata
F4	1	Comparisons are findable via a search interface
Accessible		
A1	2	Data is accessed over HTTP (via REST or a user interface), requires user effort to integrate the ORKG API specification
A1.1	1	The protocol is free and widely used
A1.2	1	No authentication is required to access the data
A2	1	Metadata is stored in a persistent way and available without the data itself
Interoperable		
I1	1	RDF (with type assertions) and CSV export of comparisons
I2	2	Reuse of ontologies where possible (ORKG core, Dublin core, RDF Data Cube Vocabulary). User responsible for other ontology reuse.
I3	3	For comparisons, the compared paper metadata is linked. More references are needed and can be created by users.
Reusable		
R1	1	Machine and user generated metadata is created while publishing
R1.1	1	CC-BY SA license
R1.2	1	If a registered user publishes a comparison, the user is associated with the published data
R1.3	2	Users can describe contributions using domain-relevant ontologies

1=Yes; 2=Yes, requires user effort; 3=Partially/future work

Table 5.3: Time (in seconds) to perform comparisons with 2-8 contributions using the naive and ORKG approaches.

	Number of compared research contributions						
	2	**3**	**4**	**5**	**6**	**7**	**8**
Naive	0.00026	0.1714	0.763	4.99	112.74	1772.8	14421
ORKG	0.0035	0.0013	0.01158	0.02	0.0206	0.0189	0.0204

5.5 Discussion

One of the aims of the contribution comparison functionality is to support literature reviews and make this activity less cumbersome and time-consuming for researchers. To live up to this aim, more structured contribution descriptions are needed. Existing scholarly knowledge graph initiatives focus primarily on scholarly metadata, while with ORKG we focus on making the actual research contributions machine-readable. Currently, the ORKG does not yet contain sufficient contribution descriptions in order for the comparison functionally to be practically useful for researchers. Furthermore, for an evaluation of the effectiveness of certain components of the methodology (such as finding related papers or aligning similar properties), more contribution data is needed. Publishing surveys does not rely on data quantity and is therefore evaluated more extensively in this work. The performance evaluation results indicate that the comparison feature performs well. This means the technical infrastructure is in place for the literature survey service.

In the evaluation, we focused on the aspects of the system that are necessary for researchers to use the system in practice. The information representation evaluation is a straightforward evaluation to see if existing survey tables can be regenerated with the ORKG. This is a minimal requirement for researchers when using the system since they should at least be able to recreate tables. This evaluation does not give insight into the usefulness and usability of the system, but still provides an indication that the service can be successfully used to publish literature surveys. One of the reasons for using the service is that also "dark data" in comparisons is published (as discussed in Section 5.1).

Another interesting aspect of the service is that published literature surveys rank high in FAIRness. Therefore, the second part of the evaluation focuses on how the FAIR principles are met. Merely publishing data as RDF is not sufficient to fully meet the FAIR principles. Hence, we conducted a more detailed evaluation that describes how the service complies with each sub-principle. Since FAIR is not a *standard*, the principles are permissive and not prescriptive [32]. No technical requirements are specified. Both the implementation and evaluation of the guidelines are therefore subject to interpretation. With respect to data interoperability and reusability, certain aspects of the service can be improved. For example, to improve interoperability, the contribution data should be reusing existing vocabularies where possible. Additionally, although most of FAIRification is done by the system, the researcher is responsible for adding correct and relevant metadata while publishing a survey.

5.6 Summary

Reviewing existing literature is an important but cumbersome and time-consuming activity. To address this problem, we presented a methodology and service that can be used to generate literature surveys from a scholarly knowledge graph, i.e., ORKG comparisons. This service can be used by researchers in order to get familiar with existing literature.

Additionally, the tool can be used to publish literature surveys in a way that they largely adhere to the FAIR data principles. The presented methodology addresses multiple aspects, including finding suitable contributions, aligning contribution descriptions, visualization and publishing. The methodology is implemented within the Open Research Knowledge Graph (ORKG). Since the comparison relies on structured scholarly knowledge, we discussed how to populate the ORKG with relevant data. This is done by extracting tabular survey data from existing literature reviews. In order to evaluate whether the proposed service can be used to publish literature surveys, the original survey table representations were compared with the ones generated by our service. As the results indicate, it is possible to use the service as an addition or potentially even replacement of the current publishing approach, since the same tables can be generated. The evaluation also showed how the published literature surveys largely adhere to the FAIR data principles. This is crucial for data reusability and machine-actionability. The proposed literature comparison service addresses multiple weaknesses of the current survey publishing approach and can be used by researchers to generate, publish and reuse literature surveys.

By introducing the literature comparison service, we addressed research question **RQ2**. We specifically focused on a method to create machine-actionable and comparable literature overviews. In the two consecutive chapters, we use the presented method from this chapter. We present a method to populate the graph with comparable data, based on existing literature reviews. Afterwards, we present an approach to using ORKG comparisons as the basis for dynamic literature review articles. **Challenge 2** is especially relevant for this chapter. Comparisons are designed both for human and machine consumption. To this end, the challenge is to make the data understandable for both humans and machines. This means that information has to be presented in such a manner that literature overviews are useful for researchers. We presented this knowledge in tabular form, with the possibility to hide irrelevant data. On the other hand, the knowledge used to create the literature comparison should be machine-actionable as well. We accomplished this by providing FAIR representations of the underlying comparison data. Furthermore, **Challenge 3** applies to this chapter as well. We made no assumptions about the research domain of comparisons. While for some domains structured literature overviews are more relevant than others (e.g., structured overviews are frequently used in Computer Science but less in Philosophy), the comparison service itself is not tied to a specific domain.

CHAPTER 6

Creating a Scholarly Knowledge Graph from Survey Article Tables

The Open Research Knowledge Graph (ORKG) [51] aims to build a knowledge graph infrastructure that publishes the research contributions of scholarly publications rather than only the metadata. Existing initiatives for scholarly information systems, e.g., the Microsoft Academic Graph [155] or Crossref [141] mainly focus on bibliographic metadata and not on the actual research contributions. ORKG primarily relies on synergistically combining crowdsourcing and automated extraction rather than, as other systems such as Semantic Scholar[1], exclusively on automated techniques to extract knowledge from scholarly articles. Mainly because automated extraction methods, for example, Natural Language Processing (NLP), do not have sufficient accuracy to generate the high-quality knowledge graph needed to obtain suitable state-of-the-art overviews for researchers. However, to convince users to contribute content to the ORKG, we need to establish an initial set of high-quality structured data to highlight the potential of the ORKG.

In this chapter, we present a human-in-the-loop methodology to populate a scholarly knowledge graph by extracting knowledge from survey tables. We leverage survey tables from literature review papers, specifically. Tables in survey papers generally consist of high-quality research data that has been manually curated by domain experts. Conducting a literature review is a labor-intensive task and writing a review article is often more time-consuming than writing a research article [75]. Compared to natural text, tables present information in a semi-structured manner, making the creation of a structured graph from such data less complicated. Additionally, survey tables present relevant information which is why the survey was conducted and published in the first place. We present a supervised approach to firstly extract data from survey articles and afterwards build a knowledge graph

[1] https://www.semanticscholar.org

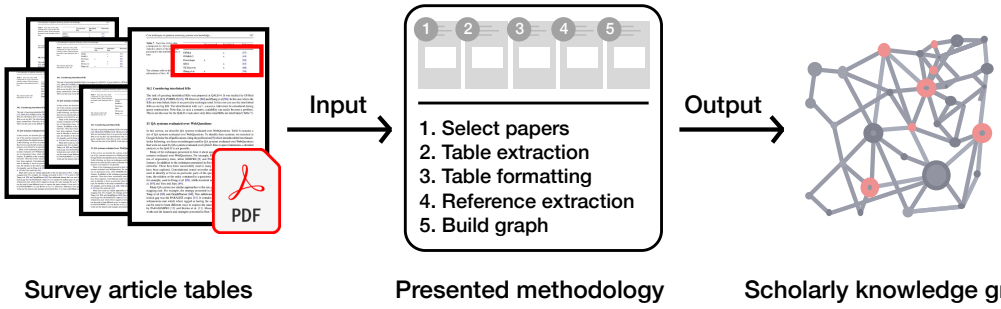

Figure 6.1: Systematic workflow in which survey articles are used to build a scholarly knowledge graph. The input of our methodology is survey articles in PDF format and the output is a scholarly knowledge graph.

from this data. Compared to sole crowdsourcing, the approach of extracting knowledge is more efficient because the review has already been conducted by the authors of the survey paper.

This chapter is related to the following research question:

> **RQ2:** How to generate machine-actionable and comparable overviews of related literature?

Machine-actionable overviews of literature can be generated once a scholarly knowledge graph is generated. With the research question, we also address the following sub-question: *How to efficiently populate a scholarly knowledge graph with high-quality knowledge?* We propose a methodology for extracting tabular survey data. This methodology is used to create a scholarly knowledge graph from survey articles. An overview of the systematic workflow is depicted in Figure 6.1.

This chapter is based on the following publication: [123]. The rest of this chapter is structured as follows. Section 6.1 explains the rationale of using the ORKG. Section 6.2 introduces the proposed five-step methodology for building the knowledge graph from survey articles. Section 6.3 presents the results. Section 6.4 discusses the presented work. Finally, Section 6.5 concludes the presented work.

6.1 Use Case: Open Research Knowledge Graph.

Extracted survey data can be imported in a variety of different (scholarly) knowledge graphs, such as the Microsoft Academic Graph, Wikidata [40] or ORKG. We chose ORKG as our use case for the following reasons. The ORKG provides tools that specifically focus on building paper comparisons (as described in Chapter 5), making it the most suitable infrastructure for this study. By using the extracted survey data, the ORKG automatically generates a similar tabular survey view as was originally presented in the review paper [150]. Additionally,

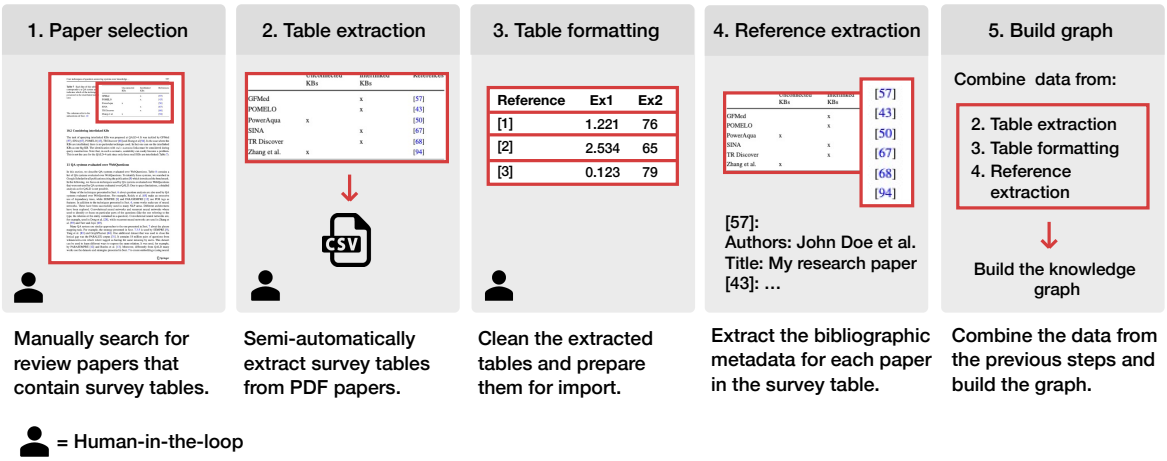

Figure 6.2: Methodology for importing survey tables into the scholarly knowledge graph.

the literature surveys within ORKG are compliant [78] with the FAIR data principles [26] thus making them Findable, Accessible, Interoperable, and Reusable. The imported survey tables are FAIR in contrast to the originally presented ones in the non-FAIR PDF article. This has several benefits, among others:

- Comparisons can evolve over time, are not static, and do not become stale after publication.
- Comparisons do represent a broader community consensus since many researchers and curators can revise, discuss and annotate.
- Via the ORKG search interface it is possible to search for specific comparisons and to create dynamic custom comparison views.
- Survey data can be reused by other researchers more easily because of its machine-readable export formats (e.g., export as CSV or RDF).

6.2 Methodology

We now present a five-step methodology for the creation of a scholarly knowledge graph from survey tables. In order to reach sufficient quality, the methodology takes a human-in-the-loop approach in which multiple steps require human interaction. Data quality improves with human evaluation and, if needed, correction of the extracted data. The methodology is displayed in Figure 6.2. The scripts required to perform the steps are available online.[2]

[2] https://doi.org/10.5281/zenodo.3739427

Table 6.1: Search engines used to find survey articles.

Search engine	Field	Evaluated papers
Google Scholar	All	335
ACM Digital Library	Computer Science	80

6.2.1 Paper Selection

In the first step, suitable survey papers are selected based on multiple criteria. The purpose is to find survey papers from a diverse range of domains. Therefore, a protocol has been designed to determine which papers are suitable for data extraction. The structured nature of the selection process is needed to be able to make conclusions about the percentage of survey papers that present the information in such a way that extracting data is relatively straightforward.

Search Strategy.

Table 6.1 lists the search engines used to find survey articles. Google Scholar is chosen to ensure that survey papers from various fields are searched. Additionally, ACM Digital Library has been selected because the ORKG currently focuses mainly on the Computer Science domain. The search is limited to 100 papers that are suitable for import. The following search criteria are used:

- Google Scholar: the article title contains the term “literature survey".
- ACM Digital Library: queries “literature review" and “literature survey".
- The survey article has been published after 2002.
- The results are sorted by relevance.

The rationale for selecting papers published after the year 2002 is because in general more recent papers are more interesting for research and should therefore have more priority in the scholarly knowledge graph. In the end, articles published before 2002 can still be part of the graph, since this criterion only applies to the survey articles themselves, and not to the papers being reviewed in those articles.

Selection Criteria.

Papers that satisfy the inclusion criteria are selected for the import process. The inclusion criteria are defined as follows:

1. The article contains at least one table that lists scientific literature (i.e., the literature is presented in a semi-structured manner).

2. The article compares literature based on published results and does not solely textually summarize the content of original papers.

3. The survey table should be in markup format and not included as raster image.

4. The table structure should be suitable for import (e.g., one table row should provide information about one publication).

5. The article is written in English.

Inclusion criterion 1 ensures that a survey article does not only textually summarize the literature, but does also provide a semi-structured comparison (in tabular form). Although papers that are textually reviewing scientific literature are interesting for importing as well, it is out of scope for this work. Criterion 2 ensures only surveys that compare actual paper results are included. This excludes surveys researching, for instance, the growth of a field. Criterion 3 excludes tables in image format. This is because of the tabular extraction method we use, which is based on character extraction and does not use Optical Character Recognition (OCR) needed to support image extraction [156]. Criterion 4 only selects tables that are suitable for import. Our methodology does only support paper import when one row in a table represents one paper. Although minor changes can be made manually (e.g. merging multiple tables), in case the structure of the table deviates significantly from the required format, the table is excluded. Finally, criterion 5 ensures a homogeneous semantic integration into the currently English monolingual knowledge graph. The result of this step is a set of the selected papers in PDF format.

6.2.2 Table Extraction

This step focuses on extracting the tables from the PDF files collected in the previous step. Not only the text within the table should be extracted, but the tabular structure should be preserved as well. As explained in the related work section, we use Tabula to perform the table extraction. Each PDF article is uploaded via the Tabula user interface. Afterwards, the regions of the tables are manually selected within the interface. Although Tabula provides a functionality to automatically detect tables, the accuracy is not sufficient for our use case. The performance is especially low for articles with a two-column layout. Additionally, not all tables within an article have to be extracted since not all of them are listing and comparing literature. Arguably, the manual selection method is most useful in this methodology since human judgment is needed in the selection process. Part of the extraction step is quality assurance after the extraction. When needed, extraction errors are manually fixed. Tabula supports two types of extraction, namely “Stream” and “Lattice”. The Stream extraction method is based on white space between columns while Lattice is based on boundary lines between columns. During the extraction it is possible to switch

between the different methods, which allows for selecting the best method for a particular table. The result of this step is a set of CSV files, in which each file represents one survey table from a review article.

6.2.3 Table Formatting

The CSV files containing the extracted tables from the review articles should be formatted in a structure that is suitable for building a graph. Since the data from the CSV file is extracted automatically, all tables should have the same format. In this step, the formatting of the tables is changed when necessary. For some tables, a considerable amount of changes is required while for other tables only minor changes are needed. Changes could include merging, splitting, adding, and removing both columns and rows. We use OpenRefine [157] to perform bulk operations on tables. A table is formatted in such a way that it adheres to the following rules:

1. The first row of the table is the header.
2. Each row represents one reviewed paper.
3. Each row has a column called: “Reference".
4. The reference cell should contain the citation key for a paper .
5. Non-literal values are prefixed with “[R]" in the column header.
6. When needed, abbreviations are replaced by the full value from the legend.

For rule 2, in some cases a multidimensional table has to be flattened. This can often be accomplished by adding additional columns to the table. Also, in some cases a table has to be transposed to ensure that each row contains one paper. Rules 3 and 4 ensure that bibliographic metadata can be fetched for each paper in the next step. Rule 5 makes a distinction between literal values and resources. The default cell type is a literal, and when [R] is prefixed to a header label, the cells are considered as resources. Finally, rule 6 makes the content of the table readable without requiring the original text from the legend. Often table legends are used to condense information to improve user readability.

6.2.4 Extracting References

As mentioned earlier, each table row represents one paper. For each row, there is a value that contains the reference key from the original paper. The reference key is often a numerical reference, in the form of $[n]$, where n represents the reference number. In another frequently used citation style, the author names combined with their publication year is used as a reference key. The citation key is used to automatically capture the bibliographic metadata

Study	Location	Study date	Methods	R_0 estimates	95% CI
Joseph et al.[1]	Wuhan	31 Dec '19 - 28 Jan '20	Stochastic Markov Chain...	2.68	2.47-2.86
Shen et al.[2]	Hubei province	12-22 Jan. '20	Mathematical model, dynamic...	6.49	6.31-6.66
Liu et al.[3]	China and overseas	23. Jan '20	Statistical exponential Growth...	2.90	2.32-3.63

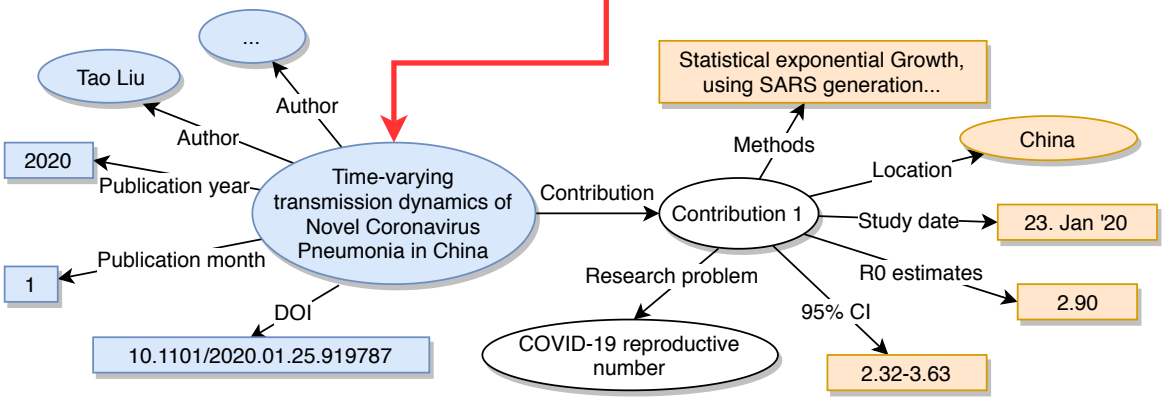

Figure 6.3: Example of the resulting subgraph for importing a single paper from a survey table. Metadata captured by reference extraction is displayed in blue. Data coming from the survey table is displayed in orange and ORKG specific data is displayed in white.

for an article. In order to extract references from article, we use the PDF extraction tool GROBID. GROBID processes the full PDF article. In the first place to extract all citations from the paper's reference list and then to connect the citation keys used in the text to their respective citation string. In case a reference key cannot be extracted from the paper's text, a reference key is generated automatically based on the author's name and publication year.

When the citation is extracted and parsed, five additional columns are appended to the table: paper title, authors, publication month, publication year, and the DOI[3]. In case a citation key could not be automatically mapped to an actual citation, a citation can be provided manually. The full citation text can be copied directly from the paper (including paper title, authors etc.) and is then parsed by GROBID to get structured bibliographic metadata. To perform the process of adding references, we created a Python script.[4] This script first tries to automatically fetch the metadata. In case the reference is not found, a command line input field is displayed to enter the citation manually.

6.2.5 Build Graph

The final step is to build a knowledge graph from the previously created CSV files. An example of the resulting graph for a single paper is depicted in Figure 6.3. Firstly, a settings file is created which lists the table numbers, a suitable title for the table, and a reference

[3] Digital Object Identifier
[4] File `4_reference_extraction.py` from `https://doi.org/10.5281/zenodo.3739427`

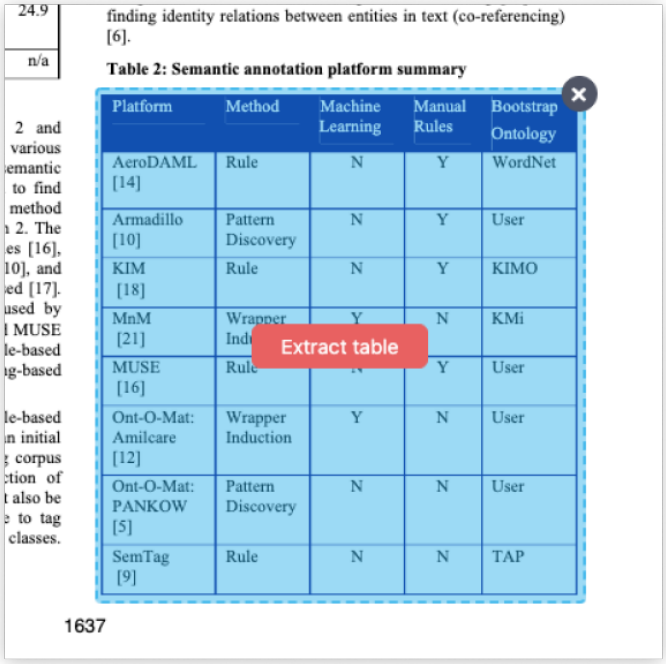

Figure 6.4: Select and extract table from PDF.

to the original survey article. The reference is required to attribute the work done by the authors of the survey article. The table title is manually created based on the original table caption. In case no suitable caption is available, a more suitable title is written.

Next, a Python script[5] is used to select all rows from the tables. For each row, a paper is added to the graph via the ORKG API. For each table, a comparison is created in ORKG. The title and reference from the previously generated settings file are attached to this comparison. The comparison can be used later in ORKG to generate the same tabular literature overview as originally presented in the survey paper.

6.2.6 User Interface

Based on the steps from our methodology, a web User Interace (UI) is created that integrates all steps into a single interface. The interface provides a streamlined process for importing survey tables as depicted in Figure 6.4 and Figure 6.5. The UI is specifically designed to make importing a table an effortless task without the need of downloading any tools or the need to be able to operate these tools. In the background, the same tools from the methodology are used to extract tables (Tabula) and extract references (GROBID). The first step is to upload a PDF file and select the survey table within this file. Afterwards, the table

[5] File `5_build_graph.py` from `https://doi.org/10.5281/zenodo.3739427`

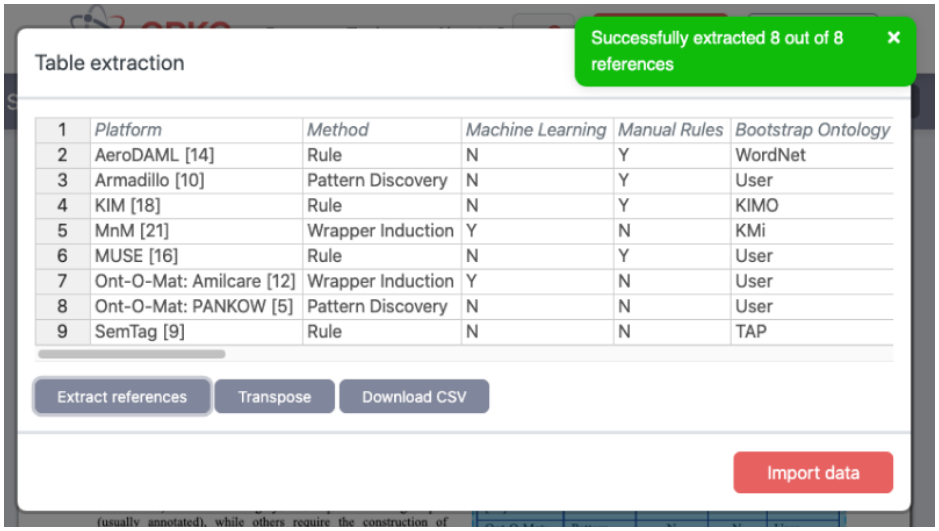

Figure 6.5: Fix table formatting, add references and ingest in graph.

is extracted and the formatting can be fixed with an integrated spreadsheet editor. Then, for each row, the respective paper reference is extracted. Finally, the data is ingested in the knowledge graph.

The UI is not used to import the surveys tables presented in Section 6.3. The interface is designed to import individual survey tables rather than importing large amounts of tables at once. In the UI, all steps required to import a single table should be performed consecutively. To increase efficiency when importing large amounts of tables, it helps to first finish a step for all papers before moving to the next step. The UI provides a method to extend the graph beyond the extracted surveys from this work. In the future, this interface will therefore be integrated in the ORKG.

6.3 Results

In this section, we report the results of the import process for each step of the methodology. Table 6.2 summarizes the results for all steps.

6.3.1 Paper Selection

The dataset of the results is published online [158]. This set contains the selected papers, the ORKG comparisons, and the ingested papers. The selected papers file lists IDs, paper titles, table references, sources, and references. The IDs are used to record any additional information about the import process for this specific paper. IDs are missing for papers that were selected in the first place but were excluded after revising the inclusion criteria. Additionally, table references refer to the original table references used in the survey article.

Table 6.2: Summary of the results of all steps.

Description	**Amount**
Paper selection	
Amount of evaluated papers	415
Amount of selected papers	92
Table extraction	
Total amount of extractions (partial tables)	265
Amount of extracted complete tables	160
Reference extraction	
Found references	2 069
Not found references	1 137
Build graph	
Individual amount of imported papers	2 626
Imported data cells (with metadata)	40 584
Imported data cells (without metadata)	21 240

In total, 335 papers from Google Scholar were evaluated against the selection criteria described in Section 6.2.1. Out of these papers, 78 met the criteria and have therefore been selected for importing. From the ACM Digital Library 80 papers were evaluated and 14 papers have been selected. In total 22% of the evaluated review papers are suitable to be imported with the presented approach.

6.3.2 Table Extraction

We extracted 160 tables from the 92 survey articles. In 22 cases, tables stretched across multiple pages, which results in a total of 265 extractions performed with Tabula. Table 6.3 lists the most frequently occurred issues with the extraction. Issue 1 and 2 occur mostly when no boundary lines are present between table columns. In this case, the Stream extraction method has to be used, which often results in rows that are not correctly merged (e.g., multi-line sentences are put in separate rows while in the original table they are in the same row). Also, issue 3 is mostly present when using the Stream method. When the Lattice method can be used for the extraction, the result is generally of higher quality. When no table borders (or boundary lines) are present, this method does not work and the Stream method has to be used. Issue 4 is caused by general extraction errors, which can result in

Table 6.3: Issues that occurred during the extraction of tables from the survey articles. Issues are counted per article.

#	Issue	Percentage %
1	Columns are not extracted correctly	26
2	Rows are not extracted correctly	14
3	Empty columns in the extracted table	14
4	Text not correctly recognized (e.g., missing letters or formulas)	12
5	Issue with table header text	12
6	Vertical text not imported correctly	4
7	Cell value not supported (e.g., use of image instead of text check marks)	3
8	Table within table not extracted correctly	3

tables with wrongly extracted text. Additionally, formulas and other text styling are not supported, which compounds this issue. Issues 7 and 8 result in tables that are not, or only partially, imported. The other issues are self-explanatory.

