Thinking Outside the Graph: Scholarly Knowledge Graph Construction Leveraging Natural Language Processing

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
Mohamad Yaser Jaradeh, M.Sc.
geboren am 15.08.1992
in Damaskus, Syrien

2022

1. Gutachter: Prof. Dr. Sören Auer
2. Gutachter: Prof. Dr. Manolis Koubarakis
Tag der Promotion: 09.11.2022

Zusammenfassung

Trotz des verbesserten digitalen Zugangs zu wissenschaftlichem Wissen in den letzten Jahrzehnten ist die wissenschaftliche Kommunikation nach wie vor ausschließlich Dokumenten-basiert. Die dokumentenorientierten Arbeitsabläufe in der wissenschaftlichen Publikation haben die Grenzen der Angemessenheit erreicht, wie die jüngsten Diskussionen über die zunehmende Verbreitung wissenschaftlicher Literatur, die Unzulänglichkeiten bei Peer-Reviews und die Krise der Reproduzierbarkeit zeigen. In dieser Form bleibt wissenschaftliches Wissen in Darstellungen gefangen, die für die maschinelle Verarbeitung ungeeignet sind. Solange die wissenschaftliche Kommunikation in dieser Form verbleibt, können wir nicht von den Fortschritten profitieren, die im Bereich des maschinellen Lernens und der Verarbeitung natürlicher Sprache gemacht werden. Solche Techniken würden die Umwandlung von rein textbasierten in (halb-)strukturierte semantische Beschreibungen erleichtern, die in einer Sammlung von großen föderierten Graphen miteinander verknüpft sind. Wir brauchen dringend eine neue semantische Infrastruktur, die in der Lage ist, wissenschaftliches Wissen zu speichern, zu bearbeiten und abzufragen. Ebenso wichtig ist eine Reihe von maschinellen Hilfsmitteln, die den entstehenden wissenschaftlichen Wissensgraphen auffüllen, kuratieren und erforschen können.

In dieser Arbeit befassen wir uns mit dem Problem der Konstruktion eines wissenschaftlichen Wissensgraphen unter Verwendung von Techniken der natürlichen Sprachverarbeitung. Zunächst befassen wir uns mit Problem der Entwicklung eines wissenschaftlichen Wissensgraphen für die strukturierte wissenschaftliche Kommunikation, der automatisch ausgefüllt und erstellt werden kann. Wir entwerfen und implementieren den Open Research Knowledge Graph (ORKG), eine Infrastruktur, die wissenschaftliche Kommunikation modellieren, speichern und automatisch kuratieren kann. Anschließend schlagen wir eine Methode zur automatischen Extraktion von Informationen in Wissensgraphen vor. Mit PLUMBER schaffen wir ein Rahmenwerk für die dynamische Zusammenstellung offener Informationsextraktionspipelines auf der Grundlage des Eingabetextes. Solche Pipelines werden aus von der Community erstellten Informationsextraktionskomponenten zusammengesetzt, um individuelle Forschungsbeiträge gemeinsam zu konsolidieren. Darüber hinaus stellen wir MORTY als einen gezielteren Ansatz vor, der die automatische Textzusammenfassung nutzt, um aus dem Text eines wissenschaftlichen Artikels strukturierte Zusammenfassungen zu erstellen, die alle erforderlichen Informationen enthalten. Im Gegensatz zum Pipeline-Ansatz extrahiert MORTY nur die Informationen, die ihm aufgetragen wurden, was es zu einem wertvolleren Werkzeug für verschiedene Kuratierungs- und Beitragsanwendungsfälle macht. Außerdem untersuchen wir das Problem der Vervollständigung von Wissensgraphen. exBERT ist in der Lage, Aufgaben zur Vervollständigung von Wissensgraphen, wie z.B. die Vorhersage von Relationen und Entitäten, auf wissenschaftlichen Wissensgraphen mittels textueller Tripel-Klassifikation durchzuführen. Schließlich verwenden wir die strukturierten Beschreibungen, die sowohl aus manuellen als auch aus automatisierten Quellen gesammelt wurden, mit einem Ansatz zur Beantwortung von Fragen, der auf den maschinenverarbeitbaren Beschreibungen im ORKG aufbaut. Wir schlagen JarvisQA vor, eine Schnittstelle zur Beantwortung von Fragen, die auf tabellarischen

Ansichten von wissenschaftlichen Wissensgraphen, d.h. ORKG-Vergleichen, basiert. JarvisQA ist in der Lage, eine Vielzahl von Fragen in natürlicher Sprache zu beantworten und komplexe Antworten auf vorselektierten Teilgraphen abzurufen.

Diese Beiträge sind von zentraler Bedeutung für die Untersuchung der Durchführbarkeit von Methoden zur Verarbeitung natürlicher Sprache in wissenschaftlichen Wissensgraphen und bilden die Grundlage dafür, welche Methoden in welchen Fällen eingesetzt werden können. Unsere Arbeit zeigt auf, welche Herausforderungen und Probleme bei der automatischen Konstruktion wissenschaftlicher Wissensgraphen bestehen, und eröffnet zukünftige Forschungsrichtungen.
Keywords: Semantic Web, Natural Language Processing, Machine Learning, Knowledge Graph Construction, Question Answering.

Abstract

Despite improved digital access to scholarly knowledge in recent decades, scholarly communication remains exclusively document-based. The document-oriented workflows in science publication have reached the limits of adequacy as highlighted by recent discussions on the increasing proliferation of scientific literature, the deficiency of peer-review and the reproducibility crisis. In this form, scientific knowledge remains locked in representations that are inadequate for machine processing. As long as scholarly communication remains in this form, we cannot take advantage of all the advancements taking place in machine learning and natural language processing techniques. Such techniques would facilitate the transformation from pure text based into (semi-)structured semantic descriptions that are interlinked in a collection of big federated graphs. We are in dire need for a new age of semantically enabled infrastructure adept at storing, manipulating, and querying scholarly knowledge. Equally important is a suite of machine assistance tools designed to populate, curate, and explore the resulting scholarly knowledge graph.

In this thesis, we address the issue of constructing a scholarly knowledge graph using natural language processing techniques. First, we tackle the issue of developing a scholarly knowledge graph for structured scholarly communication, that can be populated and constructed automatically. We co-design and co-implement the Open Research Knowledge Graph (ORKG), an infrastructure capable of modeling, storing, and automatically curating scholarly communications. Then, we propose a method to automatically extract information into knowledge graphs. With Plumber, we create a framework to dynamically compose open information extraction pipelines based on the input text. Such pipelines are composed from community-created information extraction components in an effort to consolidate individual research contributions under one umbrella. We further present MORTY as a more targeted approach that leverages automatic text summarization to create from the scholarly article's text structured summaries containing all required information. In contrast to the pipeline approach, MORTY only extracts the information it is instructed to, making it a more valuable tool for various curation and contribution use cases. Moreover, we study the problem of knowledge graph completion. exBERT is able to perform knowledge graph completion tasks such as relation and entity prediction tasks on scholarly knowledge graphs by means of textual triple classification. Lastly, we use the structured descriptions collected from manual and automated sources alike with a question answering approach that builds on the machine-actionable descriptions in the ORKG. We propose JarvisQA, a question answering interface operating on tabular views of scholarly knowledge graphs i.e., ORKG comparisons. JarvisQA is able to answer a variety of natural language questions, and retrieve complex answers on pre-selected sub-graphs.

These contributions are key in the broader agenda of studying the feasibility of natural language processing methods on scholarly knowledge graphs, and lays the foundation of which methods can be used on which cases. Our work indicates what are the challenges and issues with automatically constructing scholarly knowledge graphs, and opens up future research directions.

Acknowledgements

Throughout this exciting Ph.D. journey, I met several people who inspired and supported me all those years. First and foremost, I would like to thank Prof. Dr. Sören Auer for his constant support and patience, and for giving me the chance to pursue the degree under his supervision. His support and advice fostered this research direction. Furthermore, I would like to thank Dr. Markus Stocker for his nurturing support and for allowing me to bend his ear whenever I wanted to discuss something work-related or otherwise. Moreover, I would like to extend my thanks to Dr. Kuldeep Singh for his constant work with me on papers and for challenging me to do better in research. My journey with Kuldeep started when we meet in Bonn while I was doing my masters, and it sparks still today. Last but not least, a special thanks to Prof. Dr. Maria-Esther Vidal for her wise words and for pushing me to follow my degree at the Leibniz University Hannover. She is always the voice of reason that I can count on.

To my "partner in crime" Allard Oelen, whom I bounced off ideas (research or otherwise) without fail. Allard and I started off our Ph.D. journeys around the same time. So we shared the ups and downs, traveling or setting in the office. I have in him a true friend and an awesome colleague.

To my chess partner Nour Jaafari and all the fun time we shared together - you rock. I'm also grateful for knowing Ahmad (my old friend), Sam (my jogging partner), and Vitalis with whom I shared cherished moments throughout this journey. As well as Khier Eddine, Omar, and Golsa, who made life in the office during the pandemic much nicer and akin to normalcy. Not to mention Salomon, Manuel, Muhammad, Oliver, and all other ORKG team members for their direct and indirect contributions to this adventure. Last but not least, a special thanks to Simone Matern for all the bureaucratic and organizational support that she lent me without fail.

My heartfelt appreciation to my family, whom without their support I would not have been able to achieve this milestone. My family that supported me to travel to another country in order to peruse my Ph.D. sacrificing many comforts in order for me to achieve this. To my mom Mrs. Shahnaz Baghdadi for her constant encouragement and pure love. To my dad Mr. Wajih Jaradeh for being my cornerstone that I can always rely on. To my brother Amer Jaradeh in his role as my second mate that is always there for me. To my uncle Mr. Fathi Baghdadi for his support and unshaken belief in me throughout this journey. Finally to my aunt Mrs. Marina Sidorschinko as my second mother and for all her caring sentiments. Thank you all again for your unconditional support and you just being awesome people.

I dedicate this thesis to my gurus: Prof. Dr. Sören Auer, Dr. Markus Stocker, and Dr. Kuldeep Singh.

Contents

CHAPTER 1

Introduction

Machine-assistance is becoming the norm in many aspects of our daily lives, as data and formally represented knowledge have become key assets. It is usually expected to have automated support in finance, education, entertainment, healthcare, and business. The more the better, as long as it is helpful and "working". This means that machine support has to be accurate and performant. Artificial intelligence and its various subfields and applications such as Natural Language Processing play an integral role in all kinds of information systems ranging from cellphones to weather forecasting. However, in science and scholarly communication, specifically, the advancements are lagging behind, The ever-increasing number of document-based publications are hindering efficient scholarly knowledge use. Hence we find ourselves in need of a digital scholarly infrastructure that can represent knowledge formally (i.e., in structured and semantic manner) and is capable of running various user-tailored applications on top. Furthermore, we require the technology base for automated knowledge extraction and integration from existing legacy artifacts into the envisioned infrastructure. The overall goal is to bring scholarly communication into the digital age, through a fundamentally transformative digitalization [BK16] not mere digitization as seen in recent decades as we moved from print to digital documents. Various initiatives such as NFDI[1] and EOSC[2] are starting to address this goal and more resources are being invested[3] to materialize it [Reso7]. Simultaneously, we notice increasing research activity in this area.

As a matter of course, scientific knowledge is produced with the scientific method. While the details vary as the method is performed in research lifecycles, broad common phases can be identified. First, hypotheses and research objectives are formed. Second, experiments are conducted about the conceived hypotheses. Then, the results are peer reviewed, and finally the resulting work is published to the community for further research [BM19].

A semantic scholarly infrastructure operating as a hub for scientific knowledge requires various components to enable publishing, storing, searching, manipulating, annotating, and ingesting its data. Hence, providing a knowledge graph infrastructure to interlink concepts and entities, which in turn enables the exploration and curation of the underlying knowledge. Such infrastructure also needs to provide means of automatically ingesting existing knowledge from legacy articles into the knowledge graph as a semantically machine-actionable expression.