6.3.3 Reference Extraction

In total, we extracted unique 2 626 papers from 3 206 rows. For each paper, the respective citation was retrieved. In 2 069 cases the citation could be extracted automatically from the row (65% of the cases). In 1 137 cases it was not possible to automatically extract the reference (35% of the cases). For those cases, the citation is manually copied from the paper. There were multiple reasons why automatic reference extraction was not successful. Most issues occurred for references that used a numeric citation key. GROBID's performance for extracting numeric references from tables was low, oftentimes numeric table references were not recognized. The amount of rows is higher than the amount of extracted papers because multiple rows could refer to the same paper. Each paper only has one graph entry and any additional data is added to the existing paper.

In case a reference is only used in a table and not somewhere else in the article, automatic reference extraction was oftentimes not possible. When an author name was used as citation key, problems occurred mostly because of the different citation styles. While some citation formats only use the last name of the first author, suffixed by *et al.*, other formats could list all author names. When a format was used that deviates from the standard implementation, automatic extraction was not possible.

6.3.4 Build Graph

In total, we added 2 626 papers to the knowledge graph. These papers are used in 160 different comparisons. A complete list of the generated ORKG comparisons and a list of all ingested papers is available via [158]. In total, 21 240 table cells have been imported, excluding the bibliographic metadata. Including metadata, the total is 40 584 data cells.

6.4 Discussion

6.4.1 Time Performance

The presented methodology takes a human-in-the-loop approach as opposed to a fully automated approach. Compared to a fully manual approach, the proposed approach saves considerable time. In previous work [78], we manually imported only four survey articles. On average, this process took 4 hours per article. For each of the papers, a Python script was created specifically to import the survey table with its references and data. An example of such a script for one paper can be found online.[6] For the methodology used in this chapter, the time to import one survey article was on average 15 minutes. Compared to the 4 hours of the manual approach, this is considerably faster (i.e., 16 fold increase in speed). The minimum amount of time needed to import a relatively small table was 2 minutes. The table could be extracted without any issues. The maximum amount of required time was approximately 60 minutes. This was for a table with a complex layout, stretched across multiple pages. Also, this table did not have boundary lines. Most time was spent on fixing extraction issues. To further improve time performance, we identified two tasks that are time-consuming and can potentially be improved. The first task relates to fixing errors that occurred during the table extraction by Tabula. Most errors occurred when tables did not have boundary lines between columns and rows. A potential solution is to create an interface that supports manually drawing boundary lines between rows and columns. The second task is related to adding missing references, which have to be manually copied from the PDF article. In total, 65% of the references were extracted automatically. By applying more advanced heuristics to match reference keys with their respective reference, this percentage can be improved.

6.4.2 Impact of Methodology

The impact of the methodology relates to the number of survey papers that are suitable for our approach (i.e., surveys representing information in tabular format). To order to provide insights on the impact, a structured search protocol has been employed in the paper selection step. As the results show, out of the 415 evaluated papers, 92 of them are suitable

[6] `https://gitlab.com/TIBHannover/orkg/orkg-papers/-/blob/master/question-answering-import.py`

to be imported. This indicates that since 2002, 22% of the published survey papers contain comparison tables. Therefore, arguably our methodology can have considerable impact when applied more broadly. In the paper selection, non-survey papers were excluded. However, it is not uncommon for research articles to also contain tables with related work. Thus the paper selection step could be extended to also include other articles to have a broader impact.

6.4.3 Semantics of Data

The extracted knowledge graph consists of structured scholarly data. The quality of the knowledge graph could be further improved by providing more semantics to the data. Currently, a primitive method is used to map existing properties and resources. This is based on a lookup by resource label, in case a result is found, the resource is mapped. If not, a new resource is created. A more advanced mapping of resources and properties to existing ontologies improves the machine readability of the data. Tables containing large amounts of natural text (e.g., textually describing a methodology) could be further processed using named entity recognition and linking. This results in more structured data and therefore a higher quality knowledge graph.

Digital Library for Comparisons

In total, we extracted 92 survey articles which were used to generate comparisons. In the future, we envision the ORKG to contain many more comparisons for a variety of domains. The User Interface (UI) presented in subsection 6.2.6 can be used to support users to import survey tables and to further increase the number of comparisons. Due to the dynamic nature of the interface (especially compared to a regular spreadsheet editor), mapping properties and resources to existing concepts is better supported. In the end, we aim to import as many surveys from a specific domain as possible. The imported surveys serve as starting point for more comprehensive comparisons. There are several reasons why such an approach is useful. In the first place, ORKG can serve as a digital library for literature overviews. As discussed in the related work, the platform provides tools to better find and organize surveys. Additionally, when all existing reviews for a domain are imported, the ORKG can be used as a source to find literature surveys. In case a survey is not present in the ORKG, it means that it does not (yet) exist. This can be used as a basis to start working on new literature overviews.

6.5 Summary

Knowledge graphs are useful to make scholarly knowledge more machine-actionable. Manually building such a knowledge graph is time-consuming and requires the expertise of paper authors and domain experts. In order to efficiently build a high-quality scholarly

knowledge graph, we leverage survey tables from review articles. Generally, survey tables contain high-quality, relevant, semi-structured, and manually curated data, and are therefore an excellent source for building a scholarly knowledge graph. We presented a methodology used to extract 2 626 papers from 92 survey articles. The methodology adopts a human-in-the-loop approach to ensure the quality and usefulness of the extracted data. Compared to manually reviewing and entering research data, or to manually importing literature surveys, the methodology is considerably more efficient. In conclusion, the presented methodology provides a full pipeline that can be used to extract knowledge from PDF documents and represent the extracted knowledge in a knowledge graph. The corresponding evaluation with survey articles demonstrates the effectiveness and efficiency of the proposed methodology.

This work addressed **RQ2** by providing the data needed to generate structured and machine-actionable literature overviews. It specifically relates to **Challenge 1** which is the conversion from unstructured to structured knowledge. By leveraging existing semi-structured knowledge, in the form of survey tables, we created a methodology that helps efficiently to transform unstructured to structured knowledge. Furthermore, **Challenge 2** is relevant to our approach. Also when transforming tables to graphs, there is a trade-off between human- and machine-actionability. In our approach, we tried to make the steps as simple as possible for users. This means that the resulting graph contains mostly star-shaped patterns instead of complex nested descriptions. While this is beneficial for users importing survey tables, it potentially hinders machine-actionability which benefits from these complex descriptions.

CHAPTER 7

SmartReviews: Towards Human- and Machine-actionable Representation of Review Articles

One of the major factors that hinder effective information organization relates to the inability of machines to parse and understand the concepts expressed in scholarly articles [146]. Generally, articles are authored specifically for human readers, without taking into account the machine-actionability of the presented work [7]. Traditionally, review (or survey) articles are used to organize information for a particular research domain [71]. Research articles, also referred to as primary sources, present original research contributions. Review articles, or secondary sources, organize the research presented in the primary sources [148]. The development of research fields is weakened when reviews are lacking [75]. The importance of review articles becomes apparent in the fact that these articles are often highly cited [72] which indicates that they are valuable to the community. Although reviews are important, they suffer from several major weaknesses, which affect the potential impact review articles can have. The weaknesses can be grouped into two categories, namely their "representational format" and the "publishing method". The first category relates to the static nature of articles. For example, once review articles are published, they are generally not updated when new research articles become available. This results in review articles that are outdated soon after they are published. Two additional weaknesses are lacking community collaboration and limited coverage of research domains. The second category relates to the format in which articles are published. Weaknesses include lacking machine-actionability and lacking accessibility and a missing overarching systematic representation of research concepts. The weaknesses are discussed in more detail in subsection 3.2.1.

In this chapter, we present SmartReviews, a novel tool to author and publish review articles. SmartReview address the weaknesses from which current review articles are suffering. Reviews are authored in a community-based manner and are represented as living docu-

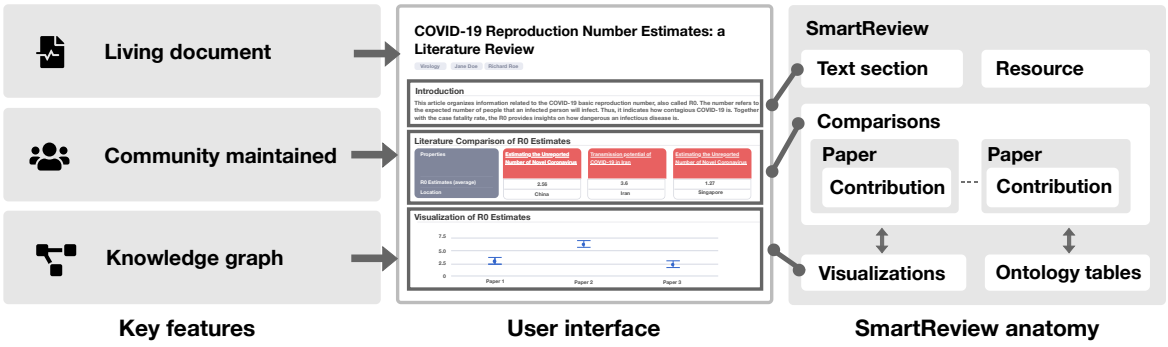

Figure 7.1: Illustration of key features and anatomy of SmartReviews. They are composed of several building blocks, including natural text, comparisons, and visualizations.

ments, meaning that they can be updated whenever deemed necessary by the community. SmartReviews are powered by a knowledge graph that supports semantic descriptions of the presented knowledge. They are implemented within the Open Research Knowledge Graph (ORKG) reusing the graph data and several components, such as comparisons and visualizations. The key features and anatomy of SmartReviews are depicted in Figure 7.1. In summary, this chapter provides the following research contributions:

(i) Detailed description of authoring and publishing semantic review articles using knowledge graphs.

(ii) Implementation of SmartReview authoring tool.

(iii) Presentation and evaluation of the representational aspects of an original SmartReview article.

The following research question is addressed:

> **RQ2:** How to generate machine-actionable and comparable overviews of related literature?

SmartReviews revolve around ORKG comparisons and they form a key component of the approach. By leveraging the existing comparison approach, as presented in Chapter 5, SmartReviews contain machine-actionable overviews of literature. Those overviews are complimented with additional content, such as natural text and visualizations.

This chapter is based on the following publications: [159, 160]. In addition to these publications, this chapter presents a more in-depth technical implementation and evaluation. The remainder of this chapter is structured as follows. Section 7.1 outlines the approach and its requirements. Section 7.2 presents the implementation based on the approach. Section 7.3 evaluates the approach, focusing on the representational aspects of SmartReviews. Finally, we discuss and conclude the work in Section 7.4 and Section 7.5 respectively.

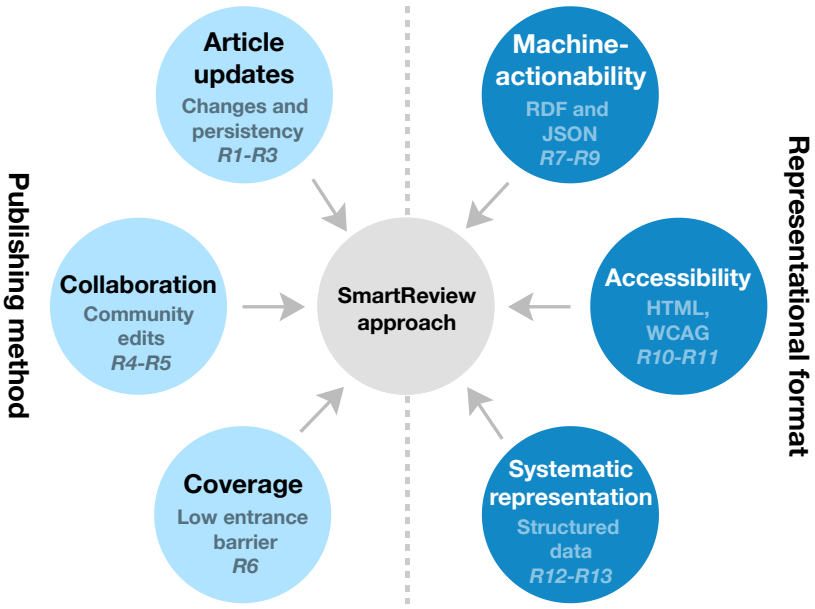

Figure 7.2: The SmartReview approach which consists of six dimensions with a brief description of the related requirements for this dimension.

7.1 Approach

Our approach addresses the previously mentioned weaknesses. Accordingly, we introduce dimensions to address each weakness individually. Figure 7.2 summarizes and categorized into two distinct groups the six definitorial dimensions. The "Publishing method" category lists dimensions related to the method of publishing review (i.e., the decoupling of writing and publishing articles). The "Representational format" category relates to the format in which articles are communicated (i.e., PDF files).

The ORKG is leveraged at the core of our approach. The use of knowledge graphs enables the reuse of existing ontologies, thus improving the machine-actionability of the data. To this end, the article has to be represented in a structured and semantic manner. The ontology of SmartReviews is depicted in Figure 7.3, it additionally shows the relations between entities within the ORKG. Research articles are generally composed of multiple (non-structured) artifacts, among others this includes natural text sections, figures, tables, and equations. Review articles, in particular, do often include an additional artifact in the form of comparison tables. These tables present the reviewed work in a structured manner and compare the work based on a set of predefined properties. A previous study indicated that approximately one out of five review articles contains such tables [123]. Due to the structured nature of comparison tables, they can be processed more easily by machines. Complemented with semantic descriptions of the data, the comparisons can become FAIR

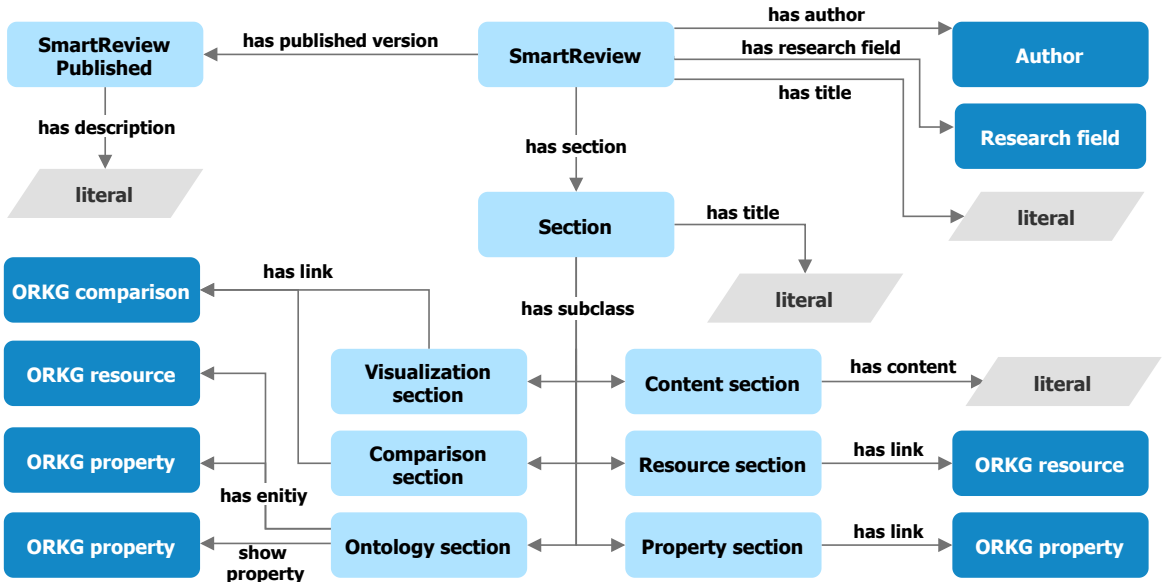

Figure 7.3: Simplified illustration of the SmartReview ontology. Dark blue classes refer to existing ORKG concepts while light blue classes are specific to SmartReviews.

data [78]. Therefore, we use comparison tables as the basis of our SmartReview approach. We leverage the comparisons tables within the ORKG which are specifically designed to be machine-actionable.

The approach leverages the SmartReview requirements as presented in [159]. Requirements are formulated using the FunctionalMASTeR template [161].

Article Updates

It should be possible to update review articles once published, resulting in “living" documents [101]. The individual versions should be citable and it should be clear which version of the article is cited. Additionally, readers should be able to see which parts of the articles have changed across versions. Based on these criteria, we formulate the following requirements:

R1 SmartReviews shall provide researchers the ability to update articles.

R2 SmartReviews shall persist all versions of published articles.

R3 SmartReviews shall provide researchers with the ability to compare different versions of the same article (i.e., diff view).

Collaboration

To fully support community collaboration for review articles, they should be editable by anyone within the community. To ensure no work is getting lost (e.g., removed by another author), it should be possible to go back in time and compare different versions (as described in R3).

R4 SmartReviews shall provide any researcher with the ability to contribute to articles.

R5 SmartReviews shall list all contributors in the acknowledgments.

Coverage

To increase the review coverage for less popular domains, the entry barrier for creating and updating SmartReviews should be low (related to R4). SmartReviews can be created even if only a limited amount of articles are reviewed. This is achieved by decoupling publishing (i.e., peer-reviewed publishing in a journal or conference) and authoring of articles.

R6 SmartReviews shall provide researchers with the ability to create articles without the need for an a priori peer review.

Machine-actionability

In order to improve machine-actionability, a systematic and structured representation in a knowledge graph should be used for knowledge representation. The resources defined within the knowledge graph serve as building blocks to create the article. This structured data is supplemented by natural text sections. To improve machine-actionability, natural text sections are complemented by types describing their contents.

R7 SmartReviews shall use a knowledge graph as data source for articles.

R8 SmartReviews shall semantically type and structure natural language text sections.

R9 SmartReviews shall provide machine-actionable formats (i.e., RDF, JSON-LD).

Accessibility

Most accessibility issues originate from the PDF format in which most articles are published. By publishing the articles in HTML instead, the article is already more accessible. Furthermore, by adhering to the Web Content Accessibility Guidelines (WCAG) [162], the accessibility is further improved.

R10 SmartReviews shall publish articles in HTML format.

R11 SmartReviews shall follow WCAG guidelines.

Systematic Representation

Review articles often use tabular representations for comparing research contributions from different articles. SmartReviews should focus on these comparison tables, and encourage researchers to use these tables to devise a structured description of the reviewed articles.

R12 SmartReviews shall devise a structured comparison of reviewed work.

R13 SmartReviews shall support linking existing resources and properties from the knowledge graph.

7.2 Implementation

The main building blocks of SmartReviews are sections. Each section has a section type that describes the section's content and its relation to the knowledge graph. The article writer has been implemented on top of the ORKG which allows for reusing artifacts already present in the graph. When adding a section, the type can be selected (Figure 7.4, node 6). The section types comprise:

- **Natural language text** sections support markup text via a Markdown editor. References are supported via a custom syntax using the same notation as R Markdown [163].
- **Comparison** sections consist of tabular overviews of scientific contributions from papers that are being compared based on a selected set of properties. Comparison sections form the core of each review article.
- **Visualization** sections provide visual views of comparison data.
- **Ontology table** sections list descriptions of the properties and resources used within comparisons.
- **Resource and property** sections show a tabular representation of used resources and properties and their definitions from the knowledge graph.

The SmartReview interface is implemented in JavaScript using the React framework, the source code is available online[1]. Additionally, a feature demonstration video is available.[2]

[1] `https://gitlab.com/TIBHannover/orkg/orkg-frontend/-/tree/master/src/components/SmartReview`

[2] `https://doi.org/10.5446/53601`

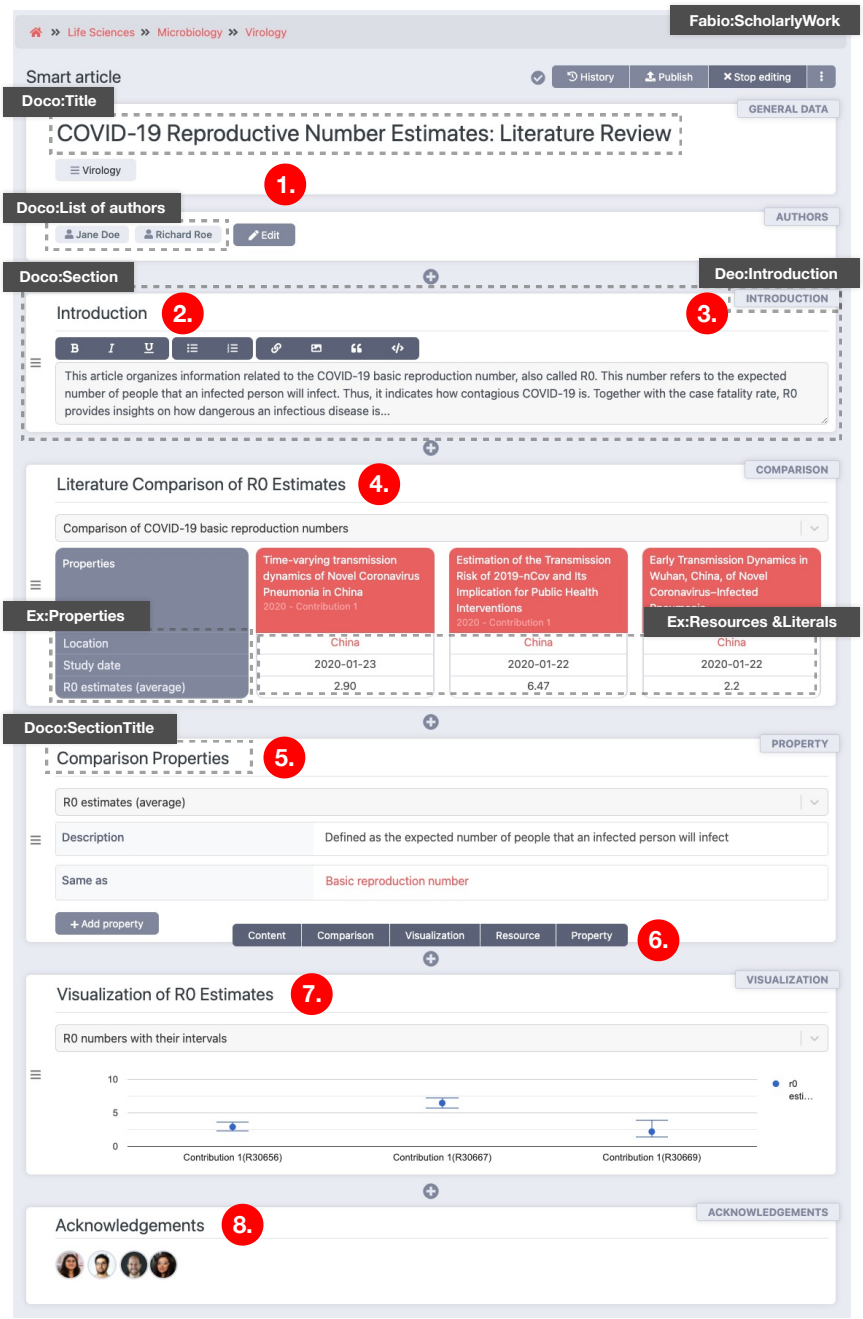

Figure 7.4: Screenshot of the implemented interface. Black labels represent the RDF types. Types prefixed with "Ex" are from the scholarly graph used for the implementation. Node 1 relates to the metadata of the article. Node 2 is the natural text content section and its Markdown editor. Node 3 shows the DEO type, which can be selected by the users when clicking on the label. Node 4 is a comparison section and node 5 is a property section. Node 6 shows the type selection for a new section. Node 7 is the visualization of the comparison shown in node 4. Finally, node 8 lists the contributors of the article.

Implementation per SmartReview dimension

The implementation consists of various components to provide an integrated and comprehensive authoring interface. Among other things, this includes support for in-text citations, an interactive article outline, and reading time estimation. These features are ordinary functionalities for authoring tools and are therefore not discussed in detail. In the remainder of this section, we specifically focus on the dimensions of the SmartReview approach since they form the basis of the implementation.

Article Updates

The requirement of updating articles (R1) combined with the requirement to keep persistent versions of articles (R2) introduces a level of versioning complexity. Especially due to the interlinking of knowledge graph resources, persistency is a complex endeavor that requires versioning at the resource level. To reduce the complexity, we added the constraint that only the latest version of an article can be updated, which we call the *head version*. The head version is the only version that is stored in the graph (Figure 7.5). This implies that always the latest version of the article is present in the graph, leaving version complexity outside the graph and thus making it easier to interact with the graph. When an article is published within the system (not to be confused with publishing the article via a publisher), a snapshot is created of the subgraph used to generate the article. This subgraph is serialized in JSON format and is stored in a separate document-based data store. This data store serves as a *create and read-only* store, preventing updates and thus providing persistent storage. The snapshot is accompanied by a user-provided update message, indicating why the change has been made (similar to a commit message in Git versioning). This approach resembles that of other collaborative curation systems (such as Wikipedia) that only allow edits of the latest version and keep a persistent history of all versions. Crucial for this approach is the ability to compare previous versions and to track individual changes (i.e., the diff view). Because articles can consist of various dynamic elements, we first convert the article's contents to natural text and then perform the diff operations. When a new version is published, for all previous versions a message is displayed indicating that a newer version of the article is available.

Collaboration

Collaboration is supported by allowing edits from any user. As with the article updates, this resembles the approach Wikipedia takes to support collaborative authoring. In Wikipedia, this has resulted in high-quality articles, which is popularly explained by the "wisdom of the crowd" principle [164]. To acknowledge researchers who contributed to the article, and to create an incentive to contribute, the acknowledgments section automatically lists anyone involved in writing the article (Figure 7.4, node 8). The method of acknowledging contributors is similar to open source projects, which often list all source code contributors to

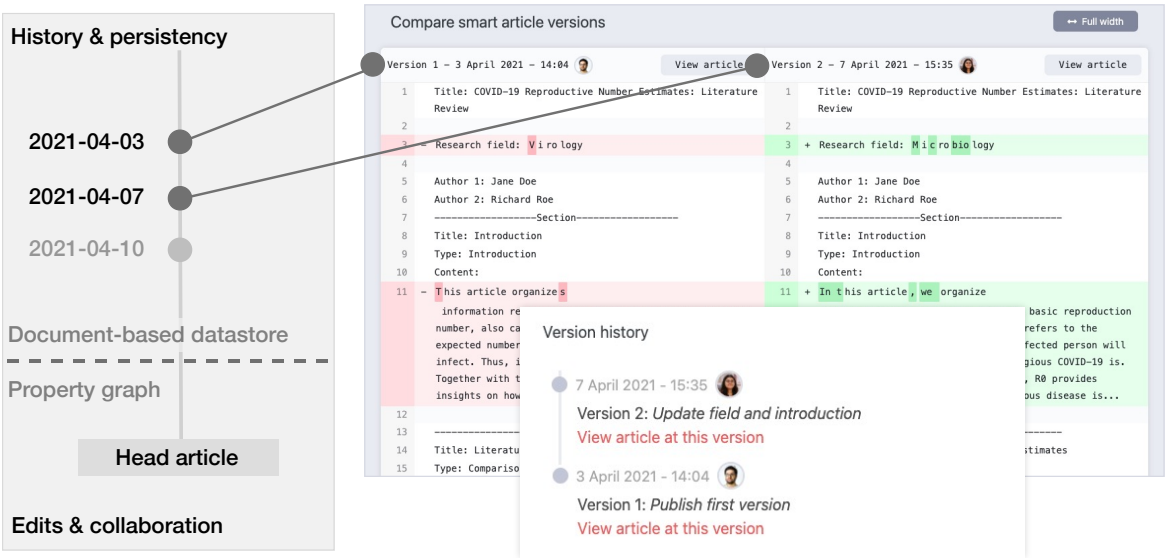

Figure 7.5: Overview of technical implementation to support article updates. The history and persistency features are supported by a document-based database. The head article is stored within a knowledge graph.

give them credit and to provide incentives to contribute [165]. The list of acknowledgments is generated by traversing the article subgraph. Based on the stored provenance data, the corresponding contributors are determined.