[1] https://www.nfdi.de/?lang=en
[2] https://eosc.eu/
[3] https://stats.oecd.org/Index.aspx?DataSetCode=ONRD_FUNDS

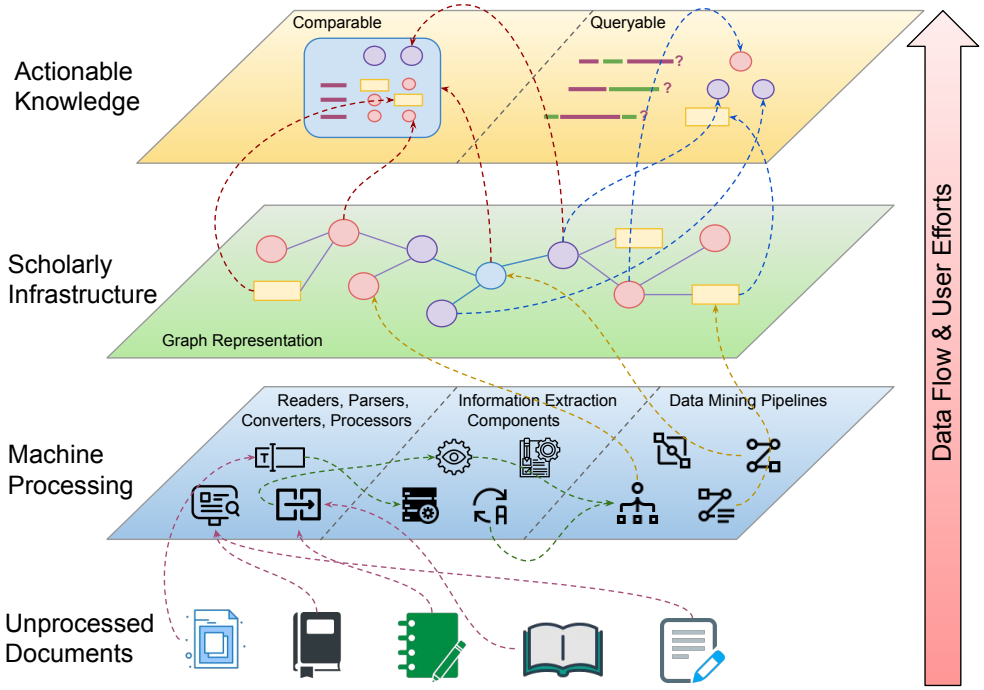


Figure 1.1: **From Legacy Documents to Actionable Knowledge**. At the bottom, the existing legacy documents in digitized but machine unreadable formats. Followed by the lowest layer, which contains all the machine support needed for the extraction of relevant knowledge and its transformation into semantic representation. The middle layer consists of the semantic scholarly infrastructure responsible for the actionable scholarly data representation, manipulation, and management. The top layer includes the user-tailored applications running over the infrastructure structured data. Data flow is bottom to top while user efforts are heaviest at the bottom and lightest at the top.

In this thesis, we aim to lay the foundations for creating and employing natural language processing tools on scholarly knowledge interconnected in a knowledge graph, and in textual publications. This work is motivated by advancement of the semantic web techniques, as well as the rapid evolution of automated natural language processing approaches, and their ever-growing role in our everyday life. We highlight the importance of utilizing automated approaches to extract information from legacy documents and transforming them into a machine-actionable representation. Furthermore, we propose the basis of a semantically enabled scholarly infrastructure, which is empowered by machine-supported techniques and automated curation methods.

1.1 Motivation

Research articles are digital documents accessible on the web, which is great for experts to read but inadequate for machines to process. The number of articles grows yearly together with the number of journals, conferences, and researchers [Laa+11]. At the time of writing this article, the lower bound of number of publications available for indexing is crossing the 205 million publications[4]. The most

[4] Indexed with https://www.semanticscholar.org/

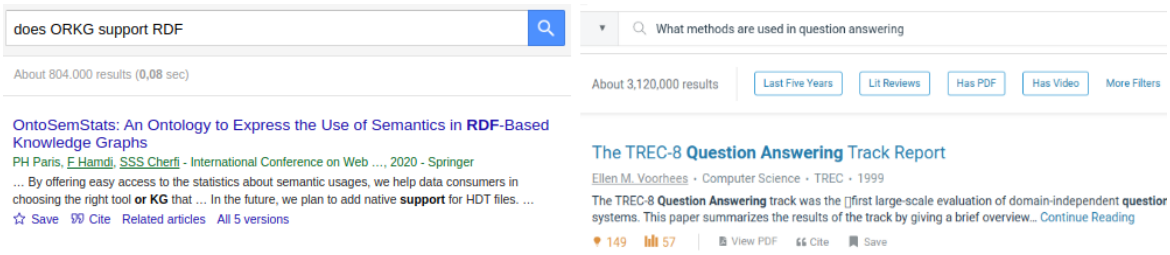


Figure 1.2: **Conventional indexing systems.** On the left side, using "Google Scholar" to search one specific property of a paper or a system. On the right side, using "Semantic Scholar" to look up a collection of approaches relating to a research problem. It is clear that both systems are failing to capture the semantic meaning of the query or retrieve suitable answers.

common format of scholarly documents online is Portable Document Format (PDF), which contains multiple content streams corresponding to different types of content. These streams only provide syntactic structure and no semantic description of what they contain. Processing and indexing such documents has been a topic at the center of many Information Retrieval (IR) systems and search engines for more than a decade now [Gre00; KT00; RR15].

Consider searching for a specific piece of information contained in a paper about some research work or asking about certain approaches in a domain. Existing systems are not able to retrieve a specific value published in some article or collect results from different sources and aggregate them as a suitable answers, rather they just retrieve documents that contain similar keywords as in the query. Figure 1.2 shows two examples of prominent systems used for literature review, and highlights how they are unable to find direct and specific answers to a natural language query. Both systems can retrieve huge numbers of publications that might be related to the input query. But they fall short in narrowing down a precise answer or when trying to examine the meaning of keywords.

When considering to create more advanced systems capable of fetching precise information based on users' natural language queries, more issues arise about how the scholarly documents are created and what are the limitations that exist when processing them. Existing techniques only parse such documents as bags of words and indexes these words (e.g., using TF-iDF [Ram+03]) with the relation "mentioned in" that document, without any more semantic information about what the document is about, what is the metadata referring to, or what are the relations between text entities and resources. Hence, creating intelligent applications requires more in-depth analysis of the scholarly documents and semantic annotations of the content of said documents.

Knowledge Graphs (KGs) serve as a stepping stone for these semantically enabled applications. Structured and interlinked information about the content of scholarly documents can be collected and stored inside a knowledge graph. The KG offers a datastore for the representation of the information as well as modeling capabilities that frame the annotated information within certain structures and schemas, link the information to each other and to external sources, and expose the stored information to be queried via structured query languages. However, populating a knowledge graph with data is a tedious task that is done either manually (takes time, effort, and expertise) or automatically (requires intelligent tools and methods). Automatic methods generally scale better than manual methods due to the resource capacity of machines and the existence of vast corpora of unannotated legacy scholarly

articles. Natural Language Processing (NLP) methods play a major role in the automatic approaches because it can scale, distill expert knowledge, and extract patterns from existing data, all applied for the objective of information extraction and knowledge graph population.

NLP approaches are able to process scholarly text from various formats (e.g., text, HTML, or LaTeX), and extract information from text, annotate and align extracted information to the knowledge graph, and then populate the KG with information. As such, various methods can be used to target factual information, metadata, predefined patterns, schema information, and targeted extractions. Furthermore, given the knowledge graph structure and the scholarly article content, an intelligent system should be able to find and complete any missing relations, resources, and entities in the destination knowledge graph. Once these natural language processing methods are in place and data is extracted at a large scale, then various information retrieval application, and data curation and manipulation tools can exists on top of the extracted structured information, which enables users and other stakeholders to interact, explore, and visualize scholarly knowledge under any set of parameters and constraints. In this thesis, we address the issues and problems mentioned previously by building a semantically enabled scholarly infrastructure powered by a knowledge graph. Specifically, when scholarly information can be represented and stored in a knowledge graph, then we can take advantage of discovered relations and entity interlinking, and automated information extractions, which in turn opens up immense possibilities to help researchers and the scientific community.

1.2 Challenges

Applying natural language processing techniques for information extraction on knowledge graphs does not come without its challenges. In particular we consider the challenges relating to scholarly knowledge graphs. In this section, we describe them and how they intertwine with our research objectives as well as how they relate to the chapters of the thesis. Challenge 1 addresses the scholarly infrastructure requirements. Challenges 2 and 3 address machine processing techniques. Finally, challenge 4 addresses applications of the resulting machinable scholarly knowledge. Figure 1.3 depicts an overview of the research challenges tackled in this thesis and how they relate to each other.

Research Challenge 1: Representing, Modeling, and Querying Semantic Annotations of Scholarly Knowledge. In order to create a scholarly knowledge graph capable of storing and handling scholarly knowledge, a set of features must exist to support various functionalities needed to achieve this goal. First of all, the data has to be represented in a structured manner that allows for expressive interlinking of information. A knowledge graph can be leveraged to represent the information [Aue+07]. However, the data still need to be modeled correctly to allow interoperability with existing and external ontologies and taxonomies describing scholarly knowledge. Provenance information as well as other organizational metadata needs to be incorporated in the modeling of such scholarly data. A semantically-enabled scholarly infrastructure does not only require representing and modeling of information. It also should support querying and manipulating data, as well as application-wide data ingestion. Querying the scholarly data is a requirement for users and applications to interact and interface with the knowledge graph and present the information to various stakeholders. Furthermore, data manipulation operations should be recorded and stored for history management and versioning. Lastly,

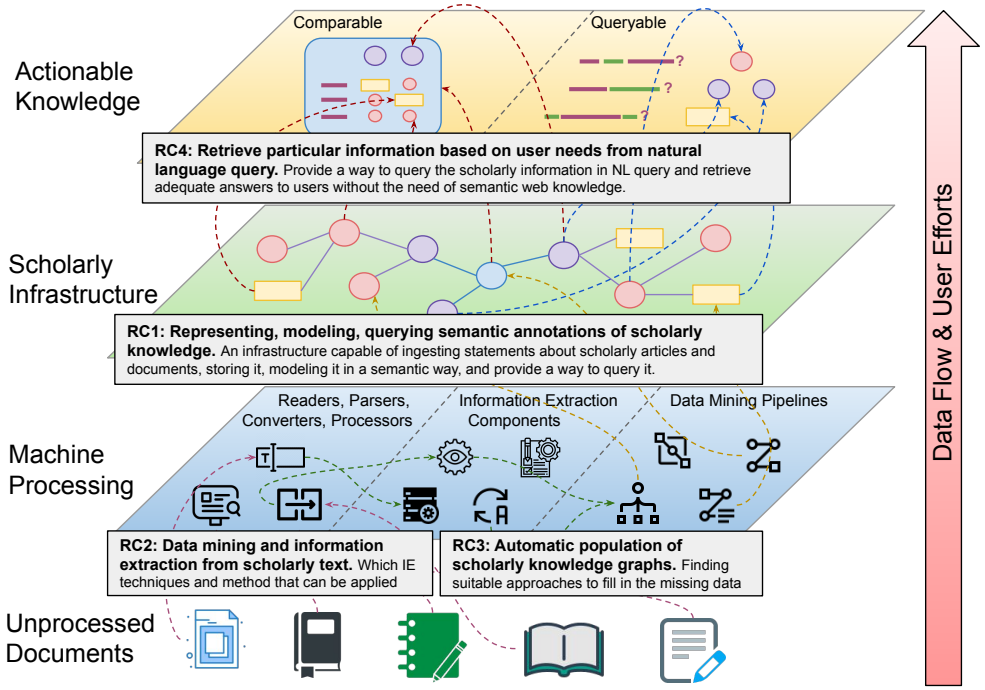


Figure 1.3: **Research Challenges (RCs)**: **RC1** - Representing, modeling, querying semantic annotations of scholarly knowledge. **RC2** - Data mining and Information extraction from scholarly text. **RC3** - Automatic population of scholarly knowledge graphs. **RC4** - Retrieve particular information based on user needs from natural language query.

the semantically structured annotated scholarly data is then ready for machine actionable application consumption. Exploratory, curation, and visualization services need to access the data to perform various functionalities and showcase different perspectives of the scholarly communications.