Coverage

The only prerequisite to be able to contribute to an article is the need for a user account. Authentication serves for tracking provenance data (needed for the acknowledgments) and as a basic abuse prevention system.

Machine-actionability

As described, the article content is available in the knowledge graph. The data itself can be accessed via various methods, including a SPARQL endpoint, RDF dump, and REST interface. To enhance machine interoperability, (scholarly) publishing ontologies were used. In Figure 7.4, RDF types prefixed with their respective ontologies are displayed next to system components. This includes the Document Components Ontology (DOCO)[3] to describe documents components. The FRBR-aligned Bibliographic Ontology (Fabio)[3] to describe the types of published work and the Discourse Elements Ontology (DEO)[3] ontology

[3] `http://purl.org/spar/{doco,fabio,deo}`

for specifying the section types. For the latter ontology, the article authors are responsible to select the appropriate type from a list of all DEO types for natural text sections (Figure 7.4, node 3).

Accessibility

Review articles are available as HTML files, which makes them by design more accessible than their PDF counterpart. Furthermore, WCAG guidelines are followed to enhance accessibility. In particular, semantic HTML tags are used as well as hierarchical headings. Finally, articles are responsive (i.e., support different screen sizes) making them suitable for high browser zoom levels and large font settings.

Systematic Representation

Comparison tables form the main component to support systematic representations (Figure 7.4, node 4). The tables are created in a spreadsheet-like editor. The papers used within the comparison are represented as structured data in the graph, including the metadata such as title, authors, and publication date. Furthermore, the properties and their corresponding values are stored in the graph. When creating the comparison table, users are recommended to use existing properties and resources to further enhance interlinking.

7.3 Evaluation

The evaluation focuses on two distinct aspects of both the SmartReview approach and implementation. We present a use case with an original SmartReview article to demonstrate how SmartReviews look like and how they differ from regular articles. Afterwards, the created article is used to evaluate the three representational dimensions of the approach. Finally, we compare the SmartReview tool with existing authoring tools and methods.

7.3.1 SmartReview Use Case

As a use case and for evaluation purposes, an original SmartReview article is authored and published online.[4] The article presents a selective literature review, titled "Scholarly Knowledge Graphs". It consists of three comparisons and reviews in total 14 articles related to various types of scholarly knowledge graphs (i.e., identifier, bibliographic, domain-specific systems). This use case highlights the differences with regular static review articles. While regular review articles generally review the literature in comprehensive (and possibly lengthy) text sections, the SmartReview example shows how, instead, comparison tables are used to compare literature. Due to the interactive nature of the tables, they can contain

[4] `https://www.orkg.org/orkg/smart-review/R135360`

Table 7.1: Evaluation of SmartReviews against the FAIR data principles.

Principle	Description
Findable	
F1	A DOI can be assigned while publishing an article
F2*	Machine-readable metadata and provenance can be attached
F3	Metadata is stored separately from the published article
F4	Articles are searchable via the ORKG search interface
Accessible	
A1	Articles and their content can be accessed over HTTP via a REST API
A1.1	The protocol is open and free
A1.2	All articles are openly available and are not protected by authentication
A2	Articles are persistent even if the original graph data disappears
Interoperable	
I1*	Existing ontologies are used, user can use additional appropriate ontologies
I2	The used ontologies are globally unique and have persistent identifiers
I3	Links (e.g., "same as") are used to link to other ontologies
Reusable	
R1*	Metadata is published alongside an article
R1.1	Articles are published under a CC BY-SA license
R1.2	Each data entry has its own provenance data (e.g., creator, creation data)
R1.3*	Domain ontologies are used to publish articles, additional ontologies can be used

* Authors responsible for compliance

more information than tables presented in static PDF files. Another notable difference is the presence of ontology tables within the article. The benefit of such tables is twofold: They improve machine readability by linking the used properties to existing ontologies and improve human comprehension by textually describing the meaning of the property.

Machine-actionability and Systematic Representation

We now evaluate SmartReviews with the FAIR data principles. These principles provide guidance on how to achieve machine-actionable data. Per sub-principle, we outline how SmartReviews implement the guidelines and best practices. Although the FAIR data principles are not prescriptive [32], assessing each principle provides insights into the machine-actionability of the system. The evaluation results are presented in Table 7.1.

```
SELECT DISTINCT ?smartReview
WHERE {
  ?smartReview  a          orkgc:SmartReview;
                orkgp:P30  orkgr:R278.
}
```

Query 7.1: Return all SmartReviews with research field (P30) information science (R278).

```
SELECT DISTINCT ?paper
WHERE {
  ?contrib  a          orkgc:Contribution;
            orkgp:P32  orkgr:R49584.
  ?paper    orkgp:P31  ?contrib.
}
```

Query 7.2: Return paper contributions (P31) addressing Scholarly Communication (R49584) as research problem (P32).

```
SELECT DISTINCT ?section
WHERE {
  ?review   a                 orkgc:SmartReview;
            orkgp:P27         orkgr:R8193;
            orkgp:P31         ?contrib.
  ?contrib  orkgp:HasSection  ?section.
  ?section  a                 orkgc:Introduction.
}
```

Query 7.3: Return all introduction sections from SmartReviews related to information science (R278).

Additionally, to demonstrate the machine-actionability of SmartReviews, we now present four SPARQL queries that are used to query the underlying data (cf. Snippet 7.1, Snippet 7.2, Snippet 7.3, and Snippet 7.4). The first query is for metadata, whereas the other queries are for the actual knowledge presented in the respective articles. The presented queries can be run against the live ORKG SPARQL endpoint.[5] The prefixes *orkgc*, *orkgp* and *orkgr* represent the class, predicate and resource URIs respectively.

Accessibility

The accessibility of SmartReviews is evaluated using the Web Content Accessibility Guidelines (WCAG). For this purpose, we adopted the *Yale WCAG 2 A and AA Checklist*.[6] The guidelines specifically focus on content accessibility for users with disabilities. A summary of the

[5] https://www.orkg.org/orkg/sparql/

[6] https://usability.yale.edu/web-accessibility/articles/wcag2-checklist

```
SELECT DISTINCT ?paper
WHERE {
  ?contrib  a            orkgc:Contribution;
            orkgp:P32    orkgr:R49584;
            orkgp:P7009  "T"^^xsd:string.
  ?paper    orkgp:P31    ?contrib.
}
```

Query 7.4: Return all scholarly communication systems (R49584) with RDF support (P7009).

Table 7.2: Summary of accessibility evaluation of SmartReviews based on the Web Content Accessibility Guidelines (WCAG 2). Numbers represent the amount of conforming checklist items. Full evaluation available as auxiliary material online.[7]

	1. Perceivable	**2. Operable**	**3. Understandable**	**4. Robust**
Supports	18	14	9	4
Supports, author responsible	9	2	0	0
Supports with exceptions	3	3	2	0
Does not support	2	1	1	1
Not applicable	8	5	4	0
Total items	40	25	16	5

accessibility assessment is presented in Table 7.2. The full evaluation checklist is published online.[7] The evaluation includes both accessibility for readers and authors (i.e., content consumption and creation). As the results indicate, SmartReviews score well with respect to accessibility. However, it should be noted that although SmartReviews support accessible content authoring, the review authors are still responsible for several tasks. Among other things, this includes providing valid figures descriptions (i.e., alt text) and adding meaningful section titles. It is indicated in the evaluation tables when the system supports an accessibility feature, but the user remains responsible for complying with the respective guideline.

7.3.2 Authoring Tools Comparison

Table 7.3 compares multiple related tools and methods with the SmartReview approach. The first part of the table focuses on the evaluation of the SmartReviews dimensions. The second part compares the tools based on other relevant aspects. As the results indicate, SmartReviews score best, followed by Stencila. These two tools include the process of publishing an article online. The remaining tools, Dokie.li, Jupyter books, Word and Overleaf do not handle publishing. This means that the dimensions related to the review publishing

[7] https://doi.org/10.5281/zenodo.5102832

method are either limited or not supported at all. SmartReviews, Stencila, and Dokie.li all support machine-actionable content to a certain extent, either by providing (RDFa [24]) type annotations or by providing direct RDF access. SmartReviews additionally provide machine-actionable research data in the form of comparison tables.

7.4 Discussion

We acknowledge that our proposed approach is radical and will unlikely be immediately adopted in every aspect by the research community. While some of the weaknesses originate from technology limitations, the main challenge is not technological in nature. Rather it is rooted in researchers' habits and mindsets and being comfortable with familiar methods. This relates to the open access movement [166] which does not face a technical challenge but complex change that involves many aspects of traditional publishing.

Our proposed approach does not solely address review authoring but also impacts the publication and dissemination process. All articles can be published and accessed via the platform's user interface or directly via the graph. Therefore, the platform serves as a digital library for review articles. As discussed, any user can author new articles and contribute to existing articles. This means that articles are not peer-reviewed in the traditional sense, rather a community-based continuous review method is performed. However, traditional peer review is still possible. For example, as soon as an article is mature enough (which is decided by the authors), it can be published with traditional publishing means. However, we want to stress that a traditional publishing body is optional and is therefore not part of our approach.

In the evaluation, we specifically focused on the "Representational format" aspects as presented in Figure 7.2. The evaluation results showed that SmartReviews comply with the FAIR data principles. Also, we demonstrated how knowledge can be accessed in a programmable way using SPARQL queries. The accessibility evaluation showed that SmartReviews largely comply with the WCAG guidelines. Finally, we compared SmartReviews to other article authoring tools. The results show that SmartReviews do indeed score best on what we deem important for review authoring. It should be noted that these outcomes are to be expected since we specifically focused on these aspects during the development of SmartReviews. However, the comparison still provides insights into how other tools score on these dimensions. To ensure an objective assessment of the score, we published a textual justification online related to each score. The evaluation of the "Publishing method" dimensions is out of scope for this work. In order to evaluate these dimensions, a user evaluation is required, assessing the interactions and actual use of the system. Additionally, a user evaluation can focus on the usability aspects of the system.

As indicated in Table 7.3, the presented tool specifically focuses on review articles. However, our approach can be generalized to research articles. Concretely it means that the article writer can be used to author any type of scholarly article. We focused on

Table 7.3: Comparison of SmartReview with related tools and publishing methods. Full evaluation including a textual justification of each score is available as auxiliary material online.[7]

	SmartReviews	**Stencila**	**Dokie.li**	**Jupyter books**	**Word**	**Overleaf**
SmartReview dimensions[1]						
Article updates	5	5	3	1	1	1
Collaboration	5	3	3	2	2	3
Coverage	5	5	1	1	1	1
Machine-actionability	4	5	5	1	1	1
Accessibility	5	5	5	5	3	2
Systematic representation	5	4	4	1	1	2
Other aspects						
Scope	Scholarly Review Articles	Scholarly Articles (core focus)	Documents	Documents	Documents	Scholarly Articles (core focus)
Native output format	HTML, RDF(a), JSON, PDF	HTML, XML	HTML, RDFa	HTML, IPYNB, PDF	PDF, Non-semantic HTML	PDF
Input format	Markdown, Visual[2]	External files (multiple formats)	Visual[2]	Markdown, Python	Visual[2]	LaTeX
Maximum level Open Data stars	☆☆☆☆☆	☆☆☆☆☆	☆☆☆☆☆	☆☆☆	☆	☆
Open Source	✓	✓	✓	✓	✗	✓
Managed article publishing	✓	✓	✗	✗	✗	✗
Pricing model	Free	Freemium	Free	Free	Paid	Freemium

[1] Likert scale, from 1 (i.e., not supported) to 5 (i.e., fully supported)
[2] A visual What You See Is What You Get (WYSIWYG) editor

review articles because several of the weaknesses are most apparent for this type of article. Furthermore, we deem the limitation of static non-updated articles as a key limitation for reviews.

7.5 Summary

We presented the SmartReview tool, an application to author and publish scholarly review articles in a semantic and community-maintained manner. With the implementation, we address the current weaknesses of review article authoring and demonstrate a possible future of publishing review articles. A knowledge graph is used at the core of our approach, which increases the machine-actionability of the presented knowledge. In the evaluation, we specifically focused on the representational format of SmartReviews. The results indicate that SmartReviews comply with the FAIR data principles. Additionally, the accessibility evaluation showed SmartReviews are following the WCAG guidelines, which makes the articles accessible to users with disabilities.

In the chapter, we addressed **RQ2** by presenting an approach that incorporates literature overviews into machine-actionable review articles. **Challenge 2** is relevant to our approach as there is a trade-off between human- and machine-actionability. SmartReviews can contain natural text sections which are not machine-actionable. These sections specifically focus on human consumption. To address this issue and to increase machine readability, discourse classes are assigned to such text sections. These classes describe the function of the content presented within text sections. Further semanticication of the knowledge from natural text sections is possible but requires more user involvement and thus additional effort. **Challenge 5** relates to prospectively and retrospectively addressing scholarly communication. With SmartReviews, we presented a method to author scholarly articles and thus provide a method to prospectively improve scholarly communication. Furthermore, comparisons within SmartReviews provide structured data of already existing publications, thus retrospectively addressing scholarly communication.

CHAPTER 8

Crowdsourcing Scholarly Discourse Annotations

Populating knowledge graphs with scholarly metadata is a relatively straightforward task due to the low task complexity and high accuracy of automated parsing tools (such as GROBID [122]). Examples of knowledge graphs of scholarly metadata include Semantic Scholar [41] and Microsoft Academic Graph [42]. In contrast, generating graphs of the contents of research articles (i.e. research contributions) is a considerably more complex task which can currently hardly be performed by Natural Language Processing (NLP) tools alone. Crowdsourcing can be a solution: By including paper authors in the process of creating structured knowledge, it is possible to leverage human intelligence. However, crowdsourcing also comes with its challenges. Firstly, crowdworkers have to decide *what* to model, which requires a thorough understanding of the research topic. Secondly, crowdworkers have to decide *how* to model the knowledge, which is a cognitively demanding task that also relies on skill in conceptual modeling and possibly relevant technologies.

In this chapter, we present a methodology and web-based graphical user interface that serves as a first step towards intertwining human intelligence (via crowdsourcing) and machine intelligence (via machine learning) for the creation of a scholarly knowledge graph. The interface is designed to perform the task of annotating key sentences within scholarly PDF articles. This task focuses specifically on the aforementioned challenge of *what* to model. The user has to select a sentence and afterwards annotate this sentence with an appropriate class. The set of classes consists of a predefined set of 25 discourse elements (e.g., background, contribution and methods). During the annotation process, the user is supported by Artificial Intelligence (AI) tools. With this machine-in-the-loop approach [167] synergy is achieved between crowdsourcing and autonomous NLP extraction. The AI components are available for the tasks that require human judgement and provide support during the decision process. For example, selecting important sentences is supported by automated sentence highlighting. Additionally, selecting suitable classes for sentences is

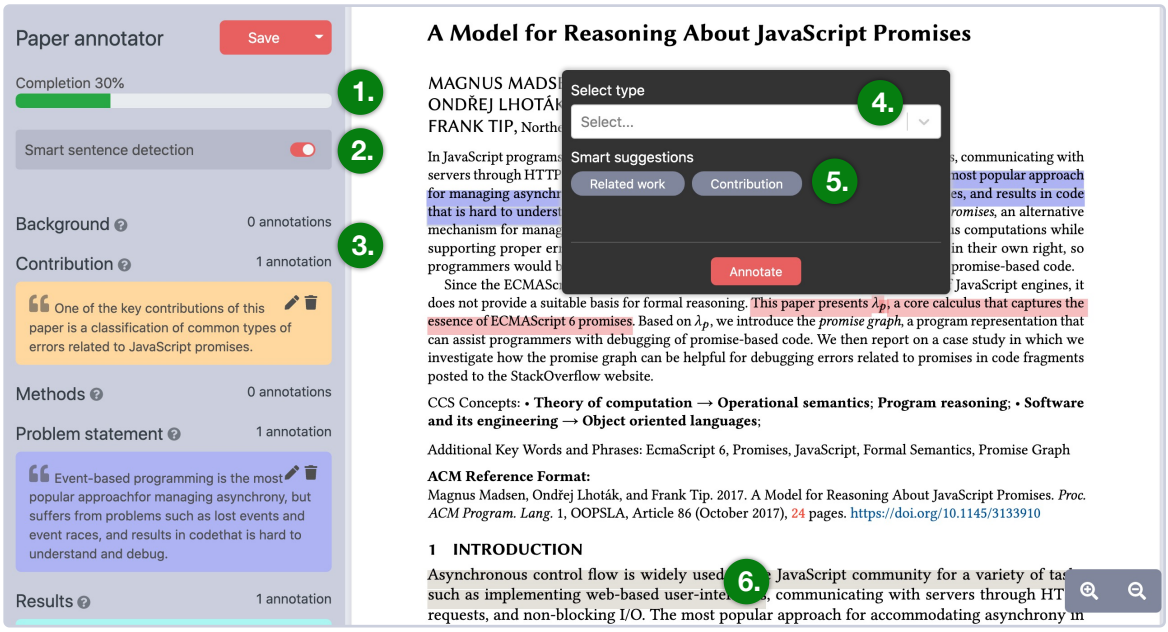

Figure 8.1: Screenshot of the annotation interface. Numbers indicate system components that are explained in detail in subsection 8.2.2. Legend: 1. Completion indicator 2. Automatic sentence highlighting 3. User annotations 4. Annotation class selector 5. Automatic class suggestions 6. Automatically highlighted sentence.

supported by a class recommendation tool. We envision that this interface is integrated in paper submission systems to produce a more structured description of the paper's content. Having annotated sentences is a crucial step towards generating truly structured semantic scholarly knowledge. Among other things, it is possible to further process the annotated sentence to create better structured and more semantic data.

In this chapter, we address the following research question:

> **RQ3:** How to intertwine machine and human intelligence for populating and curating a scholarly knowledge graph?

The research question is divided into two sub-questions:

(i) How to design an intelligent user interface to populate a scholarly knowledge graph using crowdsourcing?

(ii) How to employ a machine-in-the-loop approach to assist users in this process?

These questions are addressed by devising use cases which are used to determine the requirements. Based on the requirements, system components are designed that address those requirements. Finally, to evaluate whether the requirements are met, a user evaluation is conducted.

This chapter is based on the following publication: [168]. The rest of this chapter is structured as follows. Section 8.1 presents use cases for data entry and data consumption. Section 8.2 presents the system design and its requirements. Section 8.3 presents the evaluation. Section 8.4 discusses the approach and evaluation. Finally, Section 8.5 summarizes this chapter.

8.1 Use Cases

We now discuss multiple use cases supported by illustrative examples from the literature. We begin with two use cases in which the annotation interface is used to generate structured data (*data entry*). The four use cases that follow outline the usefulness of the generated annotations (*data consumption*).

8.1.1 Data Entry

Paper Submission

The annotation interface is mainly designed to be used as part of paper submission processes. More specifically, when the camera-ready article is uploaded. This prevents additional workload when uploading the paper for review. The interface can be integrated in open-access platforms such as arXiv [169] or CEUR Workshop Proceedings (CEUR-WS)[1]. A similar approach has been taken by arXiv, which integrated the ScienceWISE [170] platform where authors can add automatically generated entity annotations to their uploaded articles. Additionally, CEUR-WS has been frequently used as data source for semantic publishing approaches (e.g., [43, 171, 172]).

Literature Review

Sentence annotations can also be generated while reading articles. In this case, not only authors but also other researchers can create annotations. Compared to the *paper submission* approach, this will most likely produce less complete and possibly lower quality results. Less complete results are expected because readers will presumably only annotate what is of interest to them at the time of reading the article. Lower quality results are likely because readers are less familiar with the article's content than the authors. However, due to the scalability of this approach the generated annotations combined are still valuable. Although our interface is not designed to support this use case directly, it could be adopted easily.

[1] `http://ceur-ws.org`

8.1.2 Data Consumption

Further Semantification

The result of the annotation task is a set of sentences annotated with a relevant discourse class. These sentences must be transformed into more machine-readable descriptions. This can be done automatically using Named Entity Recognition and Classification (NERC) [173]. The resulting recognized entities can leverage the already determined discourse class. For example, if a method is recognized in a discourse element with the *Background* class, this means that the method is discussed within the paper. However, it does not necessarily mean that this method has been employed, since it is discussed as background information. Furthermore, the sentence can be modeled using existing ontologies. However, this task relies on domain experts with knowledge of data modeling.

Structured Abstract

Based on the annotated sentences a structured abstract can be generated automatically. Structured abstracts have a long history [174] and are commonly used in certain domains, most prominently in life sciences. Research shows that structured abstracts make it easier for researchers to select appropriate articles more quickly [175]. Within our user interface, the annotator is urged to only annotate the most important sentences. This results in an abstract that provides a relevant summary of the article.

Effective Search

With annotated sentences, search can be improved in two ways, by more effectively finding papers and by enhanced navigation within the paper. It is possible to more effectively look for concepts that are related to specific discourse elements. This can be further enhanced with additional semantification. Based on an experiment, Ribaupierre and Falquet [176] reported that the participants found more useful results using faceted search compared to keyword-based search. The facets were generated by extracting discourse elements and using annotations. Additionally, annotations can help in navigating the paper, displaying the highlighted sentences and their classes to readers. Highlighting sentences within a text has proven to increase information comprehension and retention [177, 178].

NLP Training Data

Finally, annotations can serve as gold standard for NLP-related tasks. A frequently recurring task in dataset generation is human annotation of the data. After the data is annotated (or labeled) it can be used to train and test machine learning algorithms. Labeling of datasets is oftentimes done manually by expert users (cf. [179]). This is an expensive and time-consuming task and therefore other methods have been proposed, for example, leveraging

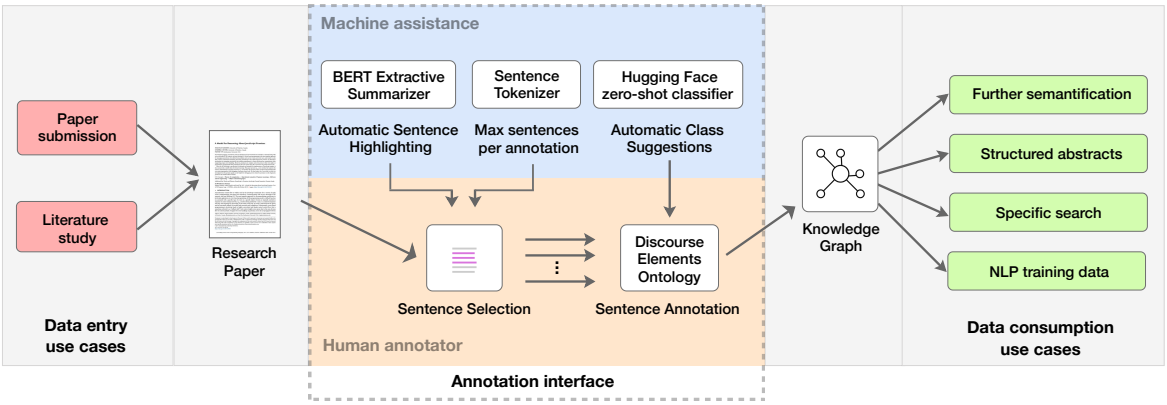

Figure 8.2: Overview on the system design illustrating the intertwining of human and machine intelligence for the scholarly article annotation.

crowdsourcing for dataset labeling [69]. The resulting data from our annotation system can be used to train NLP systems in multiple ways. Among other tasks, this includes the task of recognizing various discourse elements within a scholarly article.

8.2 System Design

In this section, we discuss requirements, present the system architecture and its components and outline the technical implementation. The annotation interface is integrated in the Open Research Knowledge Graph (ORKG) and is available online.[2]

8.2.1 System Requirements

Based on the use cases described in Section 8.1, we determined the system requirements from which the most essential ones are listed below. Additionally, we used the findings of a previously conducted user study where we asked seminar students (n = 14) to generate structured descriptions from papers they read. For this task, they had to use a tool that was designed to populate a scholarly knowledge graph by creating entities and the relations between them. In contrast to the annotation tool presented in this chapter, the tool did not rely on text annotation but on manual structured data creation. It was designed in such a way that it did not require any technical skills to perform the task.

The interface must adhere to the following functional requirements:

FR1 **Sentence annotation.** The interface should provide a method that enables users to select sentences within scholarly articles in PDF format. The selected sentences are annotated with an appropriate discourse class.

[2] `https://www.orkg.org/orkg/pdf-text-annotation`

FR2 **Task separation.** The task should focus on *what* to model and not *how* to model it. According to the seminar user study, 71% of the students indicated that the data modeling aspect is the most time-consuming aspect. By separating the task of data selection and data modeling, we provide a task that is more feasible for crowdsourcing during a paper submission process.

FR3 **Machine assistance.** Users should be supported by machine assistance during the annotation process. The machine assistance should provide guidance during the user's decision process. This includes guidance for which sentences to annotate and help to decide which class to annotate the sentence with.

Furthermore, we defined the following non-functional requirements:

NFR1 **Straightforward.** The task should be easy to perform. This means that the task is not cognitively demanding, has a low complexity, and takes little time to complete, which are typical characteristics used in crowdsourcing tasks [180]. The task easiness should not be confused with the usability of the system. In the seminar user study, the tool was evaluated with a System Usability Scale (SUS) [181] score of 67, which is average. Still, the task of modeling scholarly data in a structured way was a complicated endeavor.

NFR2 **Time efficient.** To convince paper authors to annotate their papers during the submission process, the task should not be time-consuming. We consider less than 10 minutes as time-efficient.

NFR3 **Well defined.** The task definition has to be unambiguous. This contributes positively to the quality and consistency of the generated data [182]. If the resulting annotations are according to the task description, it means the task is well understood and we consider the interface well defined.

8.2.2 Architecture and Components

The overall system architecture is shown in Figure 8.2. A key concept is to intertwine human and machine intelligence. A core component is the human user-driving sentence selection, which is facilitated by the two machine intelligence components, Extractive Summarizer and Sentence Tokenizer. Similarly, the second step Sentence Annotation using the Discourse Elements Ontology is facilitated by automatic class suggestions of the zero-shot classifier. We now discuss the individual system components. For each component, we explain how its design ensures that the system requirements are met.

Discourse Knowledge Representation

Users can choose between a predefined set of discourse classes to annotate a selected sentence (Figure 8.1, node 4). To support interoperability with other systems, we build on

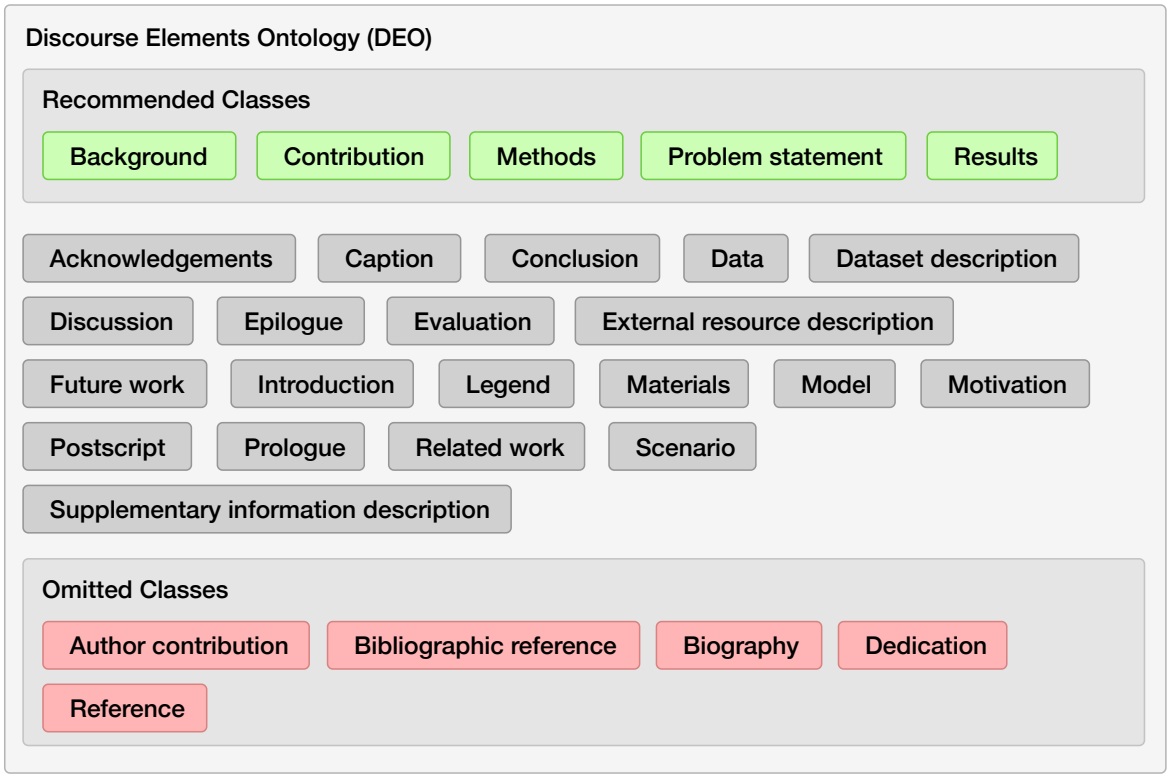

Figure 8.3: Discourse annotation classes. Green boxes indicate the recommended classes, red the omitted classes, and grey the remaining classes. In total, our model uses 25 classes.

the existing Discourse Elements Ontology (DEO) [49] to model the data. This ontology is part of the Semantic Publishing and Referencing (SPAR) ontologies which are designed to describe the scholarly publishing domain [48]. Our discourse knowledge representation model is illustrated in Figure 8.4. We omitted five classes as they are irrelevant for this annotation task (either because it is straightforward to extract this data automatically or because the data is not useful for the data consumption use cases). The omitted classes are: 1. Author contribution, 2. Bibliographic reference, 3. Biography, 4. Dedication and 5. Reference. The resulting set of discourse elements consists of 25 classes. These classes are listed in Figure 8.3. This component is part of FR1. Additionally, limiting the number of classes also contributes to NFR1 and NFR2.