Research Challenge 2: Data Mining and Information Extraction from Scholarly Text. The output of scholarly communication (as a process) is a published scholarly article in form of digital natural language text documents. These documents are sources for extraction and mining techniques to produce structured scholarly information. However, such documents (usually in PDF format) are not machine readable and do not have any machine-actionable structure that can be leveraged for processing [JPS22]. Hence, they require pre-processing steps to transform them into (semi)-structured formats (e.g., XML), and then apply natural language processing (NLP) techniques. Still, it is not a straightforward task to extract semantically rich information from the resulting textual result. Data mining techniques and information extraction approaches have to be employed to find hidden relations and patterns in the scholarly text, as well as the extraction of relevant statements that are then aligned to a knowledge graph and interlinked with resources and relations. The information that can be extracted from the text can range from factual statements (such as "Text is ambiguous") to more complex multi-level detailed statements (such as "The basic reproductive number of the SARS-CoV-2 in Italy has been estimated to 3.1 in a period of two months"), each with their own extraction criteria. Due to the different knowledge granularity used to describe research contributions in the text, and the complex

knowledge structures that go beyond simple text descriptions, extraction techniques won't be enough by themselves, but need to be complemented with background knowledge and other interlinking methods to connect the information from the text to the information in the knowledge graph.

Research Challenge 3: Automatic Population of Scholarly Knowledge Graphs. In general, the population of a knowledge graph is a time consuming task and usually requires either the manual effort of data annotation and concept alignment, or the existence of a (semi-)structured dataset that can be automatically or manually aligned and added to the knowledge graph. For the task of automatic population, if no structured dataset exists then it is required to use machine-supported approaches to extract statements (or triples) from the natural language text and adding it to the graph. Natural language processing techniques can be leveraged to perform the extraction but it also needs to understand the schema of the graph and be able to align the extracted information to the interlinked concepts (resources, and properties) of the knowledge graph. An approach for automated population (a.k.a. knowledge graph completion) must be able to perform various sub-tasks such as head/tail and relation predication. Head/tail predication is the process of predicting one end of a triple (subject or object) given the property, the other end of the triple, and the destination knowledge graph (or at least its schema) [Che+20]. Relation predication addresses the prediction of the relation (property) between two graph entities forming a complete triple that can be added to a destination graph. The population task on a scholarly knowledge graph is complicated by the intricate knowledge structures and the heterogeneity of knowledge representation in the source text. Such a task would require any NLP approach to have a better understating of the scope of the task as well as any background knowledge (schemas, templates, classes, and relations) contained in the scholarly knowledge graph.

As such, research challenges 2 and 3 serve as the a base layer for research challenge 1, where they conform to the semantic scholarly infrastructure representation and modeling, and populate the knowledge graph from legacy unprocessed scholarly documents.

Research Challenge 4: Retrieve Particular Information Based on User Needs from Natural Language Query. The most natural method of communication for a human is natural language (NL), and this extends to interacting with a machine as well. A natural language posed query would require much processing and transformations to convert from its input form into a more machine-actionable one. Such queries are used to retrieve information from knowledge management systems such as a semantic scholarly infrastructure. Information retrieval techniques performs better when the underlying data source is structured and semantically described. However, the challenge is to understand the user's intent with the natural language query and to locate the answer that best suits their need. Question Answering (QA) addresses the issue of conversion from a NL question to a structured query or an intermediary representation that can operate on a knowledge graph and retrieve answers. Scholarly QA faces challenges when distinguishing schemas and instances in the graph as well as disambiguating the NL query due to the information-dense nature of scholarly data. Furthermore, various types of questions and answers can be posed and expected from a QA system in a scholarly context where each entail different processing techniques and retrieval synopsis. Another obstacle for a QA system over a scholarly knowledge graph is to locate the subgraph that a user means or targets with their input query, to be able to perform an accurate transformation of the query, and answer collection.

Challenge 4 acts as the top of the pyramid of challenges. It builds on everything below to create applications capable of consuming the semantically annotated actionable knowledge extracted from lower layers and deposited in the scholarly infrastructure that it operates on.

1.3 Research Questions

Based on the challenges identified and mentioned in the previous section, we formulate the following research questions.

> RQ1: What are the requirements and possible implementation options for a semantic scholarly knowledge graph representing scientific contributions?

In order to answer this research question (Chapter 4), we prototype a generic scholarly digital infrastructure that is capable of storing, querying, and visualizing structured semantic annotations of key research contributions created by expert users and researchers. We also design an experiment to see the acceptance rate and usability of the infrastructure for unfamiliar users. For this, we leverage the RDF data model to represent scholarly data in a similar manner. Moreover, we integrate automatic techniques for data extraction with manual curation mechanisms and describe their use cases. Finally, we evaluate system usability and evaluate some automated components empirically.

> RQ2: How does the dynamic selection of pipelines based on the input text affect the end-to-end information extraction task?

To answer this research question (Chapter 5), we first analyze isolated community efforts to create information extraction components for various natural language processing sub-tasks. We present a novel approach to combine these components into information extraction pipelines for the blind information extraction based on textual input. Then, we study empirically the effects of error propagation throughout information extraction pipelines. Finally, we perform detailed ablation studies of various components to get insights about the success and failure criteria of such components and how can we take advantage of these findings.

> RQ3: How can we guide information extraction processes on scholarly documents?

To answer this research question (Chapter 6), we study natural language comprehension deep-learning models in the literature. We investigate the efficacy of leveraging text summarization techniques to create structured summaries of scholarly articles, which can be used to complete knowledge graphs or auto-fill literature comparison tables. Moreover, we describe a tailored dataset for this task derived from expertly and manually created semantic annotations. Finally, we evaluate our approach on various models and baselines, describe the limitations of the approach, and investigate the integration in a user curation workflow of a scholarly infrastructure.

> RQ4: What is the impact of task-specific context on scholarly KG completion?

In order to answer this research question (Chapter 7), first we analyze existing state-of-the-art methods of graph embedding and related work. We propose to model the task as a sentence classification problem

that can be tackled by language model techniques. We describe how to enrich the textual representation of a triple to perform more accurate classification. Then, we describe two new datasets adapted from scholarly themed knowledge graphs that can be used for the task of knowledge graph completion. Finally, we evaluate the performance of our proposed solution with respect to three common sub-tasks.

> RQ5: How can we leverage state-of-the-art question answering techniques for scholarly knowledge?

To answer this research question (Chapter 8), we look into existing information retrieval methods and typical question answering systems in the literature. We present an approach to run a question answering technique on tabular views of scholarly knowledge graphs. Also, we describe a question answering dataset of natural language questions posed on literature review tables of a digital scholarly infrastructure. Finally, we empirically evaluate the performance over the dataset with details on variations in questions.

1.4 Thesis Overview

In this section, we present an overview of our main contributions on the research problems investigated throughout this thesis and related scientific publications.

1.4.1 Contributions

Figure 1.4 depicts the main contributions presented in this thesis:

- **Contribution 1:** *Next Generation Infrastructure for Semantic Scholarly Knowledge.*[5] An infrastructure to store, represent, manipulate, and query structured scholarly information. We present the Open Research Knowledge Graph (ORKG), a semantic scholarly infrastructure capable of modeling scholarly communications as a set of interconnected nodes and relations. We outline the architecture of the infrastructure with all the necessary features such as provenance, versioning, and history keeping, as well as extensible applications on top of the structured interfaces. Furthermore, we formalize two key features of the ORKG: the contribution similarity and the contribution comparison systems, and the underlying data model used to create generic domain objects capable of representing all types of structured scholarly information. ORKG is also evaluated in two phases: The frontend part (i.e., the user interface) with a user evaluation conducted by several researchers from various domains, and backend components on performance, coverage ability, and time requirements. Moreover, we present a number of assorted data curation methods that are implemented in the ORKG, such as "paper wizard", "CSV importer", and "contribution editor", as well as some automated tools for user support, thus answering research question **RQ1**.

- **Contribution 2:** *Orchestrating Dynamic Information Extraction Pipelines.* In order to leverage community-wide created natural language processing tools to perform information extraction

[5] This work was conducted by a larger team. My contribution here was to the modeling of the data, backend development, graph querying, and the implementation of automated curation methods.

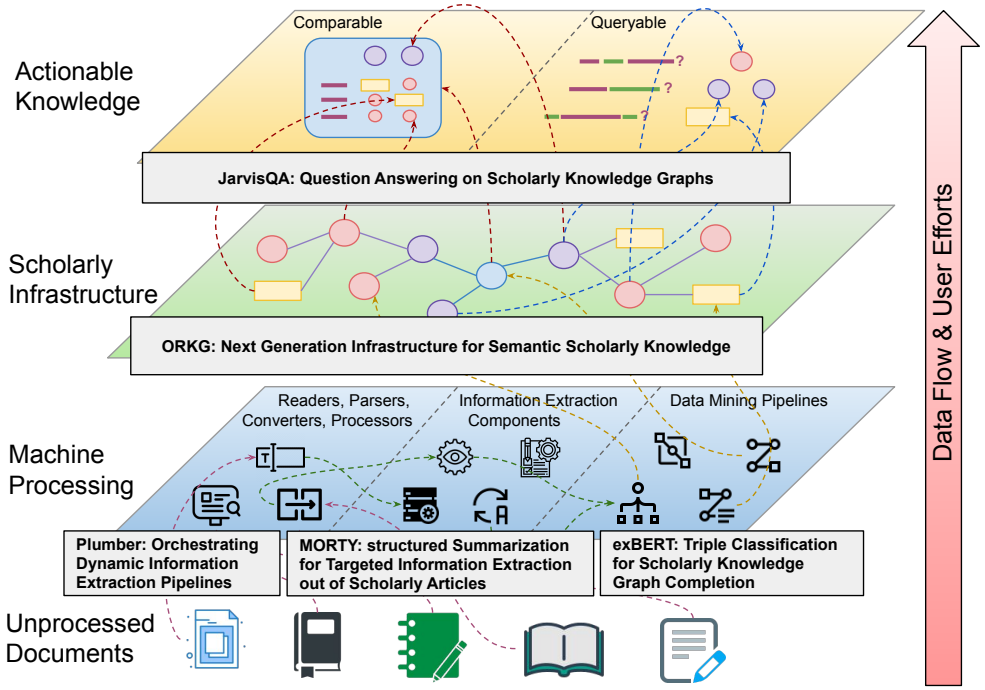


Figure 1.4: **Thesis Contributions**. Five main contributions in this thesis. (i) A next generation semantic scholarly infrastructure. (ii) Orchestrating dynamic blind information extraction pipelines. (iii) Guided and targeted information extraction from scholarly documents. (iv) Scholarly knowledge graph completion via triple classification. And (v) question answering approach for tabular view of scholarly knowledge graphs.

on scholarly data, we propose PLUMBER, a framework capable of generating information extraction pipelines dynamically based on input text. This framework serves as an orchestrator of community created components and tools that perform various information extraction tasks, such as "Named entity recognition and disambiguation", "Text triple extraction", "Relation extraction and linking", and "Coreference resolution". PLUMBER enlists a language model - to capture semantic representation of the input text - that decides what pipeline of components is more suitable to be composed. We formalize how each component of the pipeline is defined per task, and how a complete pipeline is created and executed. Furthermore, we evaluate the performance of our framework against other end-to-end and compositional baselines over three datasets spanning three different knowledge graphs. Moreover, an ablation study is conducted to gain insights on each component integrated in the framework, and to study error propagation per task throughout the life-cycle of pipeline creation and running. Finally, we show case how this framework can be integrated within the ORKG ecosystem and how it would fit into the data curation life-cycle, thus answering research question **RQ2**.

- **Contribution 3:** *Structured Summarization for Targeted Information Extraction from Scholarly Articles.* To make use of existing legacy scholarly articles and documents into (auto-)completing missing information in the knowledge graph and specially for literature survey views on the graph (i.e., comparisons) we bring forward MORTY. An approach that utilizes summarization techniques in its core to extract information directly from documents based on a set

of properties that guide the extraction process. MORTY creates structured summaries from pre-processed scholarly articles' fulltext, then parses out the created summary, and aligns the individual elements to the knowledge graph. Furthermore, we present a sizable dataset of openly accessible article fulltext, with their semantically enriched annotations collected from the ORKG infrastructure. We also empirically evaluate the approach on the newly created dataset using various baselines and methods, and compare it against two commonly used information extraction techniques. MORTY distinguishes itself by only extracting a set of required properties and ignoring any other available information in the text, unlike other approaches that blindly extract all possible pieces of information related or not. Lastly, we present how the approach can be integrated within the ORKG and discuss its value for expert stakeholders, hence answering research question **RQ3**.

- **Contribution 4:** *Triple Classification for Scholarly Knowledge Graph Completion.* In order to populate the scholarly knowledge graph and complete missing information we propose ExBERT. A system capable of performing three subtasks under the umbrella of knowledge graph completion: "head prediction", "tail predication", and "relation predication". These three tasks can be used in conjunction to perform the broader task of knowledge graph completion. ExBERT employs text classification of triples using deep learning language models, by transforming aligned triples into their textual counterparts (based on their labels) and performing a straightforward classification task. We present two datasets collected from scholarly sources. One from the ORKG ecosystem centered around papers and their semantic descriptions. The other is from the PwC[6] repository centered around machine learning papers and their evaluation results. Furthermore, we evaluate ExBERT on three different datasets (i.e., three distinct knowledge graphs) for all subtasks of knowledge graph completion against various knowledge graph embedding techniques. Finally, an in-depth analysis is conducted on the subtasks to note where and when approaches are performing better or worse, as well as knowledge graph related idiosyncrasies that affect the overall performance of the approach, answering our research question **RQ4**.

- **Contribution 5:** *Question Answering on Scholarly Knowledge Graphs.* To prototype question answering methods on tabular views of scholarly knowledge graphs. We describe JarvisQA, a natural language processing interface capable of digesting a NL query about a comparison (i.e., tabular representation of a sub-graph) and retrieving answers for it. JarvisQA takes advantage of advancements in large language models when processing the knowledge graph with the natural language query. Our approach converts the target sub-graph to a coherent textual representation, augments it with additional information, and processes it to detect candidate answers for user questions. We also present a novel dataset of natural language questions posed on a set of carefully selected sub-graphs from the ORKG system, covering wide range of answer types. JarvisQA is evaluated on our newly created dataset using many underlying models to see the performance, and further evaluated on a scholarly-related datasets to study its generalization and feasibility of wide range adoption. Finally, the dataset and the approach are used in the creation of a larger, more intricate scholarly question answering benchmark that is offered openly to the community, answering research question **RQ5**.

[6] https://paperswithcode.com/

1.4.2 Publications

Here is a list of all publication that this thesis builds on in its chapters. A full list of all the publications produced during the doctoral study period can be found in Appendix A.

1. **Mohamad Yaser Jaradeh**, Allard Oelen, Manuel Prinz, Markus Stocker, Sören Auer, *Open Research Knowledge Graph: A System Walkthrough*, In: Doucet, A., Isaac, A., Golub, K., Aalberg, T., Jatowt, A. (eds) Digital Libraries for Open Knowledge, TPDL 2019, Lecture Notes in Computer Science, vol 11799, Springer, Cham, 2019. https://doi.org/10.1007/978-3-030-30760-8_31

2. **Mohamad Yaser Jaradeh**, Allard Oelen, Kheir Eddine Farfar, Manuel Prinz, Jennifer D'Souza, Gábor Kismihók, Markus Stocker, and Sören Auer, *Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge*, In Proceedings of the 10th International Conference on Knowledge Capture (K-CAP '19), November 19–21, Marina Del Rey, CA, USA. ACM, New York, NY, USA, 4 pages, 2019. https://doi.org/10.1145/3360901.3364435

3. Allard Oelen, **Mohamad Yaser Jaradeh**, Kheir Eddine Farfar, Markus Stocker, and Sören Auer, *Comparing Research Contributions in a Scholarly Knowledge Graph*. In Proceedings of the Third International Workshop on Capturing Scientific Knowledge co-located with the 10th International Conference on Knowledge Capture (K-CAP 2019), Marina Del Rey, CA, USA, November 19, 2019. http://ceur-ws.org/Vol-2526/

4. Allard Oelen, **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer, *Generate FAIR Literature Surveys with Scholarly Knowledge Graphs*, In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020. JCDL '20: The ACM/IEEE Joint Conference on Digital Libraries in 2020. https://doi.org/10.1145/3383583.3398520

5. **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer, *Question Answering on Scholarly Knowledge Graphs*, In: Hall M., Merčun T., Risse T., Duchateau F. (eds) Digital Libraries for Open Knowledge, TPDL 2020, Lecture Notes in Computer Science, vol 12246, Springer, Cham, 2020. https://doi.org/10.1007/978-3-030-54956-5_2

6. Auer, Sören, Allard Oelen, Muhammad Haris, Markus Stocker, Jennifer D'Souza, Kheir Eddine Farfar, Lars Vogt, Manuel Prinz, Vitalis Wiens, and **Mohamad Yaser Jaradeh**, *Improving Access to Scientific Literature with Knowledge Graphs*, Bibliothek Forschung und Praxis, vol. 44, no. 32020, pp. 516-529, 2020. https://doi.org/10.18452/22049

7. **Mohamad Yaser Jaradeh**, Kuldeep Singh, Markus Stocker, Andreas Both, and Sören Auer, *Better Call the Plumber: Orchestrating Dynamic Information Extraction Pipelines*, In: Brambilla M., Chbeir R., Frasincar F., Manolescu I. (eds) Web Engineering, ICWE 2021, Lecture Notes in Computer Science, vol 12706, Springer, Cham, 2021. https://doi.org/10.1007/978-3-030-74296-6_19

8. **Mohamad Yaser Jaradeh**, Kuldeep Singh, Markus Stocker, and Sören Auer, *Plumber: A Modular Framework to Create Information Extraction Pipelines*, In Companion Proceedings of the Web Conference 2021 (WWW '21), Association for Computing Machinery, New York, NY, USA, 678–679, 2021. https://doi.org/10.1145/3442442.3458603

9. **Mohamad Yaser Jaradeh**, Kuldeep Singh, Markus Stocker, and Sören Auer, *Triple Classification for Scholarly Knowledge Graph Completion*, In Proceedings of the 11th on Knowledge Capture Conference (K-CAP '21). Association for Computing Machinery, New York, NY, USA, 225–232, 2021. https://doi.org/10.1145/3460210.3493582

10. Allard Oelen, **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer. *Chapter 10. Organizing Scholarly Knowledge leveraging Crowdsourcing, Expert Curation and Automated Techniques*, In: Linking Knowledge: Linked Open Data for Knowledge Organization and Visualization, 181–98, 2021. https://doi.org/10.5771/9783956506611-181

11. **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer, *MORTY: Structured Summarization for Targeted Information Extraction out of Scholarly Articles*, In Proceedings of the 24th International Conference on Asia-Pacific Digital Libraries, (ICADL 2022), 2022. https://doi.org/10.1007/978-3-031-21756-2_23

12. Sören Auer, Dante Barone, Cassiano Bartz, Eduardo Cortes, **Mohamad Yaser Jaradeh**, Oliver Karras, Manolis Koubarakis, Dmitry Mouromtsev, Dmitry Pliukhin, Daniil Radyush, Ivan Shilin, Markus Stocker and Eleni Tsalapati, *SciQA – A Question Answering Benchmark for Scholarly Knowledge*, Under review in Nature's Scientific Reports 2022.

1.5 Thesis Structure

The reminder of this thesis (see also Figure 1.5) is organized as follows. Chapter 2 introduces the prefatory notions and theoretical bases for the research conducted in this thesis. First, it introduces the Semantic Web as a whole as well as its techniques, technologies, and its applications. Second, it defines the semantic web standard for modeling data, including its various serialization formats, and its own query language and protocol. Third, we present a brief history about Natural Language Processing methods and applications. Specifically, we discuss the main categories of NLP methods and what use cases do they address. Fourth, we list some of the most common natural language processing tasks across the textual modalities with some examples. Fifth, we review Information Extraction techniques where we also briefly overview tasks and approaches under information extraction and how they relate to natural language processing. Finally, we dive into two main types of information extraction systems and describe the basic characteristics of each.

In Chapter 3 we present the related work of this thesis. We start with reviewing existing approaches and related initiatives for Semantic Scholarly Infrastructure. We continue with Question Answering techniques in the scholarly and digital libraries areas. Furthermore, we dive into Knowledge graph completion approaches, where we dissect state-of-the-art knowledge graph embedding techniques, as well as deep language model advancements. Finally, we discuss the related work information extraction. We describe solutions and approaches regarding blind information extraction as well as end-to-end solutions of domain specific targeted information extraction.

Chapter 4 presents our vision and implementation of a semantic scholar infrastructure—specifically, the Open Research Knowledge Graph (ORKG)—that can bring the worlds of semantic web and scholarly information/articles together. We begin by describing the problem statement and the weakness of existing methods. Next, we lay the foundation of the infrastructure and its aspects, in particular

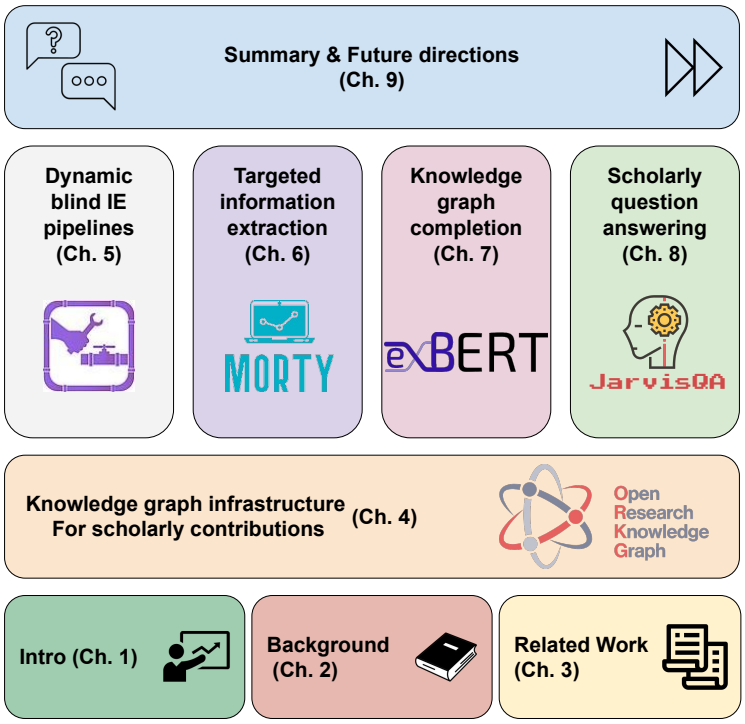


Figure 1.5: **Thesis chapter overview**. A wider look on the topics of the thesis per chapter and how they build on each other. Dependency is indicated by placement vertically.

architecture, required features, and implementation. Finally, we present our preliminary evaluation of selected infrastructure components via empirical experiments of automated components and user evaluations for the user interface.