Automatic Sentence Highlighting

To guide users during sentence selection, automatic sentence highlighting is applied. This is displayed in Figure 8.1, where node 2 and 6 respectively refer to activation and visualization of highlights. The highlights aim to ease the annotation task and are implemented for FR3 and NFR1. The highlights are generated by applying automatic sentence summarization

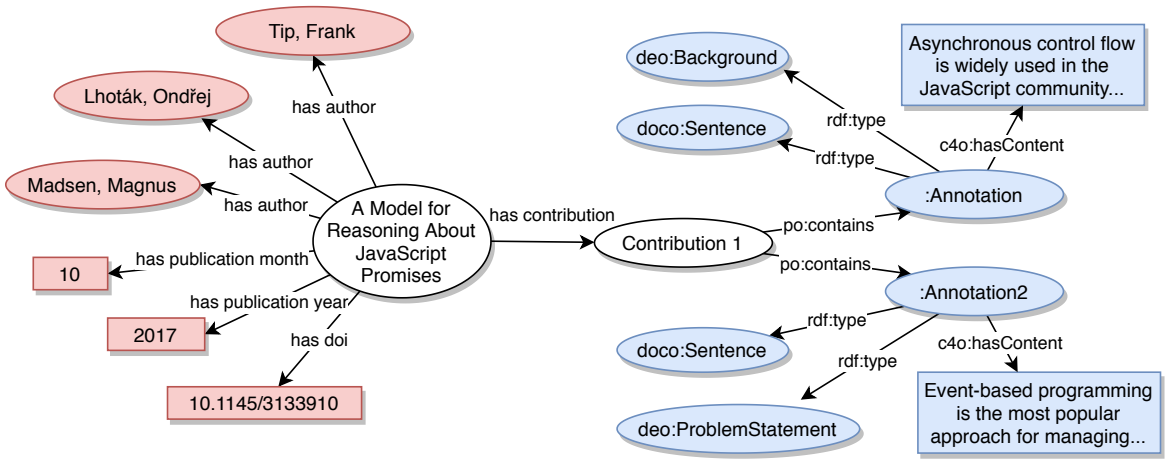

Figure 8.4: Example of the resulting knowledge graph obtained from the annotation task. The red nodes on the left depict the automatically fetched metadata (metadata types are omitted for simplicity). The white nodes are system concepts related to our internal data model. The blue nodes on the right depict two annotated sentences.[3]

to the full text of the article. The resulting summary sentences are highlighted within the text. For sentence summarization, we adopted BERT embeddings [183] for extractive text summarization inspired by the approach in [184]. Compared to abstractive summarization, where vocabulary is used beyond the specified text, extractive summarization uses the exact structures and sentences from the original text [185]. Extractive summarization is thus more suitable as it allows for tracing back and highlighting the original sentence.

Since summarization tools specifically focus on extracting key sentences from a text, we leverage this technique to highlight key sentences within the original text. Automatic text summarization techniques are not always accurate and therefore not commonly used. This is not an issue in our use case, since the highlights appear in context and can be ignored when not relevant, which contrasts to a self-contained summary where the quality of the summary plays a crucial role [186]. Furthermore, the user has the possibility to hide all automatically generated highlights (Figure 8.1, node 2).

Automatic Class Suggestions

The class suggestions help users to choose from the 25 discourse classes (Figure 8.1, node 5), thus addressing FR3 and NFR2. The class recommendations can save time during the annotation and are generated using a zero-shot classifier from Hugging Face [187]. A zero-shot text classifier is able to predict classes for text without requiring training data [188].

[3] Used ontologies: DEO (Discourse Elements Ontology) - `http://purl.org/spar/deo`, C4O (Citation Counting and Context Characterization Ontology) - `http://purl.org/spar/c4o`, PO (Pattern Ontology) - `http://purl.org/spar/po`, DoCO (Document Components Ontology) - `http://purl.org/spar/doco`

This makes such a classifier suitable for our task since the selected sentences can be classified according to the DEO ontology. The accuracy of the recommendations depends on the text structure. When certain key phrases are present in the text (e.g., "In the future..." for future work or "In the presented use case..." for a scenario) the classifier is able to make accurate suggestions. However, the accuracy drops when such key phrases are not present. A maximum of five suggestions ranked above an empirically determined threshold are displayed to the user.

Completion and Recommended Classes

The task completion bar indicates how complete the annotations are (Figure 8.1, node 1). It helps defining the task by providing guidance on the progress, which relates to NFR3. The completion rate only provides an indication, users do not have to reach 100% in order to finish the task. Completion is based on recommended classes, namely: 1. Background 2. Contribution 3. Methods 4. Problem statement 5. Results. The classes are selected based on the literature and the importance of these classes is argued as follows. Firstly, the classes are closely related to the elements from the IMRAD (Introduction, Methods, Results, Discussion and Conclusion) structured abstract style which are considered important features of articles [189]. Furthermore, findings from Ribaupierre and Falquet show that researchers are mainly looking for findings, hypotheses, methods, and definitions when reading scholarly literature [176]. These concepts are largely covered by the five recommended classes we selected. The completion rate indicator determines whether at least two annotations per recommended annotation class are created, which results in a completion rate of 100%.

Miscellaneous Guidance Functions

The following components further guide users during the annotation task.

1. **Annotation limit.** A maximum of three annotations per class can be created, thus maintaining the scope of the annotations (NFR3). It forces users to distribute the annotations across multiple classes which consequently contributes to higher data quality. The annotation limit (indicated by a warning) is not strictly enforced; hence, it is possible to deliberately cross the limit.

2. **Maximum sentences per annotation.** An annotation can only contain a maximum of two sentences. The selected text is tokenized by sentences. This also contributes to NFR3. It prevents users from annotating full paragraphs and forces them to select only key sentences within the article. As with the annotation limit, a warning is displayed as the limit is a suggestion and not enforced.

3. **Tooltips and guidance.** Tooltips are displayed throughout the interface. This contributes to NFR1. The tooltips explain system functionalities and the DEO classes. For

each class, a description explains the purpose of the class. Furthermore, a guided help tour automatically appears when using the interface. The tour explains the goal of the annotation task and provides an overview of the main functionalities.

8.2.3 Technical Implementation

The interface has been implemented in JavaScript using React, the source code and its documentation are available online[4]. For displaying PDF files, we used an extended version of PDF.js[5], which is a JavaScript library for parsing and rendering PDF files developed by Mozilla. Since PDF.js is used as default PDF viewer within the Firefox web browser it is able to correctly display PDF files within a browser environment. Additionally, PDF.js has been used successfully in other PDF annotation tools (e.g., [113, 114]). The default PDF search functionality is leveraged and extended to support multiple search queries at once. The endpoints for the machine learning components are implemented in Python.

For saving the annotations, users are requested to provide a paper title or a Digital Object Identifier (DOI) to save the data. In case a DOI is provided, additional metadata related to the article is automatically fetched via Crossref [190]. Among others, this includes the article's title, authors, and publication date. Users do not have to provide this data manually, which makes the annotation task more time-efficient (NFR2). Figure 8.4 visualizes an example of the data structure for a saved paper. Various external ontologies are used to improve data interoperability.

8.3 Evaluation

The interface is evaluated to determine whether the paper annotation task is indeed a feasible task to be performed by academics. Additionally, we want to obtain insights in the attitudes towards machine-assisted paper annotation in general. We focus specifically on evaluating the individual components discussed in subsection 8.2.2. The evaluation also provides insights into whether or not the functional requirements are met and thus if the functionalities were indeed designed as envisioned, as well as non-functional requirements and thus if the quality aspects are met. We evaluated the interface by means of a user study. Firstly, we evaluated the participants' opinions about the usability and their attitudes towards our approach in general. Secondly, we analyzed the data produced by the participants during the annotation task.

[4] https://gitlab.com/TIBHannover/orkg/orkg-frontend/-/tree/e0a6a7a8d022119d9fb5cc7b749052f0f1c194d0/src/components/PdfTextAnnotation

[5] https://mozilla.github.io/pdf.js

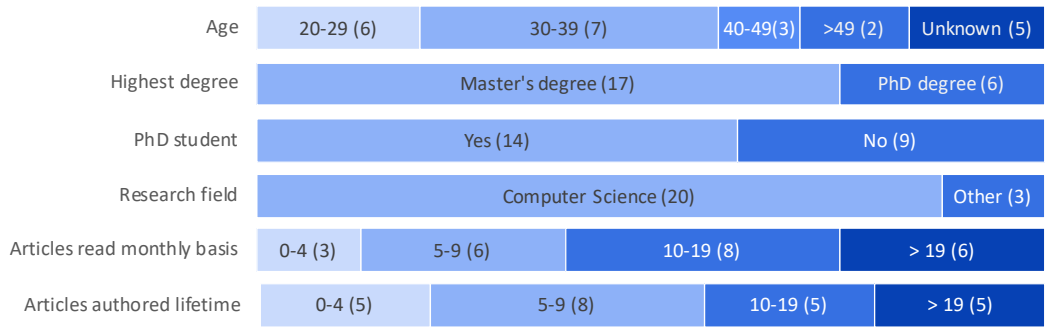

Figure 8.5: Participants' demographics (n = 23).

8.3.1 Evaluation Setup and Data Collection

An online task description was circulated among academic communities. This task description provided a brief explanation of how to participate in the evaluation. Participants were asked to annotate a paper with the paper annotation interface described here. This could either be an article they authored themselves or an article they (recently) read. Afterwards, participants were asked to complete an online questionnaire. The task description did not provide any instructions regarding the functionalities of the interface nor did we instruct the participants regarding the annotation task. This ensures that the interface can be used without external assistance and matches the real-world setting in which authors are asked to annotate their articles during submission without further help. We communicated that the evaluation takes approximately 20 to 30 minutes in total. A total of 23 researchers participated in the user study. Figure 8.5 displays the demographics of the participants, including data for the number of articles each participant reads and publishes, as a proxy for the level of expertise. Participants with more experience on reading and writing articles are presumably able to annotate more quickly and with a higher quality. As the demographics data shows, participants with varying levels of expertise participated in the study.

To determine the usability of the interface, we incorporated the System Usability Scale (SUS) [181] in the questionnaire. Furthermore, to determine the workload of the task we included questions from the NASA Task Load Index (TLX) [191]. This provides insights into the perceived workload by participants for the annotation task. To reduce the length of the questionnaire, we conducted the Raw TLX, which eliminates weighting the questions. Finally, we included additional questions to determine the participants' attitude towards the interface and the overall task. This included a question asking for general feedback about the interface.

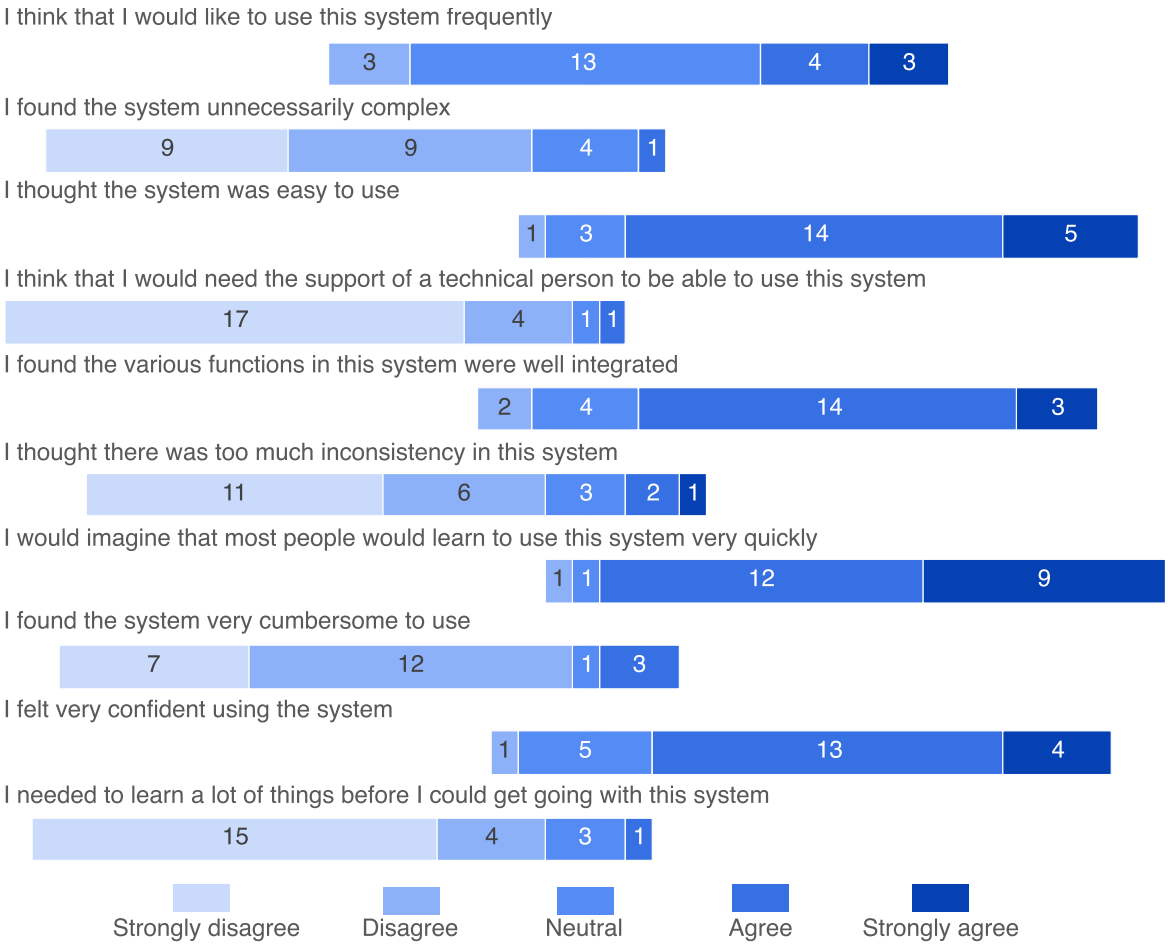

Figure 8.6: Individual System Usability Scale questions and answers, resulting in a mean score of 76.09 (SD = 14.38).

8.3.2 Evaluation Results

The System Usability Scale evaluation resulted in a score of 76.09 (out of 100) which is considered "good". The individual questions and answers are displayed in Figure 8.6. Because of the format, text selection and extraction in PDF files remains a challenging task (see also [114]). Various participants explained that the text selection should be improved. The question "I think that I would like to use this system frequently" is rated lowest. An explanation could be that the participants are not (yet) performing article annotation on a regular basis in daily work. Therefore, the question is answered based on their own situation rather than on the general usability of the system. A similar conclusion was suggested by Weber et al. [192].

The results of the TLX evaluation are shown in Figure 8.7. The mean TLX score of 35.87 is considered low compared to the mean of 45.29 found in the meta-analysis from

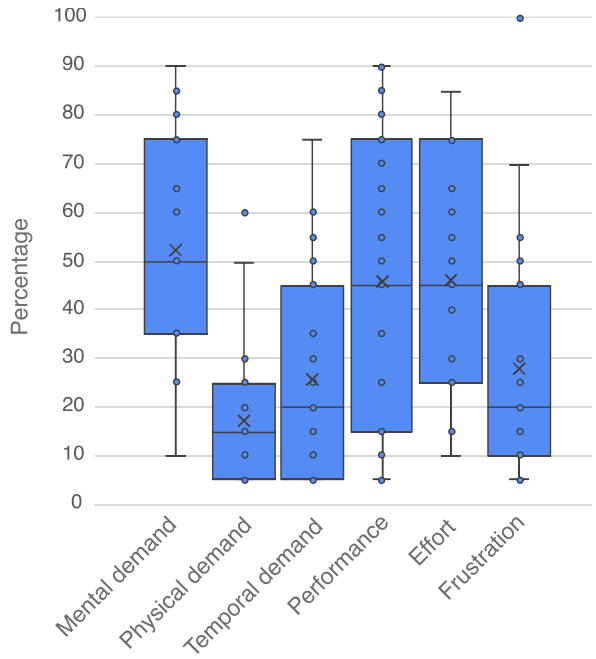

Figure 8.7: Raw NASA Task Load Index results (lower is better). The mean TLX score is 35.87 (SD = 26.17). The middle line represents the median and a cross the mean. Vertical lines represent the minimum and maximum values and circles the individual points and outliers.

Grier [193]. This indicates a low perceived workload by the participants. On average, the three highest scored questions are related to mental demand (52%), performance (45%), and effort required (46%). This means that the annotation task in general does require some mental effort. However, this is expected due to the various task constraints (e.g., annotate only the most important sentences or a maximum of three annotations per class) and will possibly be partially mitigated by increasing familiarization of users during regular use of the system. The frustration level was relatively low (28%), this is in line with the positive SUS score.

The participants' attitudes towards the interface are visualized in Figure 8.8. Participants are split on the question whether the task is time-consuming, most participants rate this as neutral. Most participants spent between five and 10 minutes to annotate their paper (52%). Of the remaining participants, 18% spent less than five minutes and 30% more than 10 minutes. No clear time difference could be observed between more experienced participants (i.e., participants with a PhD degree) and other participants. The majority of participants would be willing to annotate their paper in the submission process, given that the paper has been accepted already. The remaining questions in Figure 8.8 relate to the machine-assisted aspects of the system. The vast majority of the participants has a positive attitude towards leveraging machine-assisted technologies during the annotation task as they would like to see more artificial intelligence technologies being integrated. Participants are also split

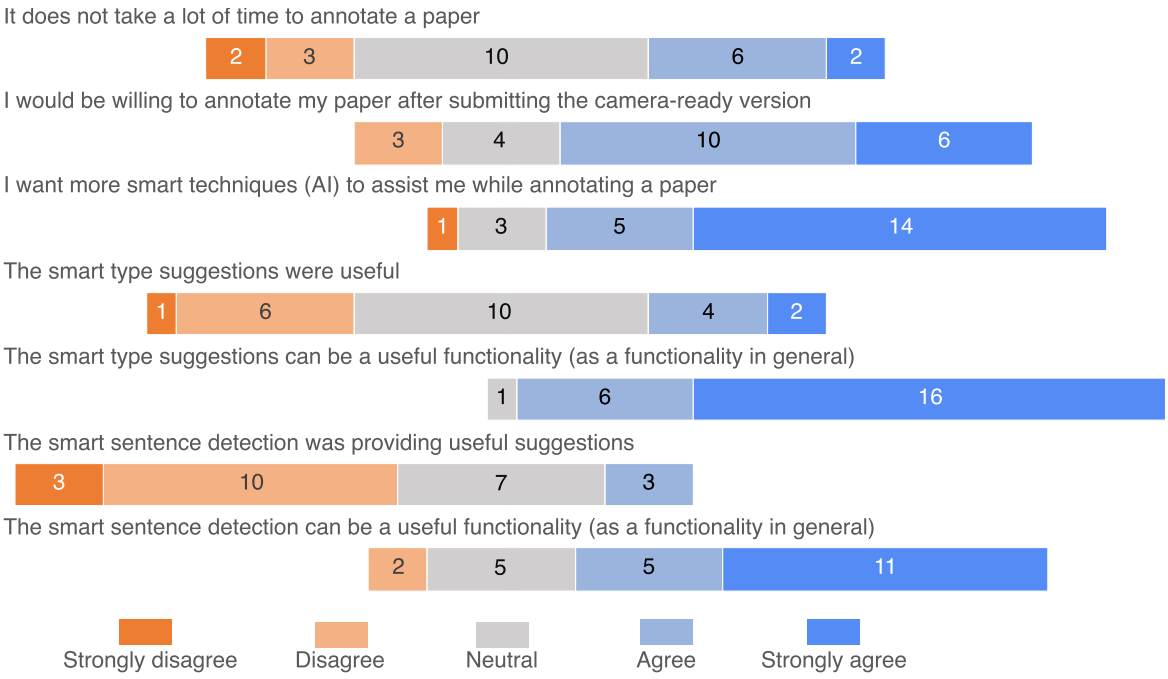

Figure 8.8: Participants' attitudes towards the annotation task, specifically focused on the machine assistance perspective. Higher values in green represent more positive attitudes.

on the quality of the automatic class suggestions (called smart type suggestions in the interface). However, the majority agrees that the functionality is useful, given that the suggested classes are more relevant. The results of the automatic sentence highlighting (called smart sentence detection in the interface) are not always helpful according to the participants. But also here, most of the participants agree that the functionality is useful in general. These relatively positive results indicate that the participants appreciate the integration of AI in a user interface, even though the individual performance of the machine learning components leaves room for improvement according to some participants. This is expected given that we did not focus on a particular scholarly domain. Overall, this confirms that our approach for sentence annotation interface is a promising direction.

Furthermore, we determined whether the preselected recommended classes are indeed of interest to researchers. In the questionnaire, participants were asked to select five discourse classes they deem most important when reading scholarly literature. The 16 most selected classes are listed in Figure 8.9. As this figure indicates, four of the recommended classes are indeed considered most important. Ranked 10th, the background class is the only exception. Since the background class was included in the recommended classes, it was more prominently positioned in the interface. Therefore, it has a relatively high annotation frequency compared to the perceived class importance. Although not considered important by the participants, background information is valuable especially when creating structured abstracts. Therefore, we suggest to keep the background class in set of recommended classes.

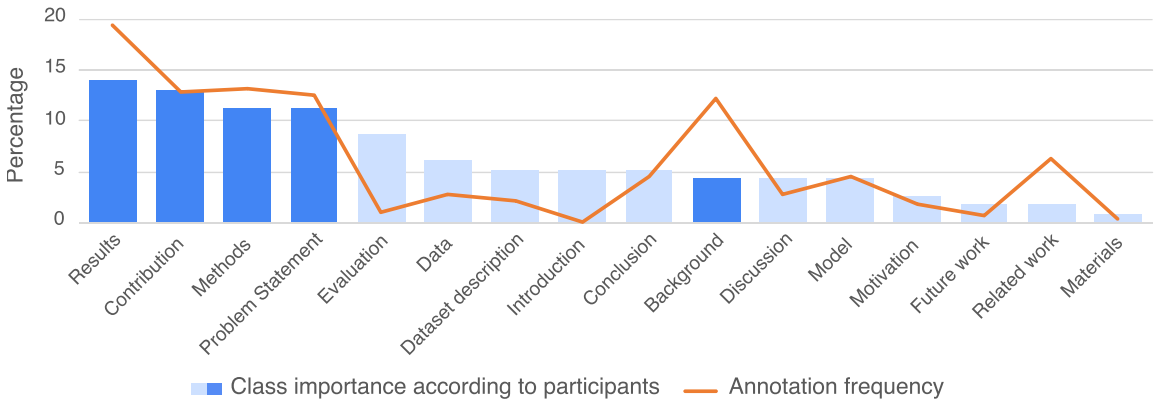

Figure 8.9: Top 16 discourse classes ranked by importance according to the participants (in dark green the recommended classes). The orange line shows the annotation frequency per class.

Table 8.1: Statistics of the generated annotations per article from the user evaluation.

	Mean	SD	Max	Min
General				
Annotations per article	13.18	6.52	24	3
Completion ratio	72.72	24.72	100	10
Extra recommended classes	1.90	1.94	7	0
Non-recommended classes	3.91	4.42	14	0
Machine-assisted components				
Selected class in suggestions ratio	56.55	29.40	94.44	33.33
Selected class as first suggestion ratio	17.24	14.65	50	0
Annotations over two sentence limit	0.95	1.56	4	0
Annotations over three class limit	0.82	1.97	9	0

Furthermore, this figure displays the number of annotations per the listed discourse classes. As expected, the recommended classes are used most frequently, as they are prominently present in the interface. Interestingly, the related work class is used relatively frequently as well. This could be explained by the assumption that is it straightforward to recognize related work within an article.

Table 8.1 reports statistics for the generated annotations during the evaluation. An average completion rate of 73% has been reached. The completion rate only provided guidance and it was not mandatory for the participants to reach 100%. The relatively high completion rate indicates that participants were indeed guided by the completion bar, but did not feel obligated to reach the full completion. Per article, on average 3.9 non-recommended annotation classes were used. Indicating that the interface was successful

in guiding users towards the recommended classes but also allowing other classes. With respect to the machine-assisted components, 57% of the suggested classes were indeed selected by the user. In 17% of the cases, this was the first suggestion in the list (i.e., the class with the highest certainty, as determined by the classifier). This leaves room for improvement which is in line with the results from the questionnaire (Figure 8.8). Finally, on average one annotation per article contained more than two sentences. In this case, a warning was shown to the user, which did however not hinder saving the annotation. The same applies to the maximum number of annotations, which was set to three. On average, an annotated article had one annotation class with more than three annotations. These relatively low numbers of crossing the limits indicate that warning participants about violations, but not enforcing them, is indeed effective.

8.4 Discussion

In order to answer our first sub-question, "How to design an intelligent user interface to populate a scholarly knowledge graph using crowdsourcing", we determined the system requirements based on several use cases. Our user evaluation focused on determining whether the requirements are met. Based on the functional requirements, we implemented a PDF sentence annotation component. The annotation task was focused towards what to model and not on how to model data. For example, users do not have to decide what ontologies to use or how to structure the data. Related to the second research question, "How to employ a machine-in-the-loop approach to assist users in this process", we integrated multiple machine-assisted technologies. This includes machine learning-based components, such as the automatic sentence highlighting and the automatic class suggestions. With respect to the non-functional requirements, we conclude that the task was indeed straightforward as suggested by the NASA Task Load Index (TLX) results. Furthermore, the evaluation shows that most participants spent less than 10 minutes for the task. We consider that as time-efficient, although the results were divided for the question "It takes a lot of time to annotate a paper". Despite that, most authors are willing to annotate their paper during the camera-ready submission. This suggests that a crowdsourcing approach, in which authors are included to generate structured paper data, is viable in practice. Finally, participants were able to perform the task without requiring additional help. They reported high levels of confidence and low frustration levels while using the system. This indicates that the task was well defined.

Once the scholarly knowledge graph contains a sizeable number of articles it can potentially revolutionize scholarly communication. For example, by providing more effective search or as a tool to analyze scholarly knowledge more efficiently. Our annotation interface serves as a step towards more structured scholarly communication. Generally, the more structured the data in the graph is, the better machines can read and process this data. Specifically, the annotated sentences can be complemented with structured data to

further improve the data's machine readability. This can be done in an automated fashion by leveraging techniques such a Named Entity Recognition (NER) [173] to automatically detect concepts in a sentence. This results in additional structured data which in turn can be further enhanced by linking these concepts to other knowledge graphs, by means of Entity Linking [133]. These technologies can be effectively employed by leveraging the annotated sentences, thus applying them targeted on a specific sentence rather than on the full text of an article. Future work will focus on applying these technologies on the annotated sentences.