Chapter 5 presents a novel approach (named PLUMBER) of composing information extraction pipelines from community created and publicly available components. First, we formalize the problem and our proposed solution. Then, we discuss our system architecture and we base it onto published and time-tried architectures. We show how to integrate the research community's individual efforts under one umbrella with more than 40 components. Moreover, we report on our empirical studies of the contribution. We show detailed error analysis to understand the advantages and disadvantages of each system available. As well as ablation studies on individual component level to gain valuable insights about the task as a whole.

In Chapter 6 we discuss a novel system for the extraction of information from scholarly articles and populating literature comparisons in automation. First, we show the need for such systems specifically in the digital library domain and with the exponential growth of publications. Then, we outline our approach with its implementation (code named MORTY). Furthermore, we describe an expert-curated dataset of scholarly text and their structured annotations (i.e., aligned triples). We also, discuss the experimental setup, empirical results, limitation of the approach, and how it integrates in a larger agenda usecase of literature review completion.

Chapter 7 puts forward a method of knowledge graph completion based on triple classification. We start by motivating the problem and looking at existing knowledge graph embedding techniques.

Next, we define our approach in details and describe its workflows. We also, present two datasets collected from scholarly data sources and we leverage them for the evaluation. Moreover, we show extensive evaluation with various knowledge graph embedding baselines and language model ones. Complimented by a set of ablation studies to understand the particularities of the approach and under what circumstances it performs better. Lastly, we discuss the results and summarize the work.

In Chapter 8 we present our contribution of creating and evaluating a Question Answering system on top of scholarly knowledge graphs (a.k.a. JarvisQA). We begin by motivating the issue as an Information Retrieval problem as well as Information Extraction one. We describe a dataset collection effort for one of the early QA systems working on a scholarly infrastructure (i.e., ORKG). Then, we present our system, and review in details different experiments in order to evaluate the robustness of the approach.

Chapter 9 concludes the thesis with a summary of the main results, contributions, and challenges of applying natural language processing techniques on the sensitive topic of scholarly data. Finally we present possible future directions for subsequent research work and open questions for the research community.

CHAPTER 2

Background

In this chapter , we present basic concepts and theoretical foundations for the research conducted in this thesis. First we examine the semantic web technologies as they are the basic building block that this thesis uses as a target format and a destination of resulting artifacts. Then we move on to talk about natural language processing fundamentals as they touch every core chapter of the thesis. And we end with background knowledge of information extraction techniques and methodologies that impacts the work presented throughout this document.

2.1 Semantic Web Technologies

From its name, the semantic web relies on web technologies and improves on them. The web or more formally the world wide web (WWW) is an information space in which documents and other resources live and can be accessed via web browsers and other web consuming clients [JA19].

The web was invented by Tim Berners-Lee in 1989 at CERN [McP09], and then quickly spread like wild fire where institutes around the world started using it. Through the years after its conception, the web evolved to web 1.0 and then web 2.0. These two terms indicated what actions users are able to perform with interconnected resources on the internet, read for web 1.0 and read, write, and interact with web 2.0 [Sim22].

Such resources are of several types and resides on the web in an interconnected format of hyperlinks represented in a hypertext markup language (HTML) that extends the markup language XML [RLJ18].

Though the normal web or (web) is highly usable by users, machine still suffer from not being able to understand what are resources and what do they mean. Due to the nature of how the web stores and represents its resources which is based on text, machines can not understand what is each resources and what type of relations do resources have in between each others.

Thus the inception of the semantic web began by extending existing technologies that the web uses in order to include semantically meaningful information that machines are able to process and deal with. At its core the semantic web relies on a set of standards like Resource Description Framework (RDF), RDF schema, and Web ontology language (OWL) for the purpose of linked resources representation [BHL01a].

Figure 2.1: **RDF Graph** showing two RDF triples. Each triple consist of a subject (resource indicated in an oval shape), a predicate (of a verb, indicated as a directed arrow), and an object which could be a literal (simple values indicated via rectangles) or a resource (a dereferencable entity indicated via ovals).

2.1.1 Resource Description Framework

The Resource Description Framework is a graph based data model that can be used to represent information and resources on the web[1]. The RDF data model is built around the concept of triples. Each triple is composed of subject-predicate-object. RDF makes a distinction between entity types, resources and literals. Resources are entities that can have relations with other resources, literals, and can be further described. Resources can be in the subject or the object position of a triple. Unlike literals where they represent ground values (i.e., strings and numbers) which can only fall in the object position of a RDF triple.

RDF links resources by leveraging the URI attribute of entities. A Uniform Resource Identifier (URI) is a unique sequence of characters that identifies a resource used by web technologies [Fen+11]. Such identifiers are used to refer to other resources and entities that are hosted on the web and interlink them together via a unique descriptor.

A collection of RDF triples forms a RDF graph (Figure 2.1). In this thesis, we use the terms Knowledge Graph (KG) and RDF Graph interchangeably. Usually these KGs represent one or multiple aspects of the world via resources and their relations. Resources (a.k.a. nodes) can reference other resources from graphs via the IRIs (Internationalized Resource Identifier) which are a more generic version of URIs that can represent a bigger spectrum of textual encoding [DS05]. Such graphs can be seen as namespaces that hold knowledge about a certain topic. And as such they are referred to via a prefix that identifies each namespace. Some serializations (further on this in Subsection 2.1.2) indicated these prefixes in the form: `prefix:resource-id` (e.g., dbr:Hannover[2]). Literals in a knowledge graph can be further augmented via XML schema datatypes [BMC+04] and language tags [MMM+04].

All the above is exposed for consumption via native web access interfaces and for machine dereferencing via SPARQL endpoints (more on the SPARQL protocol in Subsection 2.1.3). Figure 2.2 shows a real set of triples from DBpedia knowledge graph [Aue+07] about the city of Hanover, Germany[3].

RDFS or RDF Schema is also part of the W3C recommendation and it lays down the basics of how to define structured and semantically meaningful user-created vocabularies for the RDF data model. RDFS builds on top of RDF by extending to define what is a class and hierarchies between different classes with `rdfs:Class` and `rdfs:subClassOf` respectively. Similarly it allows the definition of properties and hierarchies between such properties via `rdf:Property`, and `rdfs:subPropertyOf`.

[1] https://www.w3.org/RDF/
[2] https://dbpedia.org/resource/Hanover
[3] We assume the following prefixes:
dbr: <http://dbpedia.org/resource/>
dbo: <http://dbpedia.org/ontology/>

Figure 2.2: **Concrete RDF Graph Instance from DBpedia** [Aue+07] depicting a resource (the city of Hanover) and two relations about it. To which country it belongs (Germany) and its population (integer number).

Furthermore, it allows the restriction of membership type of resources with regards to a certain relation via domains and ranges (`rdfs:domain`, and `rdfs:range` for resources in the subject and object position respectively). RDFS also introduces annotation properties for human readability of resources and properties (`rdfs:label` and `rdfs:comment`). Last but not least, RDFS introduces a set of 13 "simple" entailment rules that are used to infer new triples in the knowledge graph from the existing base of triples [HP14].

2.1.2 RDF serializations

RDF model dictates how the data can be modeled in a knowledge graph to deliver semantically meaningful information. Serialization is the syntax that is used to represent the RDF modeling for machines and humans alike. Many serialization syntaxes have been developed each is capable to representing the RDF model as a whole, each have their benefits and detriments.

One type of serialization is called **RDFa** which is a mechanism to embed RDF representation inside existing XML or HTML tags [AB07]. RDFa 1.0 based on XHTML (W3C Recommendation

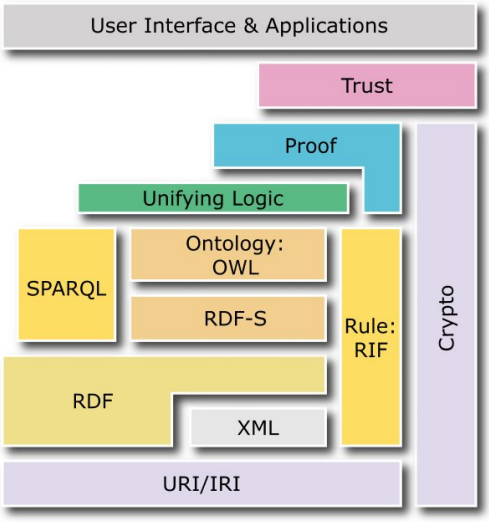


Figure 2.3: **The Semantic Web Technology Stack**, this stack shows how RDF plays a key role as the base many building blocks of the technology [Kha15].

2008). With RDFa, triples are embedded into web documents and can be accessed or extracted by clients and machines. IRIs can be used within (XML and HTML nowadays typically encoded as UTF-8 Unicode) to dereference and identify linked resources. This syntax enables generic RDF annotations in (X)HTML documents by reusing existing HTML attributes of tag elements. Excerpt 2.1 describes a small RDFa snippet that is akin to normal HTML tags with a some extra HTML attributes (e.g., `property`, `resource`, and `vocab`). This serialization has the benefit of tightly integrating with existing web documents and enriching it with semantically meaningful information. Similarly to MicroData [GBM16], RDFa can be used to describe people, events, organizations, and products. Which in turn is digested by search engines (e.g., Rich Snippets by Google [GGH09]) to further enrich and make web pages more interactive.

```
<div vocab="http://xmlns.com/foaf/0.1/">
  <div resource="#john" typeof="Person">
    <span property="name">John Does</span> knows
    <a property="knows" href="#jane">Jane</a>.
    <div rel="#depiction">
      <img style="float:left; width:20%"
       ↪  src="https://i.imgur.com/y7qqVde.jpeg" />
    </div>
  <div resource="#jane" typeof="Person">
    <span property="name">Jane Doe</span> is a colleague at Doe labs.
  </div>
</div>
```

Excerpt 2.1: **RDFa toy example** snippet describing two entities of type `Person`, with a few relations between them.

RDFa has the advantage of handling the integration of human representation (HTML) and the machine one (RDF). It also reuses a number of HTML features rather than creating a completely new subset. Moreover and most important, RDFa enforces the principles of interoperability of metadata [HAV14]. Which are i) Publisher Independence: where every website can use their own representation in their existing web resource. ii) Data Reuse: no need for data to be replicated, and the same sections of RDF and HTML can be used to represent the same content. iii) Self Containment: RDF knowledge although is incorporated in the HTML syntax, it can still be extracted from its special attributes and contained. iv) Schema Modularity: where the modeling attributes of RDFa can be reused for representing different entities. v) Evolvability: the model can be extended with additional attributes and fields. Some disadvantages of RDFa are low readability and backward compatibility issues with RDFa 1.0.

Another type of serialization, which is most common with modeling experts due to its brevity, expressiveness, and human readability is **Turtle** which stands for "Terse RDF Triple Language". Turtle uses URIs in angle brackets (e.g., <http://dbpedia.org/resource/Hanover> which is the URI of the city of Hanover Germany), It encodes literals inside quotes (e.g., "Hannover"@de which is the German name of the city of Hanover). Each triple in the turtle syntax is terminated by a dot. Furthermore, URIs can be shortened by using prefixes (e.g., dbr for <http://dbpedia.org/resource/>).

```
dbr:Hanover     rdf:type              dbr:City;
                dbo:country           dbr:Germany;
                dbo:elevation         "55.000"^^xsd:double;
                dbo:areaCode          "0511";
                dbp:mayor             dbr:Belit_Onay;
                dbp:north             dbr:Hamburg;
                dbp:state             "Lower Saxony"@en;
                rdfs:label            "Hannover"@de.
```

Excerpt 2.2: **Turtle example** snippet describing the city of Hanover, Germany. The turtle syntax here shows different variations of objects that can be used (Resource via URI, literals, typed literals, and language tags literals).

Excerpt 2.2 shows how compact information can be represented with the turtle syntax[4]. Many shortcuts exists in this syntax for readability such as the ";" which indicates that the following triple has the same subject without the need to repeat it. Turtle syntax overtakes other serialization syntax by being concise, efficient to store, and being easy to read by humans. On the other hand, turtle lacks in tool support compared to other serialization formats. Turtle is a subset of other serialization formats like "N-triples" and "N3" which have the greater expressiveness but with a smaller variety in syntax.

Last but not least, **JSON-LD** is yet another serialization format. It stands for: "JavaScript Object Notation for Linked Data". This format integrates better into programming paradigms and data structures due to using the JSON format for data representation. JSON-LD introduces some special keywords to make explicit the semantic context that is communicated via the JSON data, like `@context` which includes mappings of name to IRI, and `@id` to assign IRI to entities and resources in the data.

Excerpt 2.3 shows semantic information about John Lennon in three triples. The `@context` defines what each JSON property maps to in the knowledge graph. Furthermore, it describes what each property has as an expected value in terms of an ID of another resource or a typed literal. JSON-LD shines because of its very good tool support (since every programming language is able to parse JSON structures), it also is a suitable format for exchange between applications due to its compact nature. It suffers from being harder to read for humans compared to turtle, specially in the cases of RDF structures that go beyond property/object pairs of a given subject.

2.1.3 RDF Querying Language

In order to query and manipulate the RDF graph data, we need to use SPARQL, which stands for "SPARQL Protocol and RDF Query Language". SPARQL is a 2008 W3C recommendation and it outlines how the RDF data can be retrieved, edited, and added. As well as, how the query results are formatted to specification [PS08].

[4] We assume the following prefixes are also defined:
dbp: <http://dbpedia.org/property/>
xsd: <http://www.w3.org/2001/XMLSchema#>
rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```
{ "@context": {
    "name": "http://schema.org/name",
    "born": { "@id": "http://schema.org/birthDate",
              "@type": "http://www.w3.org/2001/XMLSchema#date"
    },
    "spouse": { "@id": "http://schema.org/spouse",
                "@type": "@id"
    }
  },
  "@id": "http://dbpedia.org/resource/John_Lennon",
  "name": "John Lennon",
  "born": "1940-10-09",
  "spouse": "http://dbpedia.org/resource/Cynthia_Lennon" }
```

Excerpt 2.3: **JSON-LD example** describing one John Lennon as a resource.

A SPARQL endpoint exposes one or multiple RDF graphs to clients, usually via the HTTP protocol. A client can request a query via GET or POST actions. The query itself, is similar to SQL [CLF09]. It contains a selection expression, filtering expression, and an aggregation expression.

Excerpt 2.4 depicts a simple SPARQL query that have two graph patterns it looks for. First the query looks for a variable (`?person`) that has a membership to the class `foaf:Person` (i.e., is an instance of this class). Second, with these persons found, it looks for the human-readable name property that each instance has and places the value in the variable (`?name`). The `WHERE` clause in the query filters the triples in the graph given certain expressions, while the `SELECT` clause chooses what variables to project in the query results set. We note that the SPARQL syntax uses similar notations to the RDF serialization format of turtle.

SPARQL syntax and expressiveness allows of a large degree of freedom to query complex and nested structures in the graph, with a variety of aggregation and filtering expressions.

In a more complex scenario, Excerpt 2.5 shows a higher degree of functionality that can be achieved via the query language . In this example, the query inserts new triples in the knowledge graph. These triples are the result of a sub-query that selects cities and the maximum population count via aggregation functions (i.e., `MAX()`). It also uses predicate patterns with "|" (i.e., which means one predicate or the other). The query also group results (via the `GROUP` clause) by a variable (in this case by `?city`).

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
PREFIX foaf: <http://xmlns.com/foaf/0.1/>.
SELECT ?name
WHERE {
    ?person rdf:type  foaf:Person;
            foaf:name ?name .
}
```

Excerpt 2.4: **Simple SPARQL query** that fetchs a person and their name using the friend of a friend vocabulary.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX schema: <http://schema.org/>
PREFIX dbp: <http://dbpedia.org/property/>
INSERT {
    ?city dbo:populationTotal ?pop
} WHERE {
    {
      SELECT ?city (MAX(?apop) AS ?pop) {
          ?user schema:location ?city .

        SERVICE <https://dbpedia.org/sparql> {
           ?city dbo:populationTotal | dbp:populationCensus ?apop .
        }
      }
      GROUP BY ?city
    }
}
```

Excerpt 2.5: **Complex SPARQL query** that inserts new knowledge into the RDF graph by selecting some data first.

Furthermore, the SPARQL protocol allows to query data that is not stored in one place but to federate queries through multiple endpoints and RDF graphs in order to collect the results[5].

2.1.4 Beyond RDF

Linked data is a set of best practices for publishing and linking machine-readable data on the world wide web [Zin+21]. Linked data enables the consumption of semi-structured and structured data sources on the web by machines and humans alike. Linked Open Data on the other hand, adds to this by releasing data and resources under an open license which encourages the freedom of reusing and impedes embargoed access. Tim Berners-Lee describes the linked data on a scale of one-to-five stars [BK11]. Each star adds another layer of interoperability and openness.

Figure 2.4 shows the different levels of linked open data. It is clear here, that RDF plays a vital role in achieving this vision. Such objectives pushed data providers and other organizations to publish large linked datasets from different domains, creating the Linked Open Data Cloud (LOD Cloud) [McC+19]. In May 2007, the LOD Cloud had only 12 datasets in total (which included DBpedia, DBLP, and Geo-Names, etc..). While as of May 2021, the cloud grow so large that it has more than 1300 datasets, all linked and openly available[6].

While RDF did do a good job in bridging the gap between the normal web and the semantic web. Its data model lacked a bit and suffered from missing features. Hence, the community extended RDF into RDF* (pronounced as RDF star) [Arn+21]. This extension allows descriptions to be added to predicates (or properties) in a graph such as scores, values, temporal information and provenance

[5] https://www.w3.org/TR/sparql11-federated-query/
[6] https://lod-cloud.net/

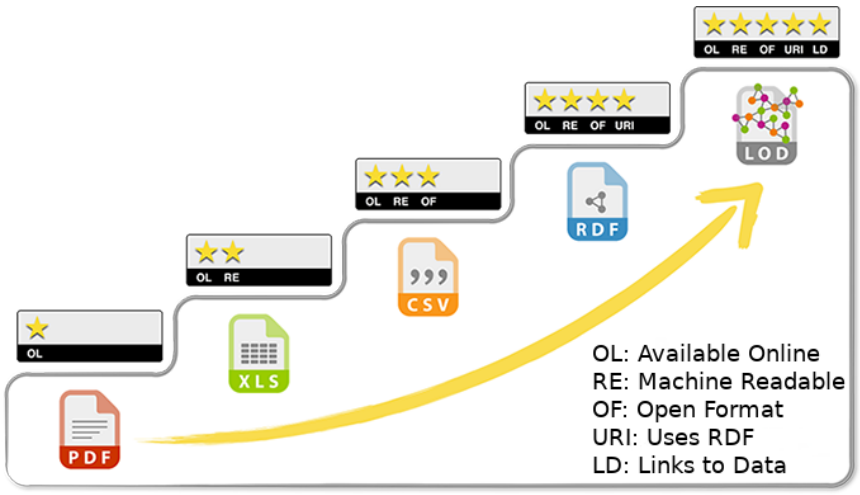


Figure 2.4: **Linked Open Data Levels**. Showing what different file/resource formats online are scored on the five star linked open data score [ZBW20].