The evaluation results indicate that the usability of the system is "good" and that the workload is acceptable. With respect to the machine assistance, specifically the automatic sentence highlighting and automatic class suggestions, participants suggested that the quality of the recommendation could be improved. Improving these specific machine learning algorithms is out of scope here. More interestingly, participants indicated that they appreciate the overall integration of Artificially Intelligence (AI) within the user interface. Despite the quality of recommendations not being optimal, they would prefer more AI-powered support during the annotation. We conclude that the quality of the assistance does not have considerable negative impact on the user experience nor does it significantly influence the participants' attitude towards such technologies. This contributes to the concept of machine assistance, whereby a machine could help a user but is not critical to complete the task. Participants were able to ignore class suggestions and to disable sentence highlights if they considered them to be irrelevant. Therefore, we argue that the possibility to dismiss machine assistance is crucial for a system's usability.

The presented interface and the findings from this work are not exclusively applicable to the scholarly domain but can be transferred to other domains as well. The PDF format is widely used in various fields and applications (legal documents, patents, etc.) where they dominate as a digital means to share knowledge. With minor adjustments, the presented interface can be adopted to annotate such documents and ultimately generate structured data from them. In principle, merely the ontology for annotation classes has to be changed to support other use cases. Furthermore, our findings related to users' attitudes towards machine-assisted user interfaces are relevant to interface design in general.

Limitations

Arguably, our evaluation could be larger and include more participants, which would improve the validity of the results. We target participants with an academic research background, which are notoriously hard to recruit. Moreover, the task is not suitable for online crowdsourcing platforms such as Amazon Mechanical Turk. Additionally, a more thorough evaluation of the effectiveness of the intelligent system components is required. We acknowledge that the evaluation is limited in scope and are considering to conduct a broader evaluation with a more diverse audience for future research. Despite these limitations, the evaluation still provides helpful insights and clear indications on

the participants' attitudes towards the overall approach. Moreover, Tullis and Stetson [194] have shown that the System Usability Scale (SUS) provides reliable results even with relatively small sample sizes (e.g., n = 12).

Most of the participants have a Computer Science related background (Figure 8.5). This could introduce a bias affecting the usability score and overall attitude towards intelligent technologies. Indeed, computer scientists are generally more experienced in adopting novel computer user interfaces. However, the interface was designed to also allow non-technical users to annotate papers. For example, technical jargon is avoided to make the interface understandable for users with different backgrounds. Furthermore, text annotation has been successfully employed in other domains (e.g., [195, 196]), indicating that the task itself is generalizable across domains.

8.5 Summary

We presented a web-based user interface to crowdsource scholarly discourse annotations. The interface integrates several machine-assisted components to guide users during the annotation process. This work is part of a larger research agenda and a corresponding open science infrastructure development. We deem that the integration of human and machine intelligence for creating a comprehensive knowledge graph representing research findings is a key prerequisite for solving scholarly communication deficiencies such as the proliferation of publications, the reproducibility crisis, or the deterioration of peer-review. In particular, we envision that the interface is integrated in paper submission processes where paper authors are requested to annotate their own papers. A scholarly knowledge graph is created using the annotated sentences combined with the paper's metadata. Our user study results indicate that the annotation interface has a good usability and that the annotation task does not require significant cognitive workload. This suggests that sentence annotation is a feasible task to be performed by researchers.

This chapter addressed **RQ3** by intertwining human- and machine-intelligence to populate a scholarly knowledge graph. With our machine-in-the-loop approach, we demonstrated how users can benefit from machine assistance to accomplish the task of generating structured scholarly knowledge. **Challenge 1** relates to this chapter since the key point of the annotation interface is to transform unstructured knowledge into a more structured form. Different from semi-structured tabular extraction presented in Chapter 6, in this chapter users annotate unstructured sentences. Therefore, also **Challenge 2** is relevant to this chapter. The output of the annotation task should be usable (i.e., actionable) for both humans and machines. Compared to the original form of the presented knowledge from the narrative articles, annotated sentences are significantly more machine-readable. But, as we discussed previously, annotating sentences is just the first step to more machine-actionable scholarly knowledge.

CHAPTER 9

Intertwining Natural Language Processing with Microtask Crowdsourcing

For centuries, scholarly knowledge has been communicated in a narrative document-based, and largely unstructured form [146]. In order to create a scholarly knowledge graph, structured knowledge has to be either extracted from the unstructured documents or produced directly upfront in the research workflow [197]. There are different strategies to support the extraction process. It is possible to manually extract structured knowledge with human labor. Although this will most likely result in high-quality data, this approach does not scale well. Another approach is to automatically extract structured knowledge using machine learning techniques. Specifically, Natural Language Processing (NLP) is able to interpret natural language and transform unstructured content into a structured, machine-readable representation. However, NLP tools are not sufficiently accurate to generate a high-quality knowledge graph, in particular, due to the complexity of the conveyed information, the required context-awareness, or the varying levels of semantic granularity. Naturally, quality is a crucial aspect for a scholarly knowledge graph to become a valuable resource for researchers. Thus, in this chapter, we propose a hybrid method where we combine human and machine intelligence via microtasks to create a structured scholarly knowledge graph. This results in a synergy and combines the advantages of each approach, i.e. the quality aspect from *human* intelligence and the scalability aspect from *machine* intelligence.

We present *TinyGenius*, a methodology to create a scholarly knowledge graph leveraging intertwined human *and* machine intelligence. Firstly, NLP tools are used to autonomously process scholarly articles. Secondly, the NLP results are transformed into a paper-centric scholarly knowledge graph. Finally, the statements are presented to humans in the form of microtasks. Humans can vote to determine the correctness of the statements. Votes are

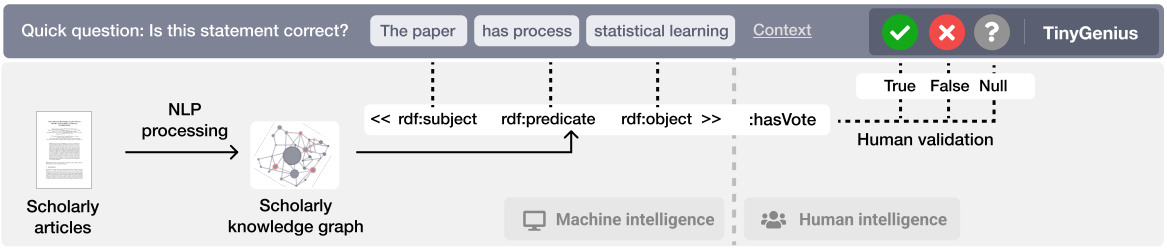

Figure 9.1: Graphical abstract. Workflow of the TinyGenius methodology. Scholarly articles are processed by NLP tools to form a scholarly knowledge graph (*machine intelligence* part). Afterwards, the extracted statements are validated by humans by means of microtasks (*human intelligence* part). User votes are stored as provenance data as part of the original statements.

stored as provenance data on statement level. Based on the votes, an aggregated score is computed to indicate the correctness of a statement. TinyGenius is designed to be integrated into an existing scholarly knowledge graph infrastructure, specifically in the ORKG. The task of transforming unstructured into structured knowledge, even with NLP assistance, is a complex and time-consuming endeavor. We, therefore, propose a method that decomposes this large task into a set of smaller microtasks. Once integrated into the platform, microtasks are displayed throughout the web interface. This enables regular visitors to be *content producers* not just *content consumers*. For each NLP tool, a specific microtask is designed. This is to ensure task simplicity and to provide a task that can be answered without contextual knowledge (i.e., without reading the article). A screenshot of the interface showing an example of a microtask is depicted in Figure 9.1.

In this chapter, we address the following research question:

> **RQ3:** How to intertwine machine and human intelligence for populating and curating a scholarly knowledge graph?

More specifically, we make the following contributions:

(i) The TinyGenius methodology to validate scholarly NLP results using crowdsourced microtasks.

(ii) A modular architecture to create a scholarly knowledge graph at scale with NLP tools.

(iii) An evaluation of the methodology and architecture by means of a user study and case study, respectively.

The chapter is based on the following publication: [198]. In addition to this publication, an evaluation is added. The remainder of this chapter is structured as follows. Section 9.1

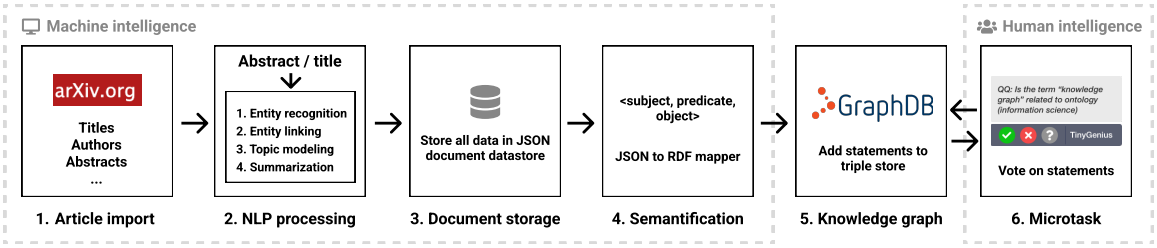

Figure 9.2: TinyGenius methodology intertwining human and machine intelligence to create a scholarly knowledge graph. ArXiv articles are imported, processed by a set of NLP tools, and the results are stored. From the results, a knowledge graph is generated. Afterwards, humans validate the knowledge graph by means of microtasks.

presents the system architecture and employed NLP tools. Section 9.2 discusses the user interface. Section 9.3 evaluates the approach and previously presented interface. Section 9.4 discusses the results. Finally, Section 9.5 summarizes the work.

9.1 Architecture and NLP

We now discuss the TinyGenius methodology. First, we focus on the technical infrastructure that is responsible for data storage and processing. Afterwards, we explain the user interface in more detail.

9.1.1 Data Model

By using a standardized data representation model, the data interchange between machines is facilitated. This increases the machine-actionability of the data, which is defined as the ability for machines to interpret the data without the need for human intervention [26]. RDF data can be queried using the SPARQL language [104]. A key aspect of our approach is storing the user votes as provenance data to statements in the knowledge graph. This means not only a final correctness score of a statement is available, but also the underlying information used to determine the score. Among others, the provenance data includes the votes, contextual information, and confidence score of the NLP tool. There are different approaches to store provenance data in RDF [199], for example, standard reification, singleton properties [200], named graphs [201], and via RDF* [202]. We adopted the RDF* representation as this provides a method that scales well and, compared to the other approaches, provides improved comprehensibility for SPARQL queries.

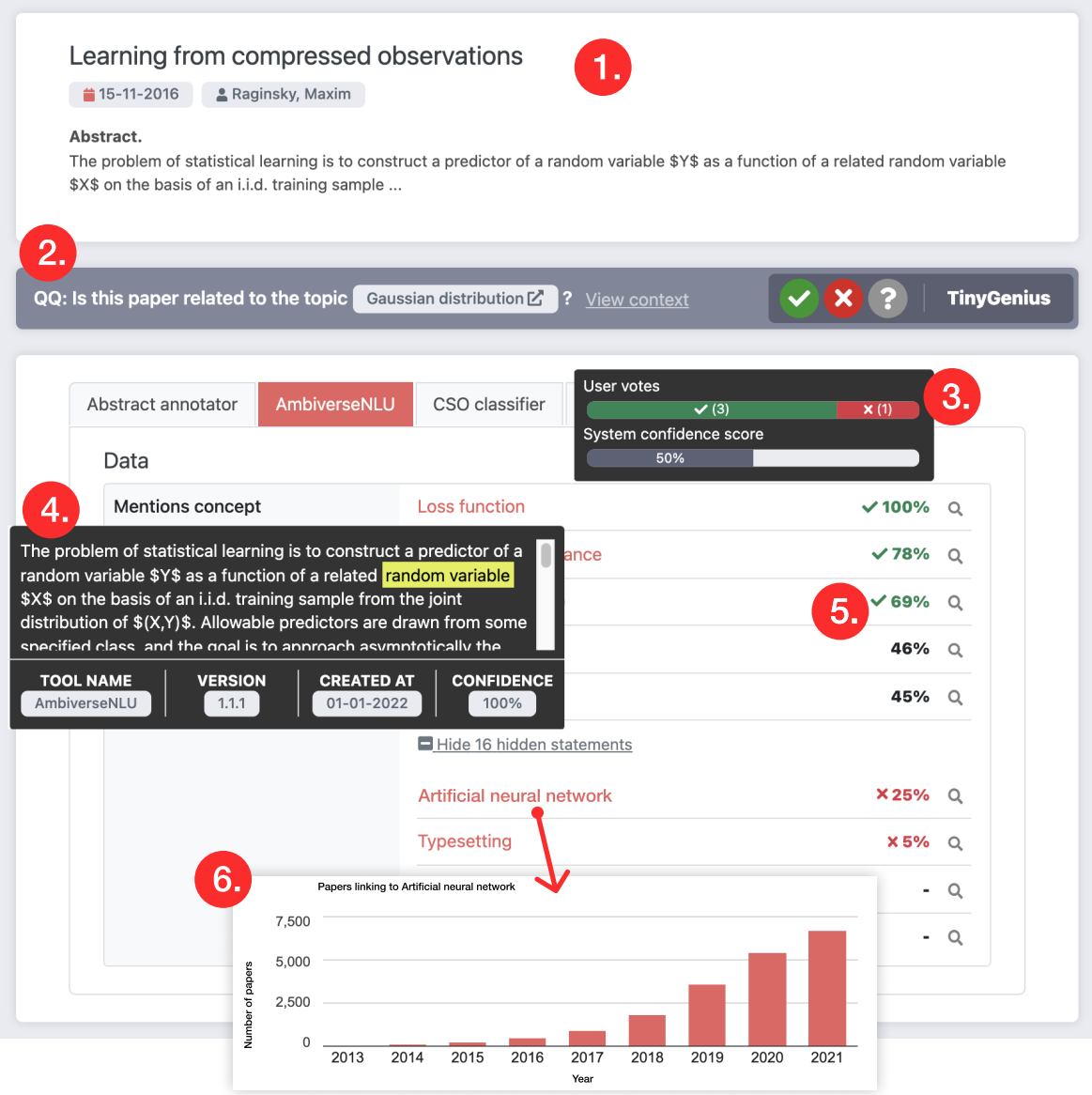

Figure 9.3: View paper page, showing the integrated voting widget and NLP statements. Node 1 displays the metadata related to the selected paper. Node 2 shows the voting widget. Node 3 is the score tooltip. Node 4 shows a tooltip that displays the context and provenance data related to a single statement. Node 5 lists the NLP-generated statements grouped by the tool. Finally, node 6 shows the use of a resource grouped by year, which is displayed when clicking on a resource.

9.1.2 Technical Infrastructure

One of the key benefits of using NLP tools to process data is the ability to perform this analysis at scale. Therefore, the infrastructure is designed to handle large quantities of data while still performing well. We outline the methodology depicted in Figure 9.2:

1. In the first step, the complete metadata corpus from the open-access repository service arXiv[1] is imported. This includes article titles and abstracts. To reduce the required computational resources and ensure a consistent level of semantic granularity, only paper titles and abstracts are processed by NLP tools (i.e., the full text is excluded).

2. Afterwards, the papers are processed by different NLP tools, which we discuss in subsection 9.1.3.

3. In the third step, the output of the paper import process and the resulting data from the NLP tools are stored in a document-based JSON data store. Notably, the NLP results are stored in their native data model and are not transformed to make them suitable for knowledge graph ingestion.

4. The semantic transformation process takes place in the fourth step, i.e. semantification. This step converts the native NLP data models to a triple format, as required by the RDF data model. The original data from step three remains available in the original JSON data store. This allows to create a different mapping from the NLP models to RDF at any time in the future and it separates the concerns between data processing and data modeling.

5. In the fifth step, the data is ingested in a triple store. As discussed previously, we adopted an RDF* provenance data model. Therefore, a GraphDB[2] triple store is used, which supports RDF* natively. The data model, including an example of data provenance statements, is depicted in Figure 9.4. To increase machine-actionability, existing ontologies concepts are used when possible.

9.1.3 NLP tools

We employed a set of five different NLP tools to process the articles. The TinyGenius methodology itself is not limited to this set of tools and can be easily extended with other NLP tools. The tools are listed in Table 9.1. The selected tools provide a representative sample of different NLP tasks. The CSO classifier takes an article abstract as input and outputs extracted topics. This classifier is a domain-specific model designed to classify Computer Science abstracts [203]. The related microtask asks users whether the extracted

[1] `https://arxiv.org/`
[2] `https://graphdb.ontotext.com/`

Table 9.1: List of employed NLP tools and their corresponding task and scope. The question template shows how the microtask is presented to the user.

Tool name	NLP task	Scope	Question template
CSO classifier	Topic Modeling	Domain-specific	Is this paper related to the topic {topic}?
Ambiverse NLU	Entity Linking	Generic	Is the term {entity} related to {wikidata concept}?
Abstract annotator	Named Entity Recognition	Domain-specific	Is this statement correct? This paper {type} {entity}
Title parser	Named Entity Recognition	Domain-specific	Is {entity} a {type} presented in this paper?
Summarizer	Text Summarization	Generic	Does this summarize the paper correctly?

topic is indeed relevant for the paper. The Ambiverse Natural Language Understanding (NLU)[3] [204] tool links entities found in a text to a corresponding entry in Wikidata [40]. The microtask is related to determining whether the concept is correctly linked. Users can visit the corresponding Wikidata page to determine the correctness. The Abstract Annotator extracts four classes from paper abstracts: data, material, method, and process [143]. The related microtask lets users validate whether the entity indeed belongs to the selected class. The Title parser is similar to the Abstract annotator, but focuses specifically on titles, which typically follow certain conventions exploited by this tool. Finally, the Summarizer takes an abstract as input and summarizes that into a text piece of maximum 120 characters. The microtask asks users to indicate whether the generated abstract is indeed a reasonable summary.

9.2 Microtask Crowdsourcing User Interface

The user interface consists of two main components: the voting widget and the view paper page. The voting widget is self-contained, meaning that it contains all the required context to perform the microtask. The view paper page integrates the voting widget for the displayed paper.

9.2.1 Voting Widget

The voting widget is the key interface component and integrates the microtasks to perform the NLP validation. It is displayed in Figure 9.3. The widget is self-contained, modular,

[3] `https://github.com/ambiverse-nlu/ambiverse-nlu`

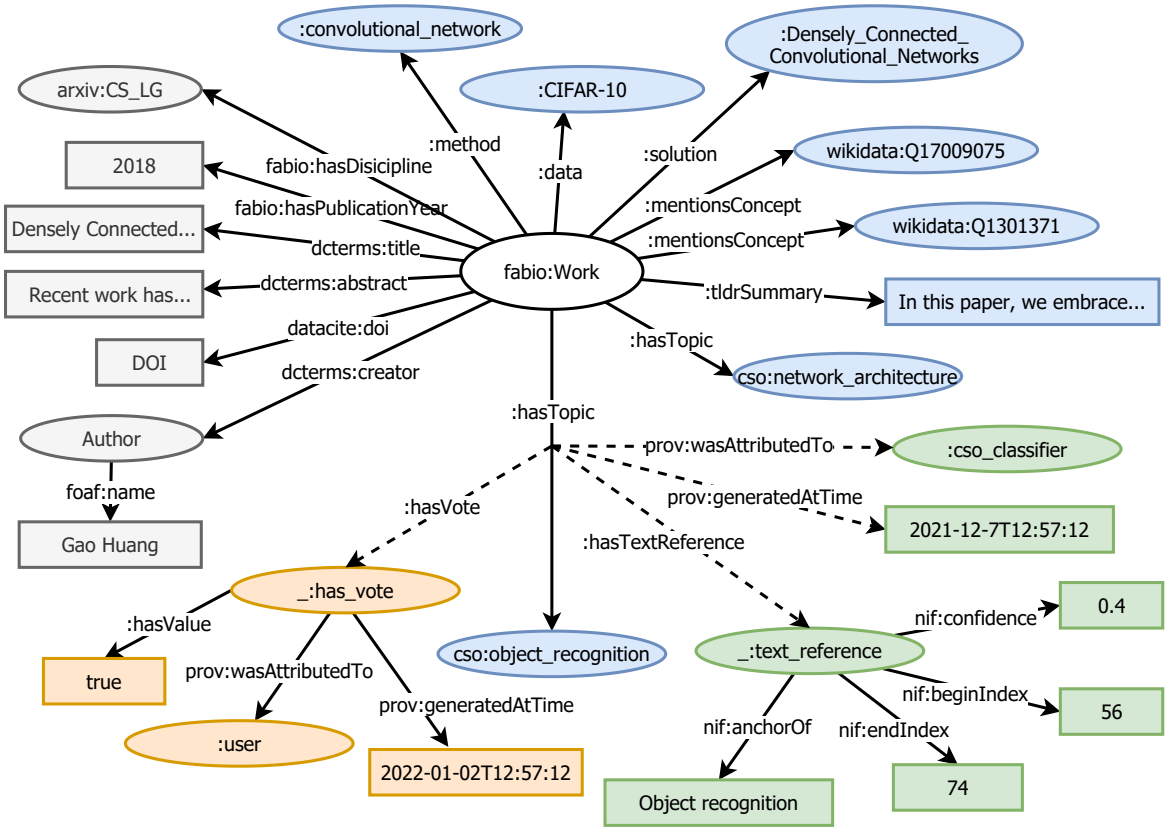

Figure 9.4: Example of paper subgraph including provenance data. Grey nodes represent metadata related to the work. Blue nodes indicate NLP-generated knowledge for the respective paper. The dashed lines represent provenance data for the statement. In this example, green nodes indicate provenance data related to the context (i.e., the explanation of how to NLP tool came to this result). The orange nodes represent the data for a single user vote on the statement.

and is designed to be integrated into a scholarly knowledge web platform. Each NLP tool has a different question template, as listed in Table 9.1. This question template is used to display the microtask in the widget. The widget itself displays the context required to make an informed decision about the correctness of the statement. In most cases, the context displays an excerpt of the abstract and highlights the words used by the NLP tool to extract the data. Finally, users are able to vote about the correctness. A vote can either be correct, incorrect, or unknown. After a user has voted, a positive affirmation (e.g., “Good job!” or “You rock!”) is displayed to encourage the user to continue with the next statement. The next statement is automatically displayed after voting. Statements are selected in random order and statements are only displayed once to a specific user.

9.2.2 View Paper Page

Figure 9.3 shows a screenshot of the view paper page. It shows how a single paper is displayed when integrated within the ORKG. All data displayed on the page is coming from the TinyGenius knowledge graph and is fetched using SPARQL. The previously discussed voting widget is also displayed on this page. A score is displayed for each listed statement, indicating how reliable a statement is. When hovering over the score a tooltip becomes visible, explaining how the score is determined. This is an aggregation of user votes, counting for 75% of the score, and the system's confidence level, counting for the remaining 25% for the score. By default, statements with a score below a certain threshold (40%) are hidden. Finally, the context tooltip shows statement provenance data. This includes an excerpt from the abstract used by the NLP tool to generate the result. Furthermore, additional data related to the tool, version, and date are displayed. The listed statement resources link to a page that shows the use of the respective resource over the years (see node 6 in Figure 9.3).

9.3 Evaluation

The objective of the evaluation is two-fold. Firstly, we conduct a data evaluation to gather general statistics about our approach and to assess the technical performance. Secondly, we use a sample of the data generated in the first evaluation to conduct a user study. The user study is an exploratory evaluation, and its results are used to guide further development and to assess the feasibility of the approach.

9.3.1 Data Evaluation

We imported the arXiv corpus and processed a subset with selected NLP tools. All articles classified as "Machine Learning" by arXiv[4] are processed. This results in a total amount of 95,376 processed articles, which is approximately 5% of the complete arXiv corpus. We consider this a sizable amount to estimate statistics such as processing time per article, number of extracted statements per article, and to determine the performance of the setup. We chose the machine learning field because several NLP tools are trained specifically on machine learning abstracts. The processing time in seconds per NLP tool is listed in Table 9.2. In addition to the total number of triples, an approximation of the number of metadata and provenance triples is listed. The tools ran on a machine with 40 CPU cores and no dedicated GPUs. As the summarizer tool requires GPUs to run efficiently, we did not apply this tool to the entire dataset. Instead, we ran the summarization tool for the sample of articles used in the user evaluation.

[4] arXiv category: Machine Learning (cs.LG)

Table 9.2: Overview of the data evaluation statistics.

Description	**Measure**
General statistics	*Number*
Processed articles	95,376
Triples metadata	1,521,492
Triples provenance	47,595,706
Triples total	65,608,902
Average number of triples per article	688
Processing time	*Seconds*
CSO classifier	27,803
Ambiverse NLU	137,060
Abstract annotator	62,056
Title parser	87
Summarizer	N/A

9.3.2 Performance Evaluation

To determine the performance of the triple store with the ingested data, we now present three prototypical SPARQL queries and their respective execution time. The queries also demonstrate how data can be accessed via our data model, as outlined in Figure 9.4. The previously presented user interface uses the listed queries to render the paper data and, therefore, the queries are representative for use in an actual system.

The queries are executed on the same machine as used for the NLP processing. Furthermore, the same data is used to query data as listed in Table 9.2 (i.e., 65,608,902 triples). Query 9.1 demonstrates how statements can be retrieved based on the NLP tools used to generate them. Query 9.2 queries all available data for a single paper. A similar query is used in the user interface to display paper data. Finally, Query 9.3 counts all articles that are related to a specific resource, grouped by year. The plotted result of this query is displayed in Figure 9.3 node 6. The resulting execution time indicates that the triple store performs well, even for more complex queries such as aggregating data from all papers in the graph.

9.3.3 User Evaluation

We created an online evaluation environment to evaluate the TinyGenius approach. We focused on evaluating the voting widget, specifically targeting the microtasking aspect of our approach.

```
SELECT DISTINCT * WHERE {
    <<tinygenius:1802.01528  ?pred  ?obj>>
                dcterms:creator  tinygenius:ambiverse_nlu .
    tinygenius:ambiverse_nlu  dcterms:hasVersion  "1.1.1" .
}
```

Query 9.1: Select all statements created by "Ambiverse NLU" version "1.1.1" and its corresponding provenance data. Execution time: 0.1s. Retrieved triples: 23.

```
SELECT DISTINCT * WHERE {
    <<tinygenius:1608.06993  ?pred  ?obj>>  ?provPred  ?provObj .
    OPTIONAL {
        ?provObj  ?provPred2  ?provObj2 .
    }
}
```

Query 9.2: Select all statements related to a single paper, including provenance data. Optionally, include nested provenance data. Execution time: 0.2s. Retrieved triples: 653.

```
SELECT ?year (COUNT(DISTINCT ?paper) AS ?count) WHERE {
    ?paper  a                          fabio:Work ;
            fabio:hasPublicationYear   ?year ;
            ?predicate                 tinygenius:artificial_neural_network .
} GROUP BY ?year
```

Query 9.3: Count the numbers of papers that are related to the "Artificial neural network" resource. Group the results by year. Execution time: 0.4s. Retrieved triples: 15.

Experimental Setup

In total, we recruited 11 participants. All participants are researchers with a Computer Science background. Participants were asked to visit the online evaluation environment which guided them through the evaluation. An interactive help guide explained the objectives of the evaluation and what was expected from the participants. Additionally, several interface components were highlighted and explained in more detail. The appearance of the evaluation interface mimics the design of the scholarly platform where we plan to integrate TinyGenius. Interface components not needed for the evaluation were either disabled or hidden. This is to ensure participants are not leaving the page and potentially not finishing the study.