```
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX dcterms: <http://schema.org/>
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT DISTINCT * WHERE {
    <<dbr:Hanover  dbp:north  dbr:Hamburg;>>
                dcterms:creator  ex:user101 .
    ex:user101 rdfs:label  "John Doe" .
}
```

Excerpt 2.6: **SPARQL* query example** that fetches provenance data of a single triple (the triple here is one subject), getting the user that created the triple, and then fetching the human readable name.

to predicates in a RDF graph. More formally, RDF* extends the RDF data model by allowing the representation of statements about statements, i.e., metadata can be added to triples, which describe a relation in a graph, while RDF allows statements to be made only about nodes.

Having statements about statements is possible in vanilla RDF through techniques like Reification [NBS14] and Named Graphs [Car+05]. Each method suffers from pros and cons, and thus RDF* is a solution that is supported by the data model, RDF triple stores, and SPARQL engines.

Furthermore, to support the RDF* syntax, SPARQL* (pronounced as SPARQL star) is also introduced to enables expressions and graph patterns for the statements about statements concept[7].

Excerpt 2.6 shows a toy SPARQL* query that can query information about a particular triple or a statement, here the triple creator information which is then further examined.

[7] https://blog.liu.se/olafhartig/tag/rdf-star/

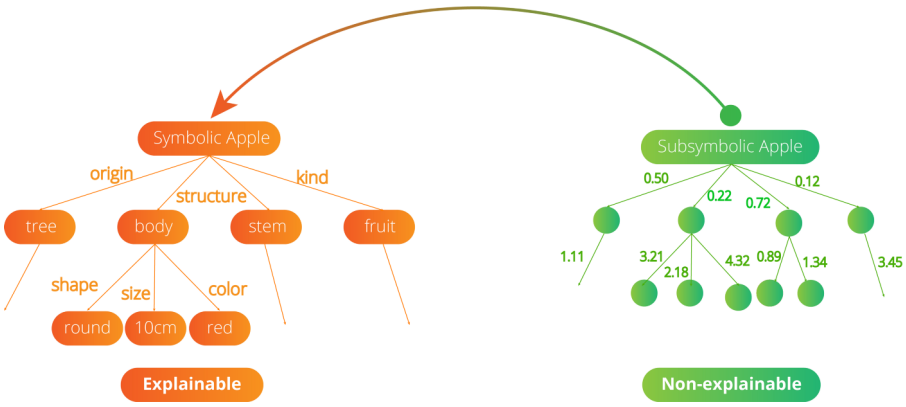


Figure 2.5: **Symbolic NLP vs Statistical NLP**. An example of a simple decision tree rule system for an apple [Yal21]. The tree on the left (`Symbolic`) is clear and easy to understand, while the one on the right (`Statistical`) is a black box.

2.2 Natural Language Processing

Natural Language Processing (NLP) is a sub research field under computational linguistics, computer science and artificial intelligence. Though it is used interchangeably sometime with machine learning, natural language processing focuses on the interactions between machines and human language [Cho20]. In particular, how the computer is able to process, analyze, and comprehend natural language text. Research into NLP seeks to accurately extract information, patterns, insights, categorize, and interactions from within textual documents.

Nowadays, NLP can be seen in variety of online assistant systems and tools. It also spans a wide spectrum of applications and usecases that it can apply to. Like question answering, language translation, and speech recognition [GG12; SB21; KLW19].

2.2.1 History

Alan Turing, one of the most known name in the computer science domain proposed in 1950 the "Turing test" [TUR50] as a machine intelligence measure, which included natural-language-centric tasks such as language generation and automated interpretation.

First models of natural language processing is **Symbolic NLP**. During this phase of NLP evolution, systems tried to emulate human understanding. We humans learn how to structure language through a set of rules, grammar, conjunctions, and a set of vocabulary. In a similar manner, a machine uses rules, lexicon, and semantics in order to emulate the human understanding.

Approaches within symbolic natural language processing systems relied on a variation of hard-coded rules, rule-based parsing, morphology, and references [JW81; Les86].

Figure 2.5 (left figure) depicts a simplified view of a symbolic system utilizing simple rules to make decision and understand the language. The benefits of such classical approaches is that its decision making is transparent and can easily be explainable and comprehensible by humans. They also require less computing power to create such systems.

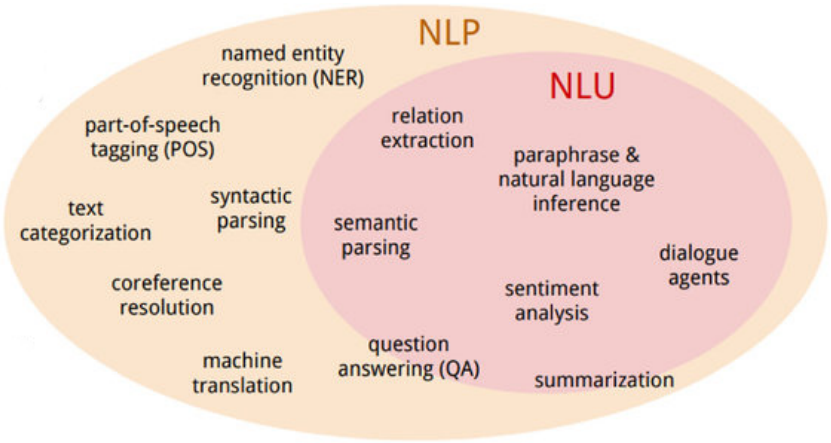


Figure 2.6: **Bird's eye view on NLP methods**. Wider view on common methods and tasks in NLP and NLU (natural language understanding).

In the 1990s, another mode of natural language processing because popular. **Statistical NLP** relies on statistical methods and numeric representations (i.e., weights) to make decisions rather than having a pre-defined set of rules (See Figure 2.5 - right figure). During this period, and due to the growth of the world wide web, significant amount of data became available online and with this statistical methods boomed and became quite popular and used for a variety for applications and usecases [Xia08]. Statistical NLP is also made possible with the increase of computational power [Moo98] that enabled mass adaption of these methods and rapid progress in research.

More recently, **Neural NLP** became the norm. This style of natural language processing, relies heavily on deep neural networks and language representation learning. These methods became so prominent due to the state-of-the-art results achieved by language models in various language understanding tasks over multiple domains and in multiple languages. Moreover, wide adoption of these techniques is happening in a diversity of topics and applications such as healthcare [TB21], and shopping [Gon+19].

2.2.2 Methods

Since NATURAL LANGUAGE PROCESSING (NLP) is a subfield of ARTIFICIAL INTELLIGENCE (AI), many of the methods and techniques applied in NLP applies to other applications of AI like in machine learning.

As mentioned previously, a wide spectrum of styles and approaches are under the NLP umbrella. Some methods rely on a set of hand-coded rules, associated with a lookup dictionary. Other methods, take advantage of learning techniques to overcome the overhead of creating hand-produced rules. There learning techniques are applied during automated learning approaches to spot and learn common patterns and cases, unlike writing rules that has to account for each specific case and would require manual and expert efforts. Furthermore, learning methods take advantage of statistical inferenceing to generalize findings over unseen, unfamiliar, and erroneous input. However, with all the benifits

of statistical methods, it remains true to commonly use symbolic approaches when the amount of training data is insufficient, as well as, for tasks such as preprocessing i.e., tokenization [Sol+18], and postprocessing [Lia+20b].

Probabilistic models are also at the core of NLP methods and automatic learning approaches. These models takes as input a large set of "features" that are engineered from the input dataset. With these numerical features, probabilistic and statistical models are able to make decision based on these read-valued weights for each input feature. When used as part of a broader system, such models offer the benefit of expressing the relative certainty of many different potential responses rather than just one, resulting in more trustworthy results and insights. Some of the prominent methods under this model are: decision trees [Sch10], and hidden markov models [Edd04].

With the neural turn, statistical methods have been mostly replaced by neural networks, due to their inherited problem of elaborate feature engineering. Hence, word embeddings model [Zuc+15] were created to capture the semantics of words in text and create said features automatically. These methods are also used to improve the end-to-end learning performance of down-stream tasks e.g., question answering and summarization, rather than relying on a pipeline of separate tasks. In other research areas, neural methods dominated due to its state-of-the-art performance on several tasks like machine translation and sequence-to-sequence transformations.

2.2.3 Common NLP tasks

Numerous NLP tasks are researched some has direct real-world applications, and some are considered sub-tasks and they facilitate solving a bigger piece of the puzzle [Kha+16]. Here we list some of the most commonly researched topics in the NLP community and are needed for the theoretical understanding of NLP techniques applied in this thesis.

The first category of NLP tasks is **morphological methods**, which focuses on the internal structure of words.

- **Part-of-speech tagging**: determines what is the part of speech for each word in a sentence. Words can have several parts of speech depending on the context and its position in the sentence.
- **Stemming**: which is the process of converting words with their derivations and variations into their base form (or stem), with the help of a set of rules.
- **Lemmatization**: to return the lemma (dictionary base form of a word) after removing certain endings. Lemmatization is different than stemming by relying on a dictionary rather than a set of rules.

Another category of tasks (**Syntactic analysis methods**) are pivotal for the syntactic structure of text and specially sentences.

- **Sentence segmentation**: finds the boundaries of a given sentence, which is usually marked with punctuation marks and periods.
- **Grammatical inference**: is the process of generating a set of formal grammar that is able to describe the syntax of a language (natural or otherwise).

- With **Parsing**: we are able to create a parse tree of a sentence. Dependency parsing and constituency parsing try to find a meaningful and grammatically sounds parse of a sentence that make sense to humans.

A third category considers the semantics of individual words and sentences within their context. **Lexical and relational semantics** are methods that seek to identify the semantics and disambiguate parts of the natural language text.

- **Named entity recognition**: determines which word(s) in the sentence are mapped to proper names i.e., people, places, and organizations.
- **Sentiment analysis**: aims to extract information of subjective nature to determine the polarity of specific subject.
- **Word-sense disambiguation**: which aspires to finding the suitable meaning of a word in the context, since words can have multiple meanings.
- **Relationship extraction**: is the process of figuring out the relationship in between named entities.
- **slot filling**: tries to assign labels to individual words or complete terms to indicate their semantic role.

One more category of methods (**Discourse methods**) go beyond the individual sentence semantics.

- **Textual entailment**: is the task of determining if a text fragment is an entailment of another given that the second one is true.
- **Coreference resolution**: aims at identifying and disambiguation mentions of the same object, such as pronouns or acronyms.
- **Topic identification**: in which the topics in a sentence are recognized and segments of the text is separated based on the identified topic.

The last and most recognized category (**High-level NLP**) can be seen as a more abstract tasks that make use of several other NLP subtasks to achieve the desired objective. Some of these tasks can be reflected back on Figure 2.6 as they are either specific to language understanding (NLU) or just a more general NLP task.

- **Conversational systems**: which are systems that engage in conversations with humans in the form of turns of questions and replies.
- **Machine translation**: is the task of converting a fragment of text from one language to another, retaining all the semantic meaning and the grammatical soundness.
- **Text summarization**: in which a system is able to transform and long chunk of text into a readable summary that is considerably shorter and it retains the core information.
- **Question answering**: that determine an answer for a natural language text. Different modality of these system exists that operate of open-ended knowledge or domain specific and source particular.

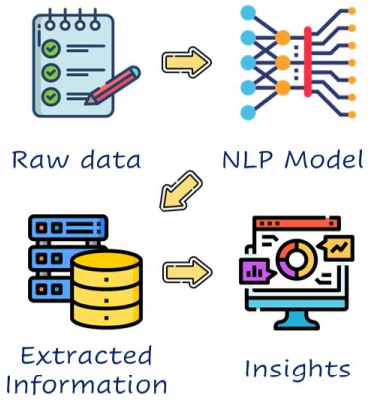


Figure 2.7: **Usual Information Extraction Workflow**. High-level view of a normal IE workflow that involves a NLP component [M21].

2.3 Information Extraction

Information Extraction (IE) is the process of extracting information from unstructured textual sources in order to locate entities, classify them, and store them in a database is known as information extraction. Semantically improved information extraction (also known as semantic annotation) connects things in a knowledge graph with their semantic descriptions and relationships. This technique overcomes numerous difficulties in enterprise content management and knowledge discovery by adding metadata to the extracted concepts.

Information extraction can be considered in part related to NLP as it tries to uncover hidden information or provide structure to natural language text. More recently these activities went in the direction of multimedia and multimodality processing from non-textual sources such as audio, video, and images.