Participants were asked to validate 114 statements coming from ten different articles. These articles were sampled from the dataset generated in the data evaluation section. The ten most popular articles from this set are selected for evaluation[5]. The rationale for

[5] Top 10 based on popularity according to `http://www.arxiv-sanity.com/top?timefilter=alltime&vfilter=all`

selecting popular articles is that those articles are likely to contain relevant knowledge, hence the popularity of the articles. From the selected articles, statements were randomly selected and limited to three statements per NLP tool. The random statement selection simulates a realistic scenario where NLP tools generate statements that are possibly clearly wrong, clearly right, or ambiguous and therefore hard to assess for correctness. Notably, the participants were not selected based on their knowledge of machine learning. Our assumption is that this knowledge is not required to perform the majority of the microtasks as most tasks consist of relatively simple questions that do not require deep domain knowledge.

After participants completed the microtasks, they were asked to fill out a questionnaire. This questionnaire consisted of 35 questions of which most are answered with a 5-point Likert scale, ranging from "strongly disagree" to "strongly agree". The questionnaire has several objectives, including determining the attitudes towards the overall voting approach, assessing how participants feel about the specific microtasks, and gathering additional feedback. Furthermore, the questionnaire contained two standardized evaluation methods. System Usability Scale (SUS) [181] questions are included to determine the usability of the voting widget interface. Additionally, questions from the NASA Task Load Index (TLX) [191] are included to assess the perceived task load by the participants.

Evaluation results

A set of answers from the questionnaire is plotted in Figure 9.5. The first five questions are related to the five NLP tools. As can be observed from the results, microtasks related to the CSO classifier (question 1), Summarizer (question 2), and Ambiverse NLU (question 4) are considered relatively straightforward by the participants. On the other hand, microtasks related the Abstact annotator (question 3) and Title parser (question 5) are considered more difficult. Most likely this type of task requires more domain knowledge, and possibly more knowledge about the actual article, as participants have to decide whether a certain term is correctly classified. Furthermore, the results suggest that most participants had sufficient context to answer questions and that the "View paper PDF" button, and especially the "View context" feature, were appreciated (question 8 and 9 respectively).

The results from the SUS evaluation are displayed in Figure 9.6. The average SUS score is 78.18 (SD = 11.68) which is considered "good". The TLX outcomes are shown in Figure 9.7. The average task load is 33.79 (SD = 17.43), which is low compared to the average of 45.29 determined by Grier [193] (lower is better). The standard deviation is relatively high, indicating that some participants considered the task more demanding than others. This also becomes apparent from the question related to the time needed to finish the evaluation. While some indicated to be finished within 10 to 20 minutes, others needed considerably more time, between 30-60 minutes, and one participant more than 60 minutes. The perceived machine learning knowledge of participants is displayed in Figure 9.5 question 11. However, the time required to finish the task does not seem to correlate with their knowledge of machine learning.

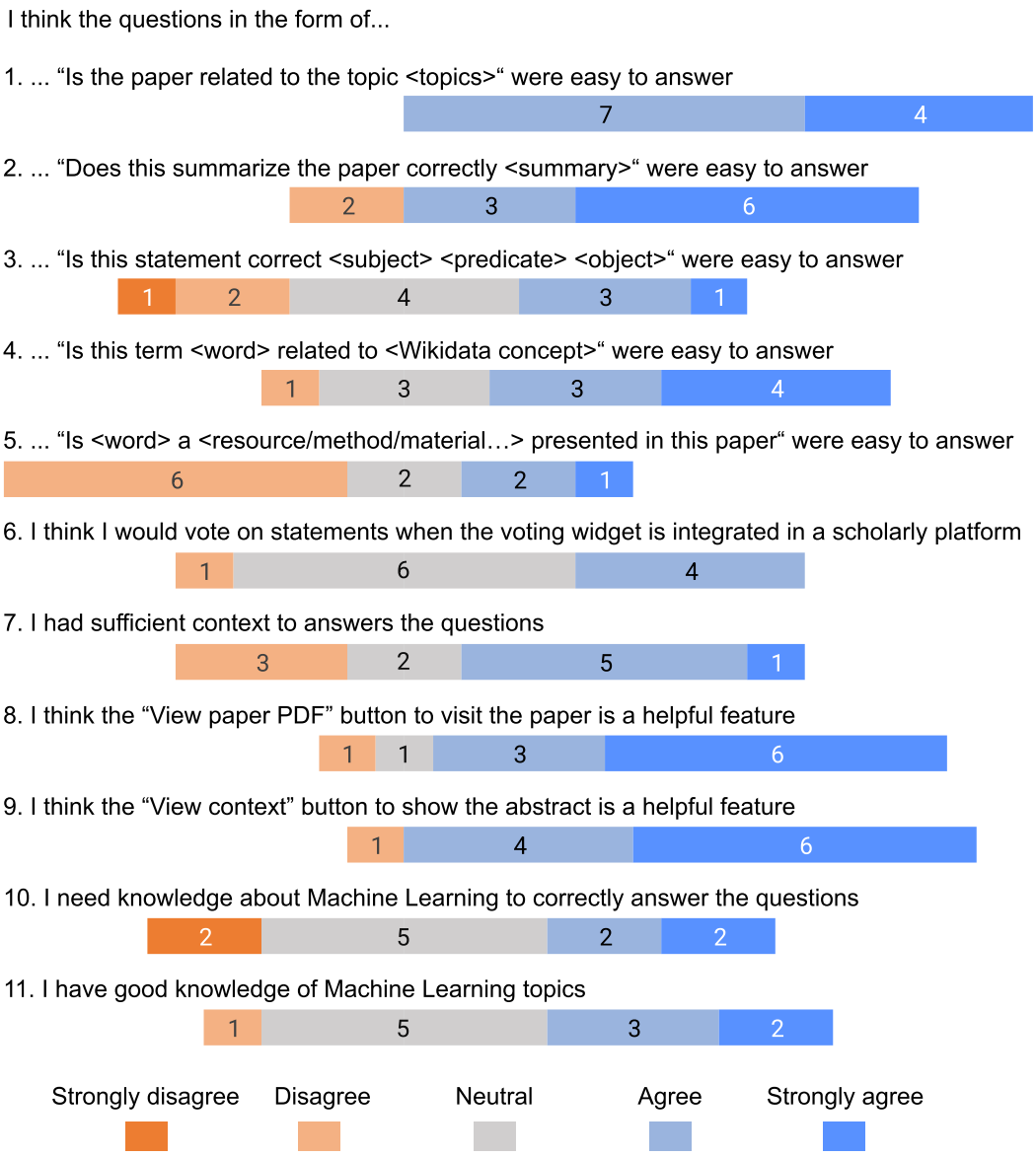

Figure 9.5: Questionnaire results from the questions with a Likert scale. The first five questions relate to the specific NLP tools. The remaining questions are either about the use of the voting widget or the participants’ knowledge of machine learning.

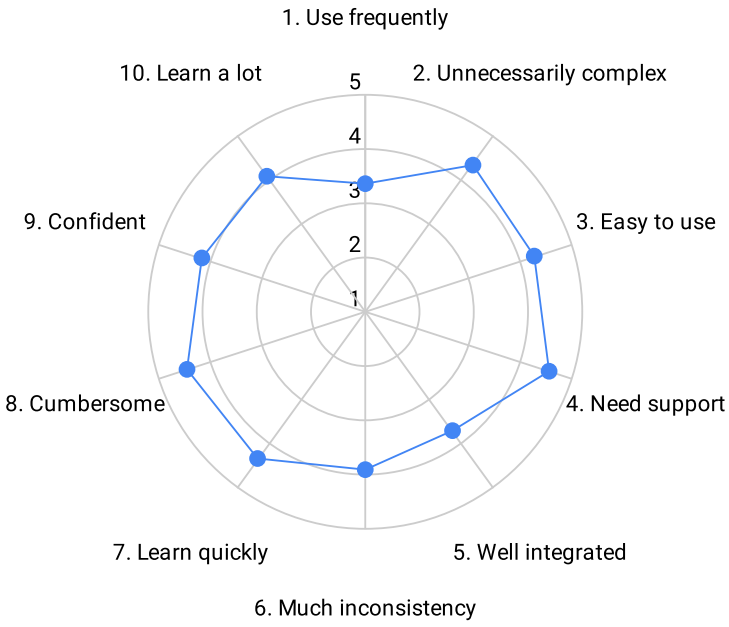

Figure 9.6: Outcomes of the System Usability Scale (SUS) questions. Questions are abbreviated, the full list of questions is provided by Brooke et al. [181]. Answers are normalized so that higher scores mean better usability (SUS uses alternating positive and negative questions to reduce response bias).

Finally, we evaluated the voting data produced by the participants. In total, 1,254 votes were collected. From these, 122 votes were "unknown", meaning that participants were not sufficiently confident to vote. To assess the agreement among participants, we determined the inter-rater reliability from the voting data. The results are listed in Table 9.3. Specifically, we calculated Krippendorff's alpha [205], which is used as a reliability coefficient when there are multiple observers (i.e., participants) and missing data. In our case, the "unknown" vote is considered missing data. The statistic ranges from -1 to 1, where 1 means unanimous agreement, 0 no agreement apart from chance, and -1 means inverse agreement [206]. We calculated the agreement per NLP tool. Interestingly, a relatively large difference between the tools can be observed. More agreement is found for the CSO classifier and Ambiverse NLU, and less agreement for the Abstract annotator and Title parser. This is in line with the results from the participants' own judgments related to the difficulty per NLP tool (questions 1 to 5 in Figure 9.5). The summarization tool has a negative agreement, indicating that this type of task in its current form is not producing meaningful results.

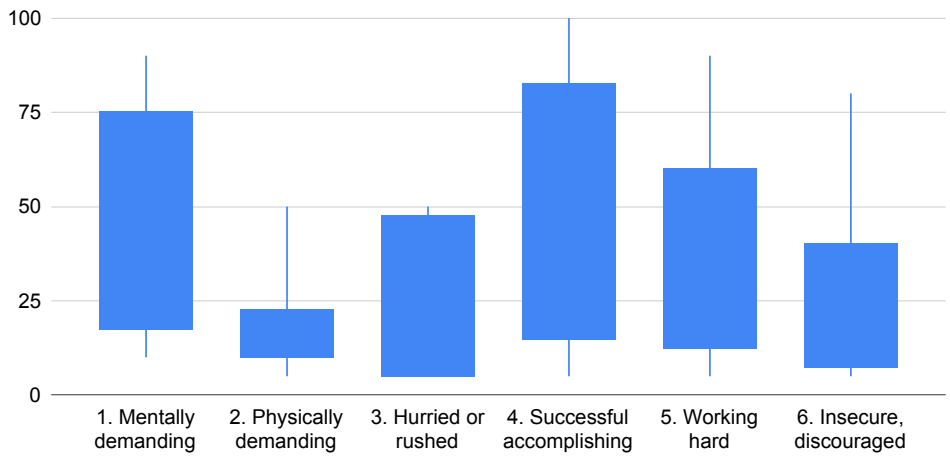

Figure 9.7: Outcomes of the NASA Task Load Index (TLX), using non-weighted questions. Higher values indicate more task load. Line endings represent minimum and maximum values. Boxes represent the first and third quartile.

9.4 Discussion

The evaluation results indicate that the presented method is promising and the proposed setup and infrastructure are suitable for the task. When the methodology is deployed in a real-life setting, the knowledge graph quality can be substantially improved. Over time, more visitors will vote on the presented statements, increasing the overall data accuracy. The user votes are stored as provenance data on the statement level, providing the opportunity for downstream applications to decide how to incorporate the validation data. Incorrect data can simply be filtered out, but it is also possible to perform more complex analysis on the validation data.

The data and performance evaluations show that the current setup performs well and is able to handle the scale of the knowledge graph without major issues. Naturally, more complex queries will result in increased execution time, especially when the knowledge graph grows in size. However, we limited our performance evaluation by running queries that are needed to render the user interface. One of the requirements for web applications is that loading times should be low, preferable below two seconds, which is considered a tolerable waiting time for web users [207]. As the evaluation results indicate, it is indeed possible to load the page within this time frame. Here, we specifically focused on machine learning articles from the arXiv corpus. Some of the selected NLP tools are domain models, specifically trained on Computer Science. However, our approach is not limited to this domain. By design, the system is modular and can be generalized to support other domains and NLP tools. Future work will focus on importing the complete arXiv corpus, which increases the number of triples approximately tenfold.

Table 9.3: Aggregated results from the voting task, grouped by NLP tools. Krippendorff's alpha indicates the agreement among the participants.

Tool name	**Krippendorff's α**	**Number of votes**
CSO classifier	0.31	330
Ambiverse NLU	0.36	330
Abstract annotator	0.021	330
Title parser	0.14	154
Summarizer	-0.032	110

The user evaluation indicated that the usability of the voting widget is good. This finding is also supported by the additional results from the questionnaire. Due to the low number of participants, no statistical conclusions can be drawn from the results. However, the System Usability Scale (SUS) is a reliable statistic for small sample sizes [194]. Furthermore, the homogeneity of the population (all participants had a Computer Science background), might make further evaluation necessary but we deem that this is a realistic setup, since the micotasks can be allocated to crowd-workers with a respective background. However, the preliminary user evaluation gives an impression of the overall approach and guides further development. The inter-rater reliability outcomes are relatively low. However, this is expected as annotators were not trained and had only little information on how to perform the task. If more extensive annotation guidelines were provided, the agreement among annotators is expected to increase. However, this goes against the principle of having low-context and easy-to-perform microtasks. Furthermore, the agreement seems to also depend on the type of microtask. Entity linking and topic modeling tasks are arguably more straightforward than named entity recognition tasks, which are generally more ambiguous and therefore harder to evaluate. Furthermore, the summarization task seems unsuitable for our microtask. Often, a summary is not considered completely wrong or right, which makes it unsuitable for a binary voting task.

The generated knowledge graph provides opportunities for multiple data consumption use cases. For example, by linking concepts between articles, scientometrics can be conducted on the data. This includes methods to plot research trends over time or to find related papers by means of commonly used materials and methods. By making the data accessible via SPARQL, we provide a powerful interface to support such use cases. Query 9.3 is an example for research trend analysis. Other use cases include data exploration interfaces, such as a dynamic faceted search to more effectively find research articles. Due to the availability of structured data, it becomes possible to perform precise search queries. Implementing data consumption use cases is out of scope for this work presented in this chapter.

The current setup implements the voting widget within a scholarly knowledge graph infrastructure. However, we envision that the widget can be implemented within third-party systems as well. For example, arXiv provides a section "arXivLabs" where additional

information related to a publication can be displayed. This section is suitable for TinyGenius related data as well, providing opportunities to collect additional user votes. Furthermore, data can be collected in a casual microtasking setting [208]. For example, data can be collected via Twitter where questions are asked in Tweets and answers can be provided via comments. Although the voting setup will be different than presented within our work, the same underlying knowledge graph and data model can be used.

9.5 Summary

We presented TinyGenius, a methodology to validate NLP statements using microtasks. We applied and evaluated TinyGenius in the scholarly domain. The method combines machine and human intelligence resulting in a synergy that utilizes the strengths of both approaches. Firstly, a set of NLP tools is applied to a corpus of paper abstracts. Afterwards, the resulting data is ingested in a scholarly knowledge graph. Finally, the data is presented to users in the form of microtasks. By utilizing microtasks, the data is validated using human intelligence. The performance evaluation indicated that the used triple store is able to handle the data quantity without issues. The user evaluation showed that the usability of the system is good. We deem this work to be one of the first, which truly combines human and machine intelligence for knowledge graph creation and curation. This combination needs much more attention since there are many important use cases, where machine intelligence alone can (due to the missing training data) not produce useful results.

We addressed **RQ3** by intertwining NLP and crowdsourcing to populate and curate a scholarly knowledge graph. Compared to the machine-in-the-loop approach from Chapter 8, we used a human-in-the-loop approach for TinyGenius. Both approaches combine human- and machine-intelligence, albeit on a different level. For the former approach, the process is initiated by a human and the human is supported by machines. It is the opposite for the latter approach, where the machine initiates the process (in the TinyGenius case, autonomously processes paper abstracts) and is supported by humans. **Challenge 1** relates to this chapter, as the objective was to transform unstructured paper abstracts into structured knowledge. We employed a set of NLP tools to perform this process. Furthermore, **Challenge 4** is related because the task should be simple but still generate high-quality data. One of the key aspects of the TinyGenius approach is to provide simple microtasks to address a complex challenge.

CHAPTER 10

Conclusion

Scholarly knowledge communication today remains mainly document-based, generally sharing knowledge in narrative article form. This way of knowledge sharing is the de facto standard today, as it has been over the last several centuries. This hinders the machine-actionability of the presented knowledge, making it particularly difficult for machines to interpret the knowledge. These challenges affect researchers across all research domains and are indeed a threat to science in general. With the ever-increasing publication rate the ability of researchers to discover new publications, or to find relevant related literature, is severely limited. In this thesis, we introduced an approach to communicate scholarly knowledge in a structured and semantic manner. Our approach addresses the weaknesses of current scholarly communication practices and provides an outlook on the future of scholarly knowledge sharing. At the core of our approach, we leverage knowledge graph technologies to provide structured and machine-actionable representations of scholarly knowledge.

We first presented the technical infrastructure that serves as the foundation of our approach. Together with the user interface components, it forms the Open Research Knowledge Graph (ORKG), an online collaborative scholarly knowledge-sharing platform (cf. Chapter 4). Afterwards, we discussed several methods related to comparable scholarly knowledge. This includes approaches to generate FAIR literature comparisons, to semi-automatically populate these comparisons, and to finally reuse the comparisons within dynamic literature reviews (cf. Chapter 5, Chapter 6, and Chapter 7). Finally, we focused on the crowdsourcing aspects of our approach. This includes approaches to leverage the crowd for sentence annotation and data validation (cf. Chapter 8 and Chapter 9). We now discuss the research questions in more detail and explain how the questions are addressed by the different chapters. Afterwards, we discuss the future work and limitations.

10.1 Discussion of Research Questions

Each research question is discussed in detail in the respective chapters. We now give an overview of how the questions are addressed.

> **RQ1:** How to organize scholarly knowledge using a manually curated scholarly knowledge graph?

The first research question is considered the main question of this thesis. The question addresses the weaknesses of scholarly communication by introducing knowledge graphs to the scholarly domain. Specifically, the research question includes the manual (i.e., human) curation of the graph, which is a key aspect of our approach. Each chapter within this thesis contributes to answering this question, but Chapter 4 in particular presents a technical solution to address this question. On the technical level, we distinguish between frontend and backend. From the backend side, separate functionalities are represented in different microservices. This modular setup facilitates code contributions and makes the ORKG an extensible infrastructure, which can be tailored toward specific use cases.

The graphical user interface of the ORKG plays a crucial role to enable non-expert researchers to generate structured research descriptions. The interface hides complexities related to data modeling and tries to avoid jargon (i.e., semantic web terminology). While the user fills out forms in the interface, in the background more complex graph structures are instantiated. Additionally, tooltips are provided when users need more guidance and a help center with user documentation is available. To ensure the ORKG is adopted in standard research workflows, it is important to keep the overhead low and the learning curve gentle. Finally, the data model is an important aspect of our approach. The data model is flexible and enables researchers to describe research contributions with their desired terminology and ontologies. Although we provide a basic ORKG vocabulary, the flexibility allows describing research data from any research domain. This approach is based on a community effort to create ontologies that work best for specific use cases. The creation of more formal small-scale ontologies is accomplished with ORKG templates. The templates are generally created by content curators, which are experts in a specific research domain. In turn, the templates are used by regular content creators to describe their research contributions. Content creators are assisted by AI-powered tools. These tools have a supportive role as they are mainly providing suggestions to assist researchers during the content creation process.

> **RQ2:** How to generate machine-actionable and comparable overviews of related literature?

The second research question focuses on the comparability of scholarly literature. This includes examining related literature, positioning work to existing efforts, and assessing

the developments of research problems and domains. These activities are all related to the availability of overarching terminology to represent the knowledge within research articles. Additionally, when this data is represented in a machine-actionable manner, innumerable novel use cases emerge. In Chapter 5 we present an approach to generate literature reviews based on data from the knowledge graph. We specifically focus on the FAIR aspects of the published comparisons, which results in machine-actionable literature comparisons. The comparison tool provides functionalities to publish data and assign persistent identifiers to the data. This makes the tool particularly suitable to be integrated into research workflows.

In Chapter 6 we present a workflow to populate the knowledge graph with existing comparisons. The approach takes existing literature reviews which are essentially already providing semi-structured overviews of literature albeit in a format that is not readable for machines. Therefore, this approach focuses on the extraction of these overviews from PDF files. This includes the extraction of the tables from review articles, as well as resolving references to form a paper-centric knowledge graph. Finally, in Chapter 7 we demonstrate how the comparisons can be used within dynamic literature reviews (i.e., SmartReviews). SmartReviews address the main limitations of traditional reviews articles. The knowledge is stored in a knowledge graph, making the data machine-actionable. Furthermore, the dynamic nature of the articles provides novel workflows, such as updating the article once new research becomes available.

RQ3: How to intertwine machine and human intelligence for populating and curating a scholarly knowledge graph?

The third research question relates to combining human and machine intelligence for knowledge graph curation. This includes the process of converting unstructured to structured data as well as validating existing data in the graph. The human intelligence aspect is mainly provided in a crowdsourced setting, while the machine intelligence is provided by Natural Language Processing (NLP) tools. We distinguish between two different approaches for intertwining human and machine intelligence, a machine-in-the-loop and a human-in-the-loop approach. Chapter 8 takes the former approach, a user interface is presented where researchers are requested to annotate sentences within their publications with a preselected set of discourse classes. During this process, humans are assisted by NLP tools to improve their experience and speed up the process. Hence the machine-in-the-loop approach because the human has the initiative and is the main actor in the annotation process. Crucial for this approach is the non-intrusive integration of NLP suggestions. At all times, users can decide to ignore recommendations when considered irrelevant or incorrect.

In Chapter 9, we present a human-in-the-loop approach to accomplish human and machine collaboration. In this approach, the machine is the main actor as it autonomously processes large quantities of scholarly text via NLP tools. In turn, users can vote on the correctness of the extracted knowledge. The approach leverages microtasks in a crowdsourced

setting, where small contributions of individual users contribute towards the larger task of validating the knowledge graph as a whole. The two different approaches demonstrate how humans and machines can complement each other, taking the strengths of both approaches while eliminating the weaknesses. Humans are particularly good at creating high-quality data and judging the correctness of extracted knowledge, while machines are able to scale to process large quantities of data.

10.2 Limitations

The presented work addresses multiple challenges within scholarly communication. Due to the scope of the issues scholarly communication is facing, it is practically impossible to provide solutions that have a guaranteed impact on scholarly communication. What we did instead, is outline how to future of scholarly communication can look like and provide the necessary tools to support this vision. Most of the proposed approaches and tools have been validated by means of user evaluations. Most of these evaluations were conducted with relatively small sample sizes and therefore resulted not in statistically significant results. It is relatively hard to recruit participants with an academic background, which was a requirement for most of the evaluations. This also means that such evaluations are less suitable to conduct with crowdsourcing platforms, such as Amazon Mechanical Turk. However, despite the small sample sizes, we were able to gather valuable feedback about our methods, and make conclusions about the approaches and usability of the tools. Also, we tailored our evaluation approaches towards smaller sample sizes, including open-ended questions and measures such as the System Usability Scale (SUS) and NASA Task Load Index (TLX). These measures have proven to be effective, even when applied on smaller sample sizes.

While it is common practice to evaluate the usability of user interfaces, only when the approaches are part of the research lifecycle, one can assess the true impact of our work. This applies even more so to the availability of high-quality structured scholarly knowledge. Tools such as the ORKG comparisons rely on structured data and will have a true impact when such data is available. Although we focused both on data generation and data consumption in this thesis, we had to assume researchers adopt our approach in order to estimate the potential impact. The process of evaluating the impact of the ORKG is an ongoing process that will continue for the years to come. This also applies to the development of the tools presented in this thesis. Although the presented foundations of the approach will remain the same, implementations will change over time to incorporate users' needs and demands.

10.3 Future Work and Closing Remarks

Finally, we provide a list of future work directions. The future work builds upon the tools and methods presented in this thesis and incorporates the lessons learned from both the user evaluations and real-world usage of the ORKG.

Improved external ontology integration. Data ingested in the ORKG is structured by design. The structured data contributes to machine-actionable scholarly knowledge. To further improve machine-actionability, data can be described using existing ontologies. Currently, there is support for mapping classes, resources, and properties to external ontologies. Since the reuse of existing vocabularies is a crucial aspect to increase machine-actionability, we plan to integrate third-party ontologies more seamlessly within the user interface. First, by implementing the Ontology Lookup Service (OLS) [209] tool within the frontend. Users will be able to select terms from a variety of ontologies. Additionally, we plan to integrate the Wikidata [40] API. Wikidata is a collaboratively created knowledge graph containing generic and common knowledge. Therefore, generic terms within Wikidata are of potential interest to reuse within the ORKG.

Self-sustained observatories. The ORKG already organizes smaller researcher communities in *observatories*, which are groups of researchers with similar research interests. These observatories are a crucial part of engaging individual research communities to contribute to the ORKG. In the near future, we strive to provide observatory members with improved data creation and curation features to accommodate the needs of their specific domains. Additionally, observatories lead the creation of domain-specific ORKG templates. We, therefore, plan to provide observatories with improved template creation tools and let them manage these templates from within their observatory. Finally, we plan to integrate a public discussion board for observatories to encourage open discussions regarding the organization and creation of the data within an observatory.

Provide additional user incentives. In order to get researchers on board and to encourage them to become ORKG contributors, we have to provide clear benefits of using the ORKG. Therefore, future work will focus on engaging researchers and motivating them to contribute to more structured scholarly communication. We strive to improve user incentives in three ways: highlight use cases, acknowledge users, and provide badge rewards. First, by providing domain-specific use cases, we can demonstrate how the ORKG can be used, and show the advantages for researchers. Second, we plan to integrate reward systems for contributors. Currently, users are already publicly acknowledged when contributing by showing their names and profile photos. We want to extend this by improving the user profile page, listing all their contributions and their activities throughout the system. Last, we plan to introduce a reward system where users are able to earn badges. Badge systems

have proven to be effective to create strong incentives for users to contribute, for example on social media sites [210]. ORKG badges can be earned by performing specific tasks or reaching a certain number of contributions.

More NLP-extracted knowledge. Furthermore, future work will continue to explore new methods to transform unstructured into structured knowledge. Given that machine learning methods are rapidly developing, these tools will presumably get a more prominent role in the transformation process. However, users will continue to play an important role in validating machine learning data for the foreseeable future. We plan to extend our *machine-in-the-loop* approaches (i.e., where a machine assists humans while creating data) with several tools. With respect to ORKG comparisons, as presented in Chapter 5, we aim to use additional machine learning methods to automatically complete comparisons with missing literature. Additionally, comparison data can be extracted with the help of NLP tools, making it less time-consuming to create literature overviews. The more scholarly knowledge the ORKG contains, the more powerful these methods become. Therefore, we are strongly focusing on getting a more comprehensive knowledge base. Additionally, we plan to extend our *human-in-the-loop* approaches (i.e., where a human assists a machine) so we can extract knowledge at scale. A crucial aspect of such approaches is the storage of provenance data, clearly indicating where data is coming from and what is the accuracy of the data. With such provenance data, the knowledge graph can be updated if models improve over time. Additionally, the accuracy helps users decide whether the data is suitable for their use case or not.

Enhanced curation features. After data is created and ingested in the ORKG, it needs to be curated. In the future, new curation tools should assist human curators in this process. For example, violations of template property domains should be clearly indicated and potential fixes should be provided. Additionally, with the help of quality and maturity models, it should become possible for users to get active feedback on how to improve their structured knowledge descriptions. A further set of tools should make it easier for curators to merge duplicate properties and resources, provide feedback to users regarding modeling, and organize ORKG content types (such as papers, comparisons, and reviews). In addition to human curation, we plan to explore the option of automated curation using the concept of *bots*. Such bots are similar to bots in Wikidata, which are responsible for a large share of the overall number of contributions to Wikidata [211]. For example, bots could be able to automatically find integrity violations when templates are used, or automatically complement missing (meta)data, such as DOIs or ORCIDs for papers and authors respectively. These new curation features contribute to a high-quality knowledge graph and provide ways to curate the graph as it continues to grow in size.

Miscellaneous. At last, we discuss several potentially interesting research directions without concrete implementation plans. Firstly, exploring *gamification* approaches for scholarly knowledge graph creation can help to provide additional user incentives to contribute to the graph. Gamified tasks can include validation and curation of data, possibly within a certain time limit. Points can be earned by answering questions correctly. Correctness can be determined based on the majority of votes from other users. Secondly, we want to explore the possibility to link extraction knowledge from papers to their textual representation in the original article. This data will be stored as provenance data on the statement level. We envision a PDF viewer (based on our existing approach presented in Chapter 8) that is able to highlight the text snippets within PDF articles to show where the data is coming from. This additional provenance information contributes to the credibility and trustworthiness of the content within the graph. Finally, we want to improve template support within the UI, having dynamically generated UI components based on the used templates. For example, when a template is used to describe units, input forms can be displayed showing an entry field for a value and a dropdown of existing units. This simplifies the data entry process for users, as they are not concerned about the modeling, but only about entering the data itself.

The majority of the presented methods and tools have been implemented in the ORKG already. This means that the tools are available to the public and can have an immediate impact on research workflows. However, in order to break free from the current scholarly knowledge-sharing practices, more effort is required. The technical foundation is laid by means of the ORKG infrastructure. But, apart from the technical changes, there needs to be a behavioral change as well. This means researchers have to change their habits and mindsets. Arguably, behavioral change is an even more complex endeavor than the technical challenges. This thesis provides a starting point to make our vision of semantic scholarly communication a reality.

APPENDIX A

List of Publications

1. **Oelen, A.**, Van Aart, C., & De Boer, V. (2018, May). Measuring surface water quality using a low-cost sensor kit within the context of rural Africa. In Perspectives on ICT4D: Proceedings of the 5th International Symposium "Perspectives on ICT4D" co-located with 10th ACM Web Science Conference (WebSci'18).

2. Jaradeh, M. Y., **Oelen, A.**, Prinz, M., Stocker, M., & Auer, S. (2019, August). *Open research knowledge graph: a system walkthrough.* In International Conference on Theory and Practice of Digital Libraries (pp. 348-351). Springer, Cham.

3. Jaradeh, M. Y., **Oelen, A.**, Farfar, K. E., Prinz, M., D'Souza, J., Kismihók, G., Stocker, M., & Auer, S. (2019, September). *Open research knowledge graph: next generation infrastructure for semantic scholarly knowledge.* In Proceedings of the 10th International Conference on Knowledge Capture (pp. 243-246).

4. **Oelen, A.**, Jaradeh, M. Y., Farfar, K. E., Stocker, M., & Auer, S. (2019, November). *Comparing research contributions in a scholarly knowledge graph.* In CEUR Workshop Proceedings 2526 (Vol. 2526, pp. 21-26).

5. **Oelen, A.**, & Auer, S. (2019, December). *Content Authoring with Markdown for Visually Impaired and Blind Users.* In 2019 IEEE International Symposium on Multimedia (ISM) (pp. 285-2855). IEEE.

6. **Oelen, A.**, Jaradeh, M. Y., Stocker, M., & Auer, S. (2020, August). *Generate FAIR Literature Surveys with Scholarly Knowledge Graphs.* In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020 (pp. 97-106).

7. Elias, M., **Oelen, A.**, Tavakoli, M., Kismihok, G., & Auer, S. (2020, September). *Quality Evaluation of Open Educational Resources.* In European Conference on Technology Enhanced Learning (pp. 410-415). Springer, Cham.

8. **Oelen, A.**, Stocker, M., & Auer, S. (2020, November). *Creating a Scholarly Knowledge Graph from Survey Article Tables.* In International Conference on Asian Digital Libraries (pp. 373-389). Springer, Cham.

9. Auer, S., **Oelen, A.**, Haris, M., Stocker, M., D'Souza, J., Farfar, K. E., Vogt, L., Prinz, M., Wiens, V., & Jaradeh, M. Y. (2020, December). *Improving Access to Scientific Literature with Knowledge Graphs.* Bibliothek Forschung und Praxis, 44(3), 516-529.

10. **Oelen, A.**, Stocker, M., & Auer, S. (2021, April). *Crowdsourcing Scholarly Discourse Annotations.* In 26th International Conference on Intelligent User Interfaces (pp. 464-474).

11. **Oelen, A.**, Stocker, M., & Auer, S. (2021, September). *SmartReviews: Towards Human- and Machine-actionable Survey Articles.* In International Conference on Theory and Practice of Digital Libraries. (pp. 181-186). Springer, Cham.

12. **Oelen, A.**, Jaradeh, M., Stocker, M., & Auer, S. (2021, November) *Organizing Scholarly Knowledge leveraging Crowdsourcing, Expert Curation and Automated Techniques.* Book: Linking Knowledge: Linked Open Data for Knowledge Organization

13. **Oelen, A.**, Stocker, M., & Auer, S. (2021, December). *SmartReviews: Towards Human- and Machine-actionable Representation of Survey Articles.* In International Conference on Asian Digital Libraries. Springer, Cham.

14. **Oelen, A.**, Stocker, M., & Auer, S. (2022, June). *TinyGenius: Intertwining Natural Language Processing with Microtask Crowdsourcing for Scholarly Knowledge Graph Creation.* In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2022.

APPENDIX B

List of Videos

For demonstration and instruction purposes, several videos are created highlighting some of the user interface features of the Open Research Knowledge Graph. Below, a set of these videos is listed.

1. **Oelen, A.**, *How to make an ORKG comparison - An Example from Virology. Open Research Knowledge Graph (ORKG).* https://doi.org/10.5446/51996
2. **Oelen, A.**, *How to create an ORKG Comparison - An example from Computer Science.* https://doi.org/10.5446/56182
3. **Oelen, A.**, *Demonstration of SmartReviews.* https://doi.org/10.5446/53601
4. Vogt, L.; Auer, S.; **Oelen, A.**; Wiens, V., *Webinar: Introduction to the Open Research Knowledge Graph (ORKG).* https://doi.org/10.5446/52956

Bibliography

[1] F. Bacon, *Meditations Sacrae and Human Philosophy*,
Reprinted in 1996 by Kessinger Publishing (1597) (cit. on p. 1).

[2] United Nations Educational Scientific and Cultural Organization (UNESCO),
Science for Society (2021), URL:
https://en.unesco.org/themes/science-society (visited on 03/11/2021)
(cit. on p. 1).

[3] C. L. Borgman, *Bibliometrics and Scholarly Communication: Editor's Introduction*,
Communication Research **16** (1989) 583, DOI: 10.1177/009365089016005002
(cit. on p. 1).

[4] C. L. Borgman and J. Furner, *Scholarly communication and bibliometrics*,
Annual Review of Information Science and Technology **36** (2002) 2,
DOI: https://doi.org/10.1002/aris.1440360102 (cit. on p. 1).

[5] A. Jinha,
Article 50 million: An estimate of the number of scholarly articles in existence,
Learned Publishing **23** (2010) 258, DOI: 10.1087/20100308 (cit. on p. 2).

[6] M. Mabe et al., *The growth and number of journals*, Serials **16** (2003) 191,
DOI: 10.1629/16191 (cit. on p. 2).

[7] B. Mons and J. Velterop, *Nano-publication in the e-science era*,
CEUR Workshop Proceedings **523** (2009) (cit. on pp. 3, 95).

[8] S. Klampfl, M. Granitzer, K. Jack and R. Kern,
Unsupervised document structure analysis of digital scientific articles,
International Journal on Digital Libraries **14** (2014) 83,
DOI: 10.1007/s00799-014-0115-1 (cit. on pp. 3, 38, 40).

[9] D. Jiang and X. Yang, *Converting PDF to HTML approach based on text detection*,
Proceedings of the 2nd international conference on interaction sciences:
Information technology, culture and human **403** (2009) 982,
DOI: 10.1145/1655925.1656103 (cit. on pp. 3, 41).

[10] T. Page, *Skeuomorphism or flat design: future directions in mobile device User Interface (UI) design education*,
International Journal of Mobile Learning and Organisation **8** (2014) 130
(cit. on p. 4).

[11] D. Cereda et al., *The early phase of the COVID-19 outbreak in Lombardy, Italy*,
arXiv preprint arXiv:2003.09320 (2020) (cit. on p. 5).

[12] B. Mons, *Which gene did you mean?*, BMC Bioinformatics **6** (2005) 142,
DOI: 10.1186/1471-2105-6-142 (cit. on p. 7).

[13] N. Choudhury, *World wide web and its journey from web 1.0 to web 4.0*, International Journal of Computer Science and Information Technologies **5** (2014) 8096
(cit. on pp. 15, 16).

[14] T. O'Reilly, *Web 2.0 compact definition: Trying again*, 2006,
URL: http://radar.oreilly.com/archives/2006/12/web-20-compact.html
(visited on 26/10/2021) (cit. on p. 16).

[15] H. Eijkman, *Academics and Wikipedia: Reframing Web 2.0+ as a disruptor of traditional academic power-knowledge arrangements*,
Campus-wide information systems (2010), DOI: 10.1108/10650741011054474
(cit. on p. 16).

[16] T. Berners-Lee, J. Hendler and O. Lassila, *The semantic web*,
Scientific american **284** (2001) 34, DOI: 10.1038/scientificamerican0501-34
(cit. on pp. 16, 40).

[17] T. Berners-Lee and R. Swick, *Semantic Web Development*, tech. rep.,
Massachusetts Inst of Tech Cambridge, 2006 (cit. on p. 17).

[18] J. Hendler, F. Gandon and D. Allemang, *Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL*, Morgan & Claypool, 2008
(cit. on p. 18).

[19] W. Kuhn, T. Kauppinen and K. Janowicz,
"Linked data-a paradigm shift for geographic information science",
International Conference on Geographic Information Science, Springer, 2014 173,
DOI: 10.1007/978-3-319-11593-1_12 (cit. on p. 18).

[20] L. Ehrlinger and W. Wöß, *Towards a Definition of Knowledge Graphs.*,
SEMANTiCS (Posters, Demos, SuCCESS) **48** (2016) 2 (cit. on p. 19).

[21] M. Kroetsch and G. Weikum, *Special issue on knowledge graphs*,
Journal of Web Semantics **37** (2016) 53 (cit. on p. 19).

[22] M. Färber, F. Bartscherer, C. Menne and A. Rettinger,
Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago,
Semantic Web **9** (2018) 77, DOI: 10.3233/SW-170275 (cit. on p. 19).

[23] T. Berners-Lee and D. Connolly, *Notation3 (N3): A readable RDF syntax*, W3C team submission **28** (2011) (cit. on p. 20).

[24] B. Adida, M. Birbeck, S. McCarron and S. Pemberton, *RDFa in XHTML: Syntax and processing*, Recommendation, W3C **7** (2008) 41 (cit. on pp. 21, 108).

[25] M. Morsey, J. Lehmann, S. Auer and A.-C. N. Ngomo, "DBpedia SPARQL benchmark–performance assessment with real queries on real data", *International semantic web conference*, Springer, 2011 454, DOI: 10.1007/978-3-642-25073-6_29 (cit. on p. 22).

[26] M. D. Wilkinson et al., *Comment: The FAIR Guiding Principles for scientific data management and stewardship*, Scientific Data **3** (2016) 1, DOI: 10.1038/sdata.2016.18 (cit. on pp. 23, 26, 27, 34, 40, 61, 75, 83, 131).

[27] T. K. Das and P. M. Kumar, *Big data analytics: A framework for unstructured data analysis*, International Journal of Engineering Science & Technology **5** (2013) 153 (cit. on p. 23).

[28] T. Berners-Lee, *Design Issues: Linked Data* (2006), URL: https://www.w3.org/DesignIssues/LinkedData.html (visited on 26/10/2021) (cit. on pp. 24, 25).

[29] S. Auer et al., *DBpedia: A Nucleus for a Web of Open Data*, The semantic web (2007) 722, DOI: 10.1007/978-3-540-76298-0_52 (cit. on pp. 26, 33).

[30] A. Hasnain and D. Rebholz-Schuhmann, "Assessing FAIR data principles against the 5-star open data principles", *European Semantic Web Conference*, Springer, 2018 469, DOI: 10.1007/978-3-319-98192-5_60 (cit. on p. 26).

[31] D. C. M. Initiative et al., *Dublin core metadata element set, version 1.1* (2012) (cit. on p. 28).

[32] B. Mons et al., *Cloudy, increasingly FAIR; Revisiting the FAIR Data guiding principles for the European Open Science Cloud*, Information Services and Use **37** (2017) 49, DOI: 10.3233/ISU-170824 (cit. on pp. 29, 34, 78, 105).

[33] R. Kahn and R. Wilensky, *A framework for distributed digital object services*, International Journal on Digital Libraries **6** (2006) 115, DOI: 10.1007/s00799-005-0128-x (cit. on p. 29).

[34] U. Schwardmann, *Digital Objects–FAIR Digital Objects: Which Services Are Required?*, Data Science Journal **19** (2020), DOI: 10.5334/dsj-2020-015 (cit. on p. 29).

[35] R. F. D. M. M. W. Group et al.,
FAIR Data Maturity Model: specification and guidelines,
Res. Data Alliance (2020) 2019, DOI: 10.15497/rda00050 (cit. on p. 29).

[36] L. O. Bonino da Silva Santos,
FAIR Digital Object Framework Documentation (Working Draft) (2020),
URL: https://fairdigitalobjectframework.org/ (visited on 02/11/2021)
(cit. on p. 29).

[37] S. Soiland-Reyes et al., *Packaging research artefacts with RO-Crate*,
Data Science (2021) 1, DOI: 10.3233/DS-210053 (cit. on p. 30).

[38] R. V. Guha, D. Brickley and S. Macbeth,
Schema.org: evolution of structured data on the web,
Communications of the ACM **59** (2016) 44, DOI: 10.1145/2844544 (cit. on p. 30).

[39] F. M. Suchanek, G. Kasneci and G. Weikum, *Yago: A core of semantic knowledge*,
16th International World Wide Web Conference, WWW2007 (2007) 697,
DOI: 10.1145/1242572.1242667 (cit. on p. 33).

[40] D. Vrandečić and M. Krötzsch, *Wikidata: A free collaborative knowledgebase*,
Communications of the ACM **57** (2014) 78, DOI: 10.1145/2629489
(cit. on pp. 33, 82, 134, 149).

[41] W. Ammar et al., *Construction of the literature graph in semantic scholar*,
NAACL HLT 2018 - 2018 Conference of the North American Chapter of the
Association for Computational Linguistics: Human Language Technologies -
Proceedings of the Conference **3** (2018) 84, DOI: 10.18653/v1/n18-3011
(cit. on pp. 33, 111).

[42] A. Sinha et al., *An overview of microsoft academic service (MAS) and applications*,
WWW 2015 Companion - Proceedings of the 24th International Conference on
World Wide Web (2015) 243, DOI: 10.1145/2740908.2742839 (cit. on pp. 33, 111).

[43] F. Ronzano, G. C. d. Bosque and H. Saggion,
Semantify CEUR-WS Proceedings: towards the automatic generation of highly descriptive scholarly publishing Linked Datasets,
Communications in Computer and Information Science **475** (2014) V,
DOI: 10.1007/978-3-319-12024-9 (cit. on pp. 33, 113).

[44] A. Ruiz Iniesta and O. Corcho,
"A review of ontologies for describing scholarly and scientific documents",
Proceedings of the 4th Workshop on Semantic Publishing (SePublica)
(Anissaras, Greece, 25th May 2014), CEUR Workshop Proceedings 1155, 2014
(cit. on pp. 33, 34).

[45] A. Hars,
Designing Scientific Knowledge Infrastructures: The Contribution of Epistemology,
Information Systems Frontiers **3** (2001) 63, DOI: 10.1023/A:1011401704862
(cit. on p. 33).

[46] S. Peroni and D. Shotton,
FaBiO and CiTO: Ontologies for describing bibliographic resources and citations,
Journal of Web Semantics **17** (2012) 33, DOI: 10.1016/j.websem.2012.08.001
(cit. on p. 33).

[47] A. Gangemi, S. Peroni, D. Shotton and F. Vitali,
The Publishing Workflow Ontology (PWO), Semantic Web **8** (2017) 703,
DOI: 10.3233/SW-160230 (cit. on p. 33).

[48] S. Peroni and D. Shotton, *The SPAR Ontologies*,
International Semantic Web Conference (2018) 119,
DOI: 10.1007/978-3-030-00668-6_8 (cit. on pp. 33, 117).

[49] A. Constantin, S. Peroni, S. Pettifer, D. Shotton and F. Vitali,
The Document Components Ontology (DoCO), Semantic Web **7** (2016) 167,
DOI: 10.3233/SW-150177 (cit. on pp. 33, 117).

[50] B. Sateli and R. Witte, *Semantic representation of scientific literature: bringing claims, contributions and named entities onto the Linked Open Data cloud*,
PeerJ Computer Science **1** (2015) e37, DOI: 10.7717/peerj-cs.37 (cit. on p. 34).

[51] M. Y. Jaradeh et al., *Open research knowledge graph: Next generation infrastructure for semantic scholarly knowledge*, K-CAP 2019 - Proceedings of the 10th International Conference on Knowledge Capture (2019) 243,
DOI: 10.1145/3360901.3364435 (cit. on pp. 36, 46, 68, 81).

[52] P. Groth, A. Gibson and J. Velterop, *The anatomy of a nanopublication*, **30** (2010) 51,
DOI: 10.3233/ISU-2010-0613 (cit. on p. 36).

[53] T. Clark, P. N. Ciccarese and C. A. Goble, *Micropublications: a semantic model for claims, evidence, arguments and annotations in biomedical communications*, en,
Journal of Biomedical Semantics **5** (2014) 28, DOI: 10.1186/2041-1480-5-28
(cit. on p. 36).

[54] S. Peroni et al., *Research Articles in Simplified HTML: a Web-first format for HTML-based scholarly articles*, en, PeerJ Computer Science **3** (2017) e132,
DOI: 10.7717/peerj-cs.132 (cit. on p. 36).

[55] S. Capadisli et al., *Decentralised authoring, annotations and notifications for a read-write web with dokieli*,
Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **10360 LNCS** (2017) 469,
DOI: 10.1007/978-3-319-60131-1_33 (cit. on pp. 36, 40, 43).

[56] A. S. Corrêa, P. L. P. Corrêa and F. S. C. Da Silva, *Transparency portals versus open government data. An assessment of openness in Brazilian municipalities*, ACM International Conference Proceeding Series (2014) 178, DOI: 10.1145/2612733.2612760 (cit. on p. 34).

[57] A. S. Corrêa and P. O. Zander, *Unleashing tabular content to open data: A survey on PDF table extraction methods and tools*, ACM International Conference Proceeding Series (2017) 54, DOI: 10.1145/3085228.3085278 (cit. on pp. 34, 38, 40, 41).

[58] E. Hyvönen, *Publishing and using cultural heritage linked data on the semantic web*, Synthesis Lectures on the Semantic Web: Theory and Technology **2** (2012) 1, DOI: 10.2200/S00452ED1V01Y201210WBE003 (cit. on p. 34).

[59] E. Mäkelä, E. Hyvönen and T. Ruotsalo, *How to deal with massively heterogeneous cultural heritage data - Lessons learned in CultureSampo*, Semantic Web **3** (2012) 85, DOI: 10.3233/sw-2012-0049 (cit. on p. 34).

[60] M. G. Skjæveland, E. H. Lian and I. Horrocks, *Publishing the Norwegian Petroleum Directorate's FactPages as semantic web data*, International Semantic Web Conference **8219** (2013) 162, DOI: 10.1007/978-3-642-41338-4_11 (cit. on p. 34).

[61] J. Starr et al., *Achieving human and machine accessibility of cited data in scholarly publications*, PeerJ Computer Science **2015** (2015) 1, DOI: 10.7717/peerj-cs.1 (cit. on p. 34).

[62] A. Rodríguez-Iglesias et al., *Publishing FAIR data: An exemplar methodology utilizing PHI-base*, Frontiers in Plant Science **7** (2016), DOI: 10.3389/fpls.2016.00641 (cit. on p. 34).

[63] M. Boeckhout, G. A. Zielhuis and A. L. Bredenoord, *The FAIR guiding principles for data stewardship: Fair enough?*, European Journal of Human Genetics **26** (2018) 931, DOI: 10.1038/s41431-018-0160-0 (cit. on p. 34).

[64] T. D. LaToza, W. B. Towne, C. M. Adriano and A. van der Hoek, "Microtask Programming: Building Software with a Crowd", *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, Honolulu, Hawaii, USA: Association for Computing Machinery, 2014 43, DOI: 10.1145/2642918.2647349 (cit. on pp. 34, 37).

[65] J. Cheng, J. Teevan, S. T. Iqbal and M. S. Bernstein, "Break It Down: A Comparison of Macro- and Microtasks", *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, Seoul, Republic of Korea: Association for Computing Machinery, 2015 4061, DOI: 10.1145/2702123.2702146 (cit. on p. 34).

[66] J. Teevan, D. J. Liebling and W. S. Lasecki, “Selfsourcing Personal Tasks”, *CHI ’14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA ’14, Toronto, Ontario, Canada: Association for Computing Machinery, 2014 2527, DOI: 10.1145/2559206.2581181 (cit. on p. 34).

[67] C. Sarasua, E. Simperl and N. F. Noy, “CrowdMap: Crowdsourcing Ontology Alignment with Microtasks”, *The Semantic Web – ISWC 2012*, ed. by P. Cudré-Mauroux et al., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012 525, DOI: 10.1007/978-3-642-35176-1_33 (cit. on pp. 34, 37).

[68] S. Komarov, K. Reinecke and K. Z. Gajos, “Crowdsourcing Performance Evaluations of User Interfaces”, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’13, Paris, France: Association for Computing Machinery, 2013 207, DOI: 10.1145/2470654.2470684 (cit. on p. 37).

[69] J. C. Chang, S. Amershi and E. Kamar, *Revolt: Collaborative crowdsourcing for labeling machine learning datasets*, Conference on Human Factors in Computing Systems - Proceedings **2017-May** (2017) 2334, DOI: 10.1145/3025453.3026044 (cit. on pp. 37, 43, 115).

[70] B. M. Good and A. I. Su, *Crowdsourcing for bioinformatics*, Bioinformatics **29** (2013) 1925, DOI: 10.1093/bioinformatics/btt333 (cit. on p. 37).

[71] B. V. Wee and D. Banister, *How to Write a Literature Review Paper?*, Transport Reviews **36** (2016) 278, DOI: 10.1080/01441647.2015.1065456 (cit. on pp. 37, 38, 62, 95).

[72] Y. Wolmark, *Quality assessment*, Gerontologie et Societe **99** (2001) 131, DOI: 10.3917/gs.099.0131 (cit. on pp. 37, 95).

[73] M. D. Gall and W. R. Borg, *Educational Research: An introduction (sixth edition)*, White Plains, NY: Longman Publishers USA (1996) (cit. on pp. 37, 62).

[74] C. Hart, *Doing a Literature Review: Releasing the Social Science Research Imagination*, Sage, 1998 1 (cit. on p. 37).

[75] J. Webster and R. T. Watson, *Analyzing the Past to Prepare for the Future: Writing a Literature Review.*, MIS Quarterly **26** (2002) xiii (cit. on pp. 37, 38, 62, 81, 95).

[76] S. Fathalla, S. Vahdati, S. Auer and C. Lange, “Towards a Knowledge Graph Representing Research Findings by Semantifying Survey Articles”, *International Conference on Theory and Practice of Digital Libraries*, Springer, 2017 315, DOI: 10.1007/978-3-319-67008-9_25 (cit. on pp. 37, 39).

[77] S. Vahdati, S. Fathalla, S. Auer, C. Lange and M. E. Vidal, *Semantic Representation of Scientific Publications*, International Conference on Theory and Practice of Digital Libraries **11799** (2019) 375, DOI: 10.1007/978-3-030-30760-8_37 (cit. on pp. 37, 39).

[78] A. Oelen, M. Y. Jaradeh, M. Stocker and S. Auer, *Generate FAIR Literature Surveys with Scholarly Knowledge Graphs*, JCDL '20: Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020 (2020) 97, DOI: 10.1145/3383583.3398520 (cit. on pp. 37, 38, 62, 83, 92, 98).

[79] E. Mendes, C. Wohlin, K. Felizardo and M. Kalinowski, *When to update systematic literature reviews in software engineering*, Journal of Systems and Software **167** (2020) 110607, DOI: 10.1016/j.jss.2020.110607 (cit. on pp. 37, 38).

[80] C. A. Meyer, *Distinguishing published scholarly content with CrossMark*, Learned Publishing **24** (2011) 87, DOI: 10.1087/20110202 (cit. on p. 38).

[81] L. M. Schmidt and P. C. Gotzsche, *Of mites and men: reference bias in narrative review articles; a systematic review*, Journal of family practice **54** (2005) 334 (cit. on p. 38).

[82] Z. Nasar, S. W. Jaffry and M. K. Malik, *Information extraction from scientific articles: a survey*, vol. 117, 3, Springer International Publishing, 2018 1931, DOI: 10.1007/s11192-018-2921-5 (cit. on pp. 38, 39).

[83] M. Lipinski, K. Yao, C. Breitinger, J. Beel and B. Gipp, *Evaluation of header metadata extraction approaches and tools for scientific PDF documents*, Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (2013) 385, DOI: 10.1145/2467696.2467753 (cit. on pp. 38, 39, 41).

[84] D. Jung et al., *ChartSense: Interactive data extraction from chart images*, Conference on Human Factors in Computing Systems - Proceedings **2017-May** (2017) 6706, DOI: 10.1145/3025453.3025957 (cit. on p. 38).

[85] P. B. Heidorn, *Shedding light on the dark data in the long tail of science*, Library Trends **57** (2008) 280, DOI: 10.1353/lib.0.0036 (cit. on pp. 38, 63).

[86] D. Ahmetovic et al., *Axessibility: A LaTeX package for mathematical formulae accessibility in PDF documents*, ASSETS 2018 - Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility (2018) 352, DOI: 10.1145/3234695.3241029 (cit. on pp. 38, 39).

[87] A. Darvishy, *PDF accessibility: Tools and challenges*, vol. 10896 LNCS, Springer International Publishing, 2018 113, DOI: 10.1007/978-3-319-94277-3_20 (cit. on pp. 38, 39).

[88] J. T. Nganji,
The Portable Document Format (PDF) accessibility practice of four journal publishers,
Library & Information Science Research **37** (2015) 254,
DOI: https://doi.org/10.1016/j.lisr.2015.02.002 (cit. on p. 38).

[89] D. Shotton,
Semantic publishing: The coming revolution in Scientific journal publishing,
Learned Publishing **22** (2009) 85, DOI: 10.1087/2009202 (cit. on pp. 38–40).

[90] P. Ziemba, J. Jankowski and J. Wątróbski, "Online comparison system with certain and uncertain criteria based on multi-criteria decision analysis method",
International Conference on Computational Collective Intelligence, Springer, 2017 579,
DOI: 10.1007/978-3-319-67077-5_56 (cit. on p. 39).

[91] A. Petrova, E. Sherkhonov, B. Cuenca Grau and I. Horrocks,
"Entity Comparison in RDF Graphs", *International Semantic Web Conference*, 2017
526, DOI: 10.1007/978-3-319-68288-4_31 (cit. on p. 39).

[92] C. P. Medina and M. R. R. Ramon,
Using TF-IDF to Determine Word Relevance in Document Queries Juan,
New Educational Review **42** (2015) 40, DOI: 10.15804/tner.2015.42.4.03
(cit. on pp. 39, 66).

[93] P. Maillot et al., *Measuring structural similarity between RDF graphs*, Proceedings of the 33rd Annual ACM Symposium on Applied Computing (2019) 1960,
DOI: 10.1145/3167132.3167342 (cit. on p. 39).

[94] S. Araujo, J. Hidders, D. Schwabe and A. P. De Vries,
SERIMI - Resource description similarity, RDF instance matching and interlinking,
CEUR Workshop Proceedings **814** (2011) 246 (cit. on p. 39).

[95] P. Shvaiko and J. Euzenat, *Ontology matching: State of the art and future challenges*,
IEEE Transactions on Knowledge and Data Engineering **25** (2013) 158,
DOI: 10.1109/TKDE.2011.253 (cit. on p. 39).

[96] W. E. Winkler, *String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage*, Proceedings of the Section on Survey Research, American Statistical Association (1990) 354,
DOI: 10.1007/978-1-4612-2856-1_101 (cit. on p. 39).

[97] V. I. Levenshtein,
"Binary codes capable of correcting deletions, insertions, and reversals",
Soviet physics doklady, vol. 10, 8, 1966 707 (cit. on p. 39).

[98] D. Gromann and T. Declerck,
Comparing pretrained multilingual word embeddings on an ontology alignment task,
LREC 2018 - 11th International Conference on Language Resources and Evaluation
(2019) 230 (cit. on p. 39).

[99] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov,
Enriching Word Vectors with Subword Information,
Transactions of the Association for Computational Linguistics **5** (2017) 135,
DOI: 10.1162/tacl_a_00051 (cit. on pp. 39, 67).

[100] C. Kohl et al., *Online tools supporting the conduct and reporting of systematic reviews and systematic maps: A case study on CADIMA and review of existing tools*,
Environmental Evidence **7** (2018) 1, DOI: 10.1186/s13750-018-0115-5
(cit. on pp. 39, 40).

[101] D. R. Shanahan, *A living document: reincarnating the research article*,
Trials **16** (2015) 151, DOI: 10.1186/s13063-015-0666-5 (cit. on pp. 40, 98).

[102] V. Barbour, T. Bloom, J. Lin and E. Moylan,
Amending published articles: time to rethink retractions and corrections?,
F1000Research **6** (2017) 1960, DOI: 10.12688/f1000research.13060.1
(cit. on p. 40).

[103] F. Manola, E. Miller, B. McBride et al., *RDF primer*,
W3C recommendation **10** (2004) 6 (cit. on p. 40).

[104] E. Prudhommeaux and A. Seaborne, *SPARQL query language for RDF* (2008),
URL: http://www.w3.org/TR/rdf-sparql-query/ (cit. on pp. 40, 131).

[105] A. Garcia-Castro et al.,
Semantic Web and Social Web heading towards Living Documents in the Life Sciences,
Journal of Web Semantics **8** (2010) 155, DOI: 10.1016/j.websem.2010.03.006
(cit. on p. 40).

[106] M. Baker and D. Penny, *Is there a reproducibility crisis?*, Nature **533** (2016) 452,
DOI: 10.1038/533452A (cit. on p. 40).

[107] M. Aufreiter, A. Pawlik and N. Bentley,
Stencila–an office suite for reproducible research, eLife Labs **2** (2018) (cit. on p. 40).

[108] T. Kluyver et al.,
Jupyter Notebooks-a publishing format for reproducible computational workflows.
Vol. 2016, 2016 (cit. on p. 40).

[109] *Jupyter Book*, Executable Books Community (2020),
DOI: 10.5281/zenodo.4539666 (cit. on p. 40).

[110] M. Ouzzani, H. Hammady, Z. Fedorowicz and A. Elmagarmid,
Rayyan-a web and mobile app for systematic reviews, Systematic Reviews **5** (2016) 1,
DOI: 10.1186/s13643-016-0384-4 (cit. on p. 40).

[111] J. Babineau, *Product Review: Covidence (Systematic Review Software)*, Journal of the Canadian Health Libraries Association / Journal de l'Association des bibliothèques de la santé du Canada **35** (2014) 68, DOI: 10.5596/c14-016 (cit. on p. 40).

[112] H. Eriksson, *An Annotation Tool for Semantic Documents (System Description)*, 4th European Semantic Web Conference (ESWC) (2007) 759, DOI: 10.1007/978-3-540-72667-8_54 (cit. on pp. 40, 41).

[113] H. Shindo, Y. Munesada and Y. Matsumoto, *PDFanno: A web-based linguistic annotation tool for PDF documents*, LREC 2018 - 11th International Conference on Language Resources and Evaluation (2019) 1082 (cit. on pp. 40, 41, 120).

[114] J. Takis, A. Q. Islam, C. Lange and S. Auer, *Crowdsourced semantic annotation of scientific publications and tabular data in PDF*, ACM International Conference Proceeding Series **16-17-Sept** (2015) 1, DOI: 10.1145/2814864.2814887 (cit. on pp. 40–42, 120, 122).

[115] M. Koniaris, G. Papastefanatos and I. Anagnostopoulos, *Solon: A holistic approach for modelling, managing and mining legal sources*, Algorithms **11** (2018) 1, DOI: 10.3390/a11120196 (cit. on p. 40).

[116] A. Bergeaud, Y. Potiron and J. Raimbault, *Classifying patents based on their semantic content*, PLoS ONE **12** (2017) 1, DOI: 10.1371/journal.pone.0176310 (cit. on p. 40).

[117] M. Traquair, E. Kara, B. Kantarci and S. Khan, *Deep Learning for the Detection of Tabular Information from Electronic Component Datasheets*, Proceedings - International Symposium on Computers and Communications **2019-June** (2019) 1, DOI: 10.1109/ISCC47284.2019.8969682 (cit. on p. 40).

[118] B. Custers and D. Bachlechner, *Advancing the EU Data Economy: Conditions for Realizing the Full of Potential of Data Reuse*, SSRN Electronic Journal (2018) 1, DOI: 10.2139/ssrn.3091038 (cit. on p. 40).

[119] T. Hassan and R. Baumgartner, *Table recognition and understanding from PDF files*, Proceedings of the International Conference on Document Analysis and Recognition, ICDAR (2007) 1143, DOI: 10.1109/ICDAR.2007.4377094 (cit. on p. 41).

[120] G. Ros, *Analysis of Tabula : A PDF-Table extraction tool*, 2019 (cit. on p. 41).

[121] M. Körner, B. Ghavimi, P. Mayr, H. Hartmann and S. Staab, "Evaluating Reference String Extraction Using Line-Based Conditional Random Fields: A Case Study with German Language Publications", *New Trends in Databases and Information Systems*, Cham, 2017 137, DOI: 10.1007/978-3-319-67162-8_15 (cit. on p. 41).

[122] P. Lopez, *GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications*,
International conference on theory and practice of digital libraries **5714** (2009) 473,
DOI: 10.1007/978-3-642-04346-8_62 (cit. on pp. 41, 111).

[123] A. Oelen, M. Stocker and S. Auer,
"Creating a Scholarly Knowledge Graph from Survey Article Tables",
Digital Libraries at Times of Massive Societal Transition,
ed. by E. Ishita, N. L. S. Pang and L. Zhou,
Cham: Springer International Publishing, 2020 373 (cit. on pp. 42, 82, 97).

[124] O. Lehmberg, D. Ritze, R. Meusel and C. Bizer,
A Large Public Corpus of Web Tables containing Time and Context Metadata,
Proceedings of the 25th International Conference Companion on World Wide Web - WWW '16 Companion (2016), DOI: 10.1145/2872518.2889386 (cit. on p. 42).

[125] Y. Liu, K. Bai, P. Mitra and C. L. Giles,
Tableseer: automatic table metadata extraction and searching in digital libraries,
Proceedings of the 2007 conference on Digital libraries - JCDL '07 (2007),
DOI: 10.1145/1255175.1255193 (cit. on p. 42).

[126] R. Rastan, H.-Y. Paik and J. Shepherd, *TEXUS*, Proceedings of the 2015 ACM Symposium on Document Engineering - DocEng '15 (2015),
DOI: 10.1145/2682571.2797069 (cit. on p. 42).

[127] M. D. Adelfio and H. Samet, *Schema extraction for tabular data on the web*, en,
Proceedings of the VLDB Endowment **6** (2013) 421,
DOI: 10.14778/2536336.2536343 (cit. on p. 42).

[128] P. Stenetorp, S. Pyysalo and G. Topi,
BRAT : a Web-based Tool for NLP-Assisted Text Annotation (2012) 102 (cit. on p. 41).

[129] K. Bontcheva, H. Cunningham, I. Roberts and V. Tablan, *Web-based collaborative corpus annotation: Requirements and a framework implementation*,
New Challenges for NLP Frameworks (2010) 20 (cit. on p. 41).

[130] R. Snow, B. O'connor, D. Jurafsky and A. Y. Ng, "Cheap and fast–but is it good? evaluating non-expert annotations for natural language tasks", *Proceedings of the 2008 conference on empirical methods in natural language processing*, 2008 254 (cit. on p. 43).

[131] G. G. Chowdhury, *Natural language processing*,
Annual Review of Information Science and Technology **37** (2003) 51,
DOI: https://doi.org/10.1002/aris.1440370103 (cit. on p. 43).

[132] J. Li, A. Sun, J. Han and C. Li,
A Survey on Deep Learning for Named Entity Recognition,
IEEE Transactions on Knowledge and Data Engineering **34** (2022) 50,
DOI: 10.1109/TKDE.2020.2981314 (cit. on p. 43).

[133] W. Shen, J. Wang and J. Han,
Entity linking with a knowledge base: Issues, techniques, and solutions,
IEEE Transactions on Knowledge and Data Engineering **27** (2015) 443,
DOI: 10.1109/TKDE.2014.2327028 (cit. on pp. 43, 127).

[134] R. Alghamdi and K. Alfalqi, *A Survey of Topic Modeling in Text Mining*,
International Journal of Advanced Computer Science and Applications **6** (2015),
DOI: 10.14569/IJACSA.2015.060121 (cit. on p. 43).

[135] O. Tas and F. Kiyani, *A survey automatic text summarization*,
PressAcademia Procedia **5** (2007) 205, DOI: 10.17261/Pressacademia.2017.591
(cit. on p. 43).

[136] A. Oelen, M. Yaser Yaradeh, M. Stocker and S. Auer,
"Chapter 10. Organizing Scholarly Knowledge leveraging Crowdsourcing, Expert Curation and Automated Techniques", en,
Linking Knowledge: Linked Open Data for Knowledge Organization and Visualization,
ed. by R. P. Smiraglia and A. Scharnhorst, 1st ed., Baden-Baden: Ergon-Verlag, 2021
181, DOI: 10.5771/9783956506611-181, (visited on 19/10/2021) (cit. on p. 46).

[137] S. Auer et al., *Improving Access to Scientific Literature with Knowledge Graphs*,
Bibliothek Forschung und Praxis **44** (2020) 516 (cit. on p. 46).

[138] M. Y. Jaradeh, A. Oelen, M. Prinz, M. Stocker and S. Auer,
"Open research knowledge graph: A system walkthrough",
International Conference on Theory and Practice of Digital Libraries, Springer, 2019
348, DOI: 10.1007/978-3-030-30760-8_31 (cit. on p. 46).

[139] J. P. Ostriker, C. V. Kuh and J. A. Voytuk,
A data-based assessment of research-doctorate programs in the United States,
National Academies Press, 2015 (cit. on p. 50).

[140] J. Corman, J. L. Reutter and O. Savković,
"Semantics and validation of recursive SHACL",
International Semantic Web Conference, Springer, 2018 318,
DOI: 10.1007/978-3-030-00671-6_19 (cit. on p. 50).

[141] R. Lammey, *CrossRef text and data mining services*,
Insights: the UKSG Journal **28** (2015) 62, DOI: 10.1629/uksg.233
(cit. on pp. 53, 81).

[142] L. L. Haak, M. Fenner, L. Paglione, E. Pentz and H. Ratner, *ORCID: a system to uniquely identify researchers*, Learned Publishing **25** (2012) 259, DOI: 10.1087/20120404 (cit. on p. 53).

[143] J. D'Souza et al., "The STEM-ECR Dataset: Grounding Scientific Entity References in STEM Scholarly Content to Authoritative Encyclopedic and Lexicographic Sources", *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020 2192 (cit. on pp. 54, 134).

[144] M. Schweiker et al., *Dynamic review tables for topical reviews on occupants' perception and behaviour in buildings*, 2021, URL: osf.io/gnvp2 (cit. on p. 57).

[145] S. Seuring, M. Müller, M. Westhaus and R. Morana, "Conducting a Literature Review — The Example of Sustainability in Supply Chains", *Research Methodologies in Supply Chain Management*, Heidelberg: Physica-Verlag HD, 2005 91, DOI: 10.1007/3-7908-1636-1_7 (cit. on p. 61).

[146] T. Kuhn et al., *Decentralized provenance-aware publishing with nanopublications* (2016) 1, DOI: 10.7717/peerj-cs.78 (cit. on pp. 61, 95, 129).

[147] E. M. Lasda Bergman, *Finding Citations to Social Work Literature: The Relative Benefits of Using Web of Science, Scopus, or Google Scholar*, Journal of Academic Librarianship **38** (2012) 370, DOI: 10.1016/j.acalib.2012.08.002 (cit. on p. 61).

[148] J. J. Randolph, *A guide to writing the dissertation literature review*, Practical Assessment, Research and Evaluation **14** (2009), DOI: 10.7275/b0az-8t74 (cit. on pp. 62, 95).

[149] R. J. Torraco, *Writing Integrative Literature Reviews: Guidelines and Examples*, Human Resource Development Review **4** (2005) 356, DOI: 10.1177/1534484305278283 (cit. on p. 62).

[150] A. Oelen, M. Y. Jaradeh, K. E. Farfar, M. Stocker and S. Auer, "Comparing Research Contributions in a Scholarly Knowledge Graph", *Proceedings of the Third International Workshop on Capturing Scientific Knowledge (SciKnow19)*, 2019 21 (cit. on pp. 62, 64, 82).

[151] N. Bikakis and T. Sellis, *Exploration and visualization in the web of big linked data: A survey of the state of the art*, CEUR Workshop Proceedings **1558** (2016) (cit. on p. 71).

[152] D. Diefenbach, V. Lopez, K. Singh and P. Maret, *Core techniques of question answering systems over knowledge bases: a survey*, Knowledge and Information Systems **55** (2018) 529, DOI: 10.1007/s10115-017-1100-y (cit. on p. 71).

[153] I. Hussain and S. Asghar,
A survey of author name disambiguation techniques: 2010–2016,
The Knowledge Engineering Review **32** (2017) 1,
DOI: 10.1017/s0269888917000182 (cit. on p. 71).

[154] R. Naidu, S. K. Bharti, K. S. Babu and R. K. Mohapatra,
"Text Summarization with Automatic Keyword Extraction in Telugu e-Newspapers",
Smart Innovation, Systems and Technologies, vol. 77, 2018 555,
DOI: 10.1007/978-981-10-5544-7_54 (cit. on p. 71).

[155] D. Herrmannova and P. Knoth, *An analysis of the microsoft academic graph*,
D-lib Magazine **22** (2016), DOI: 10.1045/september2016-herrmannova
(cit. on p. 81).

[156] M. Vasileiadis, N. Kaklanis, K. Votis and D. Tzovaras,
Extraction of tabular data from document images,
Proceedings of the 14th Web for All Conference, W4A (2017),
DOI: 10.1145/3058555.3058581 (cit. on p. 85).

[157] R. Verborgh and M. De Wilde, *Using OpenRefine*, Packt Publishing Ltd, 2013
(cit. on p. 86).

[158] A. Oelen, M. Stocker and S. Auer,
Dataset for Creating a Scholarly Knowledge Graph from Survey Articles (2020),
DOI: 10.5281/ZENODO.3735152,
URL: https://doi.org/10.5281/zenodo.3735152#.XoOTVlS2qRP.mendeley
(cit. on pp. 89, 92).

[159] A. Oelen, M. Stocker and S. Auer,
"SmartReviews: Towards Human- and Machine-Actionable Reviews",
Linking Theory and Practice of Digital Libraries,
ed. by G. Berget, M. M. Hall, D. Brenn and S. Kumpulainen,
Cham: Springer International Publishing, 2021 181 (cit. on pp. 96, 98).

[160] A. Oelen, M. Stocker and S. Auer, "SmartReviews: Towards Human- and
Machine-Actionable Representation of Review Articles",
Towards Open and Trustworthy Digital Societies,
ed. by H.-R. Ke, C. S. Lee and K. Sugiyama,
Cham: Springer International Publishing, 2021 105 (cit. on p. 96).

[161] TheSophists, *Requirements Engineering - A short RE Primer*, 2016, URL:
https://www.sophist.de/fileadmin/user_upload/Bilder_zu_Seiten/
Publikationen/Wissen_for_free/RE-Broschuere_Englisch_-_Online.pdf
(cit. on p. 98).

[162] B. Caldwell, M. Cooper, L. G. Reid and G. Vanderheiden,
Web Content Accessibility Guidelines (WCAG) 2.0,
W3C Recommendation 11 December 2008 (2008) 33,
URL: http://www.w3.org/TR/WCAG20/ (cit. on p. 99).

[163] Y. Xie, J. J. Allaire and G. Grolemund, *R markdown: The definitive guide*,
CRC Press, 2018, DOI: 10.1201/9781138359444 (cit. on p. 100).

[164] A. Kittur and R. E. Kraut,
Harnessing the Wisdom of Crowds in Wikipedia: Quality Through Coordination
(2008), DOI: 10.1145/1460563.1460572 (cit. on p. 102).

[165] M. Bergquist and J. Ljungberg, *Relationships in open source communities*,
Information Systems Journal (2001) 305 (cit. on p. 103).

[166] J. Willinsky, *The Access Principle: the case for open access to research and scholarship*,
Lhs **2** (2006) 165 (cit. on p. 108).

[167] B. Green and Y. Chen,
The principles and limits of algorithm-in-the-loop decision making,
Proceedings of the ACM on Human-Computer Interaction **3** (2019),
DOI: 10.1145/3359152 (cit. on p. 111).

[168] A. Oelen, M. Stocker and S. Auer,
"Crowdsourcing Scholarly Discourse Annotations",
26th International Conference on Intelligent User Interfaces, IUI '21,
College Station, TX, USA: Association for Computing Machinery, 2021 464,
DOI: 10.1145/3397481.3450685 (cit. on p. 113).

[169] P. Ginsparg, *ArXiv at 20*, Nature **476** (2011) 145, DOI: 10.1038/476145a
(cit. on p. 113).

[170] K. Aberer and A. Boyarsky,
ScienceWISE: a Web-based Interactive Semantic Platform for scientific collaboration,
10th International Semantic Web Conference (ISWC 2011-Demo) (2011),
DOI: 10.1007/978-3-662-46641-4_33 (cit. on p. 113).

[171] C. Lange and A. D. Iorio,
Semantic Publishing Challenge – Assessing the Quality of Scientific Output,
Semantic Web Evaluation Challenge **1** (2014) 61,
DOI: 10.1007/978-3-319-12024-9 (cit. on p. 113).

[172] M. Kolchin, E. Cherny, F. Kozlov, A. Shipilo and L. Kovriguina,
CEUR-WS-LOD: Conversion of CEUR-WS Workshops to Linked Data,
Semantic Web Evaluation Challenges **1** (2015) 51,
DOI: 10.1007/978-3-319-25518-7 (cit. on p. 113).

[173] D. Nadeau and S. Sekine, *A Survey on Named Entity Recognition*, Lingvisticae Investigationes **30** (2007) 3, DOI: 10.1007/978-981-13-9409-6_218 (cit. on pp. 114, 127).

[174] A. H. W. G. for Critical Appraisal of the Medical Literature, *A proposal for more informative abstracts of clinical articles*, Annals of Internal Medicine **106** (1987) 598 (cit. on p. 114).

[175] T. Nakayama, N. Hirai, S. Yamazaki and M. Naito, *Adoption of structured abstracts by general medical journals and format for a structured abstract*, Journal of the Medical Library Association **93** (2005) 237 (cit. on p. 114).

[176] H. de Ribaupierre and G. Falquet, *Extracting discourse elements and annotating scientific documents using the SciAnnotDoc model: a use case in gender documents*, International Journal on Digital Libraries **19** (2018) 271, DOI: 10.1007/s00799-017-0227-5 (cit. on pp. 114, 119).

[177] R. L. Fowler and A. S. Barker, *Effectiveness of highlighting for retention of text material.*, Journal of Applied Psychology **59** (1974) 358, DOI: 10.1037/h0036750 (cit. on p. 114).

[178] R. F. Lorch, *Text-signaling devices and their effects on reading and memory processes*, Educational Psychology Review **1** (1989) 209, DOI: 10.1007/BF01320135 (cit. on p. 114).

[179] A. Taylor, M. Marcus and B. Santorini, *The Penn Treebank: An Overview* (2003) 5, DOI: 10.1007/978-94-010-0201-1_1 (cit. on p. 114).

[180] A. Kittur, B. Smus, S. Khamkar and R. E. Kraut, "Crowdforge: Crowdsourcing complex work", *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011 43, DOI: 10.1145/2047196.2047202 (cit. on p. 116).

[181] J. Brooke et al., *SUS-A quick and dirty usability scale*, Usability evaluation in industry **189** (1996) 4 (cit. on pp. 116, 121, 139, 141).

[182] M. Allahbakhsh et al., *Quality control in crowdsourcing systems: Issues and directions*, IEEE Internet Computing **17** (2013) 76, DOI: 10.1109/MIC.2013.20 (cit. on p. 116).

[183] J. Devlin, M. W. Chang, K. Lee and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference **1** (2019) 4171 (cit. on p. 118).

[184] D. Miller, *Leveraging BERT for Extractive Text Summarization on Lectures* (2019) (cit. on p. 118).

[185] I. Mani, *Automatic summarization*, vol. 3, John Benjamins Publishing, 2001, DOI: 10.1075/nlp.3 (cit. on p. 118).

[186] S. Spala, F. Dernoncourt, W. Chang and C. Dockhorn, *A Web-based Framework for Collecting and Assessing Highlighted Sentences in a Document*, Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations (2018) 78 (cit. on p. 118).

[187] J. Davison, *Zero-Shot Learning in Modern NLP*, (accessed on 2020-09-30) (2020), URL: https://joeddav.github.io/blog/2020/05/29/ZSL.html (cit. on p. 118).

[188] W. Yin, J. Hay and D. Roth, *Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach*, EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference (2020) 3914, DOI: 10.18653/v1/d19-1404 (cit. on p. 118).

[189] D. MacAuley, *Critical appraisal of medical literature: An aid to rational decision making*, Family Practice **12** (1995) 98, DOI: 10.1093/fampra/12.1.98 (cit. on p. 119).

[190] R. Lammey, *CrossRef developments and initiatives: An update on services for the scholarly publishing community from CrossRef*, Science Editing **1** (2014) 13, DOI: 10.6087/kcse.2014.1.13 (cit. on p. 120).

[191] S. G. Hart, *NASA-task load index (NASA-TLX); 20 years later*, Proceedings of the Human Factors and Ergonomics Society (2006) 904, DOI: 10.1177/154193120605000909 (cit. on pp. 121, 139).

[192] T. Weber et al., *Draw with Me: Human-in-the-Loop for Image Restoration*, **20** (2020) 243, DOI: 10.1145/3377325.3377509 (cit. on p. 122).

[193] R. Grier, "How high is high? A metanalysis of NASA TLX global workload scores", vol. 59, 2015, DOI: 10.1177/1541931215591373 (cit. on pp. 123, 139).

[194] T. S. Tullis and J. N. Stetson, *A Comparison of Questionnaires for Assessing Website Usability*, Usability Professional Association Conference (2004) 1 (cit. on pp. 128, 143).

[195] K. Bontcheva et al., *GATE Teamware: a web-based, collaborative text annotation framework*, Language Resources and Evaluation **47** (2013) 1007, DOI: 10.1007/s10579-013-9215-6 (cit. on p. 128).

[196] G. T. Gobbel et al., *Assisted annotation of medical free text using RapTAT*, Journal of the American Medical Informatics Association **21** (2014) 833, DOI: 10.1136/amiajnl-2013-002255 (cit. on p. 128).

[197] M. Stocker, P. Paasonen, M. Fiebig, M. A. Zaidan and A. Hardisty, *Curating Scientific Information in Knowledge Infrastructures*, Data Science Journal **17** (2018), DOI: 10.5334/dsj-2018-021 (cit. on p. 129).

[198] A. Oelen, M. Stocker and S. Auer, *TinyGenius: Intertwining Natural Language Processing with Microtask Crowdsourcing for Scholarly Knowledge Graph Creation.*, JCDL '22: Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2022 (2022), DOI: 10.1145/3529372.3533285 (cit. on p. 130).

[199] D. Hernández, A. Hogan and M. Krötzsch, *Reifying RDF: What works well with wikidata?*, SSWS@ ISWC **1457** (2015) 32 (cit. on p. 131).

[200] V. Nguyen, O. Bodenreider and A. Sheth, "Don't like RDF Reification? Making Statements about Statements Using Singleton Property", *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, Seoul, Korea: Association for Computing Machinery, 2014 759, DOI: 10.1145/2566486.2567973 (cit. on p. 131).

[201] J. J. Carroll, C. Bizer, P. Hayes and P. Stickler, "Named Graphs, Provenance and Trust", *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, Chiba, Japan: Association for Computing Machinery, 2005 613, DOI: 10.1145/1060745.1060835 (cit. on p. 131).

[202] O. Hartig, "Foundations of RDF* and SPARQL* : (An Alternative Approach to Statement-Level Metadata in RDF)", *Proceedings of the 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web 2017 :* vol. 1912, CEUR Workshop Proceedings 1912, Juan Reutter, Divesh Srivastava, 2017, URL: http://ceur-ws.org/Vol-1912/paper12.pdf (cit. on p. 131).

[203] A. A. Salatino, F. Osborne, T. Thanapalasingam and E. Motta, "The CSO Classifier: Ontology-Driven Detection of Research Topics in Scholarly Articles", *Digital Libraries for Open Knowledge*, ed. by A. Doucet, A. Isaac, K. Golub, T. Aalberg and A. Jatowt, Cham: Springer International Publishing, 2019 296, DOI: 10.1007/978-3-030-30760-8_26 (cit. on p. 133).

[204] G. Weikum, J. Hoffart and F. M. Suchanek, *Ten Years of Knowledge Harvesting: Lessons and Challenges*, IEEE Data Eng. Bull. **39** (2016) 41 (cit. on p. 134).

[205] A. F. Hayes and K. Krippendorff, *Answering the Call for a Standard Reliability Measure for Coding Data*, Communication Methods and Measures **1** (2007) 77, DOI: 10.1080/19312450709336664 (cit. on p. 141).

[206] A. Zapf, S. Castell, L. Morawietz and A. Karch, *Measuring inter-rater reliability for nominal data–which coefficients and confidence intervals are appropriate?*, BMC medical research methodology **16** (2016) 1 (cit. on p. 141).

[207] F. F.-H. Nah, *A study on tolerable waiting time: how long are Web users willing to wait?*, Behaviour & Information Technology **23** (2004) 153, DOI: 10.1080/01449290410001669914 (cit. on p. 142).

[208] N. Hahn, S. T. Iqbal and J. Teevan, "Casual Microtasking: Embedding Microtasks in Facebook", *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, New York, NY, USA: Association for Computing Machinery, 2019 1, DOI: 10.1145/3290605.3300249 (cit. on p. 144).

[209] R. G. Côté, P. Jones, R. Apweiler and H. Hermjakob, *The Ontology Lookup Service, a lightweight cross-platform tool for controlled vocabulary queries*, BMC bioinformatics **7** (2006) 1, DOI: 10.1186/1471-2105-7-97 (cit. on p. 149).

[210] A. Anderson, D. Huttenlocher, J. Kleinberg and J. Leskovec, "Steering User Behavior with Badges", *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, Rio de Janeiro, Brazil: Association for Computing Machinery, 2013 95, DOI: 10.1145/2488388.2488398 (cit. on p. 150).

[211] T. Steiner, "Bots vs. Wikipedians, Anons vs. Logged-Ins (Redux): A Global Study of Edit Activity on Wikipedia and Wikidata", *Proceedings of The International Symposium on Open Collaboration*, OpenSym '14, Berlin, Germany: Association for Computing Machinery, 2014 1, DOI: 10.1145/2641580.2641613 (cit. on p. 150).

List of Figures

List of Tables