One major goal of information extraction is to allow the conversion of previously unstructured data into more strongly structured, well defined, and semantically annotated data.

Information extraction is part of a greater puzzle and bigger problems. It usually serves under the hood of larger disciplines such as "Information Retrieval"[8] or "Natural Language Processing".

2.3.1 Tasks and Approaches

Information extraction tasks are tightly intertwined with other data mining and text simplification activities and don't only apply to the IE domain. IE tasks span a wide range of functionality and objective based on the target goal of the task. Some of the more common tasks in IE are:

- **Knowledge graph population**, which is a tasks specialized in extracting knowledge from natural language text into the form of triples and collecting it in a graph after interlinking it. This type of tasks involves a collection of sub IE tasks to perform the greater objective. Named entity recognition and disambiguation, coreference resolution, relation recognition and linking, to name a few.

[8] Information retrieval doesn't depend on IE for it to work, rather it would benefit from it for data population, retrieval, and search operations.

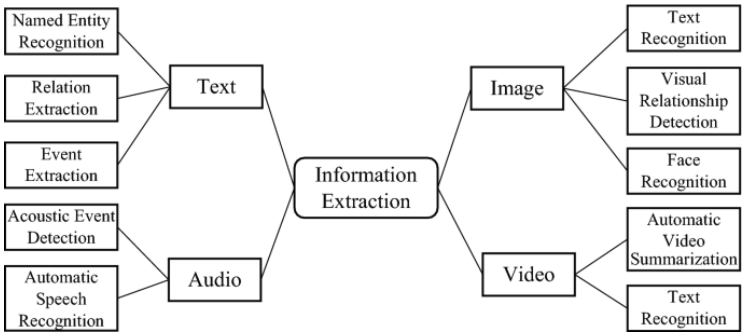


Figure 2.8: **IE tasks over multimedia**. high level view of some information extraction tasks for different data types [AA19].

- **Terminology mining**, aims at automatically and with precision extracting relevant terms from a given corpus or a text fragment. This task is important to model communities and categorize natural language content into suitable classes.

- **Semi-structured extractions**, which is a task that focuses on using semi-structured data representations i.e., tables, to extract strongly structured information in the form of interconnected triples. Table extraction is a subtask contained in this task, that focuses on locating and converting tables in documents to a more machine readable versions (e.g., store it in a database, or a KG). Furthermore, comment extraction is yet another subtask that thrives to extracting textual comments from notes, asocial posts, and tables.

- **Form completion**, which is the task of extracting a fixed set of fields from a document and assign it to a predefined template in a form. This type of task could entail other subtask for extraction of specific type of information, such as, temporal, or event information.

Similarly to the methods that NLP uses, information extraction employs similarly categorized approaches. Approaches of IE can be categorized into three main areas.

- **Hand crafted rules**, which could be some parsers, regular expressions, or pattern matching grammar.

- **Classifiers based**, which relies to classifying a piece of information into one of multiple classes i.e., type of extracted information . Classifiers are of generative [Leu07] or discriminative [KC02] nature.

- **Sequence models**, that understand the semantics of natural text and are able to convert or condense the information into the core components that are then converted to structured elements.

Figure 2.8 depicts some common IE tasks across different modalities of data sources. IE aspects addressed in this thesis pertain only to the textual part of IE and not other modalities.

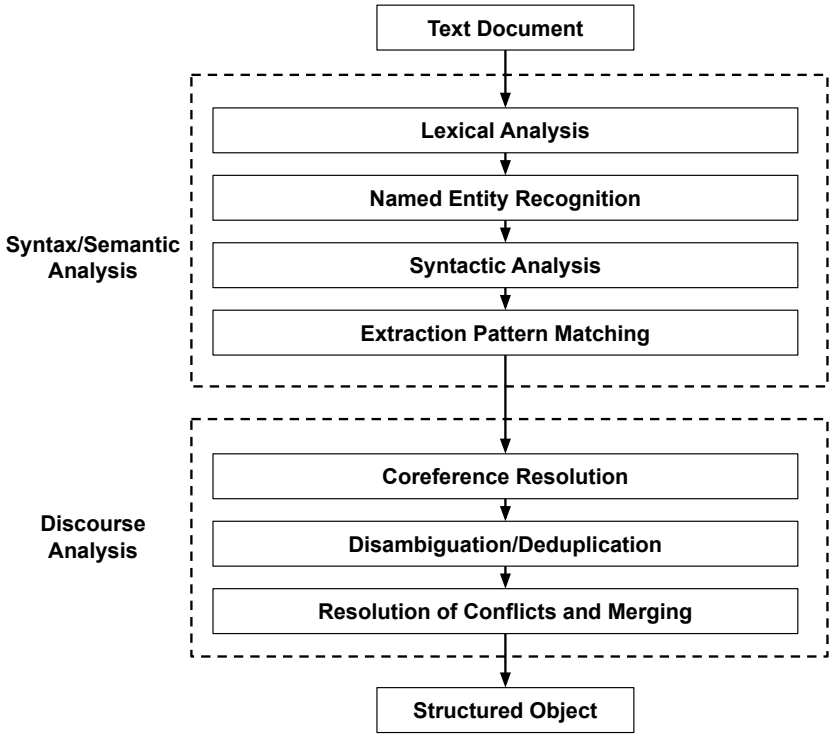


Figure 2.9: **Typical stages in OIE process**, shows how tightly NLP methods are integrated with IE methods

2.3.2 IE Types

Due to the fact that the web contains a huge amount of unstructured information in the form of text and other modalities. Creating domain specific tools or methods proved to be an unsuitable approach to IE on the web. These challenges of extracting information from the web lead to a focus of creating an **Open information extraction** (OIE) tools. OIE is a novel extraction paradigm that seeks to scale linearly to handle web-scale volumes of text and intends to be able to discover and extract an unlimited number of relations while avoiding domain-specific training sets.

An Open IE system, for instance, may operate in two phases. It would first learn a generic model of how relationships are conveyed in a given language. Second, it might use this model to build a relation-independent extractor with a corpus as its sole input and a collection of extracted tuples that are instances of a potentially infinite set of relations as its output. Unlexicalized characteristics like part-of-speech tags, and domain-independent regular expressions would be used by such a system to build a general model of how relations are expressed in a specific language [Etz+08].

Usually, open information extraction systems operate on a proposition (i.e., textual expression of a fact) level. Hence, OIE systems look for relations and the two arguments (subject, and object) that go with this relation. Various systems in OIE try to address propositions differently with a different set of assumptions, such as grammatical clauses, factuality level, meaningful relations, and reification.

In the complete opposite direction **Targeted Information Extraction**, work on a small number of relations for a specific pre-selected domain, over a certain corpora. And as such, many tools and approaches have been created by the research community to address these issues.

Some targeted IE approaches work on only a specific data source (e.g., CSV, XML, or relational data). Other systems, rely on a mapping rules and specific vocabulary for the extractions. Furthermore, some approaches are completely automatic and some require human feedback at some or all throughout the process. What joins these IE systems, is that they are build on a specific set of rules and patterns. They are usually trained on a hand crafted datasets within a perimeter of specific assumptions, which the IE systems fails to work outside.

Both open and targeted information extraction involve various subtasks (e.g., Entity linking, relation extraction, and word sense disambiguation) to work. They just employ these techniques differently and within various parameters.

CHAPTER 3

Related Work

A crucial part of research is to list related work and show how own contributions are different from other efforts. In this chapter, we review state-of-the-art approaches and studies that are related to our work in this thesis. Figure 3.1 shows an overview of the topics addressed throughout this thesis, and depicts their relation to one another. For each topic we present, the approaches and limitations of state-of-the-art existing systems and how they relate to our individual contributions. First in Section 3.1 we present the existing initiatives for scholarly knowledge infrastructure within the digital library area, with details about data modeling and knowledge representation. Then, Section 3.2 presents a review of question answering approaches for tailored domain specific areas and for open ended ones. In Section 3.3 we discuss existing solutions for the completion of knowledge graphs and their connection to language models. Finally, Section 3.4 reviews methods for information extraction via natural language processing sub-asks and end-to-end approaches, as well as related transformer models.

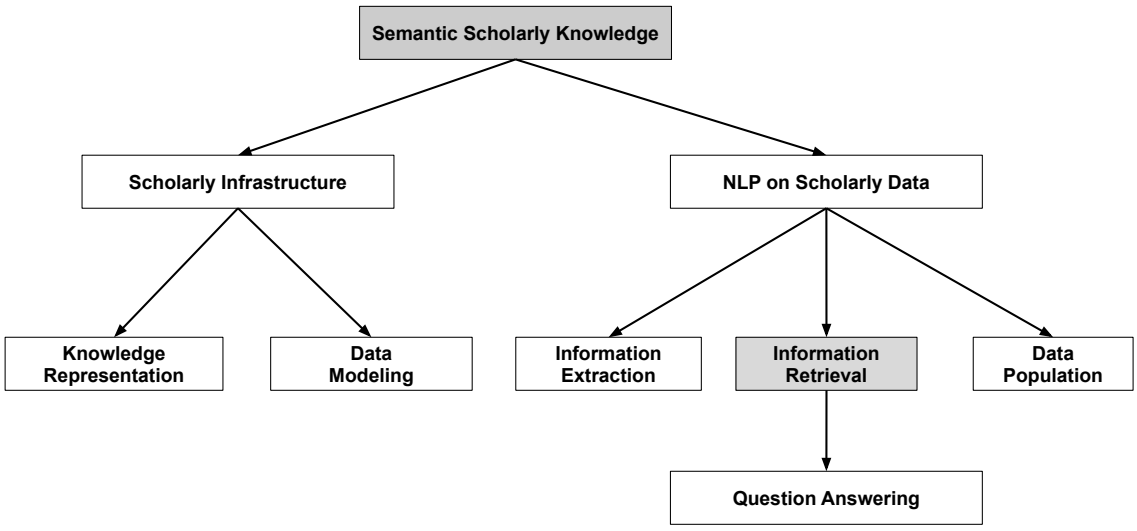


Figure 3.1: **Related work topics**. A list of topics that are related to the contribution of the thesis. Shaded boxes indicate topics that are not addressed directly but have some relevance to other topics.

3.1 Semantic Scholarly Infrastructure

Representing encyclopedic and factual knowledge in machine actionable form is increasingly feasible. This is underscored by knowledge graphs such as Wikidata [VK14], domain-specific knowledge graphs [Che+18] as well as industrial initiatives at Google, IBM, Bing, BBC, Thomson Reuters, Springer Nature, among others.

In the context of scholarly communication and its operational infrastructure the focus has so far been on representing, managing and linking *metadata* about articles, people, data and other relevant entities. The Research Graph [AW17] is a prominent example of an effort that aims to link publications, datasets and researchers. The Scholix project [Bur+17], driven by a corresponding Research Data Alliance working group and associated organizations, standardized the information about the links between scholarly literature and data exchanged among publishers, data repositories, and infrastructures such as DataCite, Crossref, OpenAIRE and PANGAEA. The Research Objects [Bec+10] project proposed a machine readable abstract structure that relates the products of a research investigation, including articles, data and other research artifacts. The RMap Project [HDD15] aims at preserving "the many-to-many complex relationships among scholarly publications and their underlying data". Sadeghi et al. [Sad+17] proposed to integrate bibliographic metadata in a knowledge graph.

Some initiatives such as the Semantic Publishing and Referencing (SPAR) Ontologies [Per14] and the Journal Article Tag Suite [DSM15] extended the representation to *document structure* and more fine-grained elements. Others proposed comprehensive *conceptual models* for scholarly knowledge that capture problems, methods, theories, statements, concepts and their relations [Haro1; De +06; BFY08; Mei17]. Allen, R.B. [All12; All17] explored issues related to implementing entire research reports as structured knowledge bases. Fathalla et al. [Fat+17] proposed to semantically represent key findings of survey articles by describing research problems, approaches, implementations and evaluations. Nanopublications [GGV10] is a further approach to describe scholarly knowledge in structured form. Natural Language Processing based Semantic Scholar [Amm+18] and the Machine Learning focused (PWC)[1] are related systems. Key distinguishing aspects among these systems and the ORKG are the granularity of acquired knowledge (specifically, article bibliographic metadata vs. the materials and methods used and results obtained) and the methods used to acquire knowledge (specifically, automated techniques vs. crowdsourcing).

There has been some work on enriching documents with semantic annotations. Examples include Dokie.li [Cap+17], RASH [Per+17] or MicroPublications [CCG14] for HTML and SALT [Gro+07] for LaTeX. Other efforts focused on developing ontologies for representing scholarly knowledge in specific domains, e.g., mathematics [Lan13]. Table 3.1 shows a comparison of the semantic representation of the ORKG ecosystem and other approaches discussed here. This table is created via the ORKG comparison system and it compares several papers across multiple properties.

A knowledge graph for research as proposed in this thesis must build, integrate and further advance these and other related efforts and, most importantly, translate what has been proposed so far in isolated prototypes into operational and sustained scholarly infrastructure. There has been work on some pieces but the larger puzzle has obviously not been solved or more of scientific knowledge would be available today in structured form.

[1] http://paperswithcode.com

Table 3.1: **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). This comparison looks at the acquisition methods of each system, how it works to ingest data, data type it supports, the scope of collected information, and several other properties [Oel22].

	Acquisition	Optics	Data type	Scope	Discourse	High level claims	Metadata	NL statements	Supports research data	Knowledge representation
ORKG [Jar+19b]	Authors Crowdsourcing Automatic	Prospective Retrospective	Free text	Summary	Partially	✓	✓	✓	Partially	Metadata RDF
Nanopublication [GGV10]	Authors Automatic	Prospective Retrospective	Free text	Statement level	Partially	✓	✓	✗	✓	RDF
Micropublication [CCG14]	Authors	Prospective	Free text	Full paper	✓	✗	✓	✓	✓	OWL RDF
RASH [Per+17]	Authors	Prospective	Quoted text	Full paper	✗	Partially	✓	✓	✗	HTML RDFa
Dokie.li [Cap+17]	Authors	Prospective	Quoted text	Full paper	✓	✗	✓	✓	✗	HTML RDFa

Table 3.2: **Comparison of QA benchmarks.** Comparing a set of common question answering benchmarks across different dimensions [CK22].

	Questions	Answers	Domain	Formal language	Knowledge base	Language	Paraphrases	Question types
SciQA [Aue+22]	100	✓	Scholarly communication	Community	ORKG	EN	✓	Complex Factoid Non-factoid
LC-QuAD 2.0 [Dub+19]	30000	✗	Encyclopedic	SPARQL	DBpedia Wikidata	EN	✓	Complex Factoid Non-factoid
QALD-9 [Usb+18]	408	✓	Encyclopedic	SPARQL	DBpedia	EN NL FR DE IT	✗	Non-factoid
Yih et al. [Yih+16]	4737	✓	Encyclopedic	SPARQL	Freebase	EN	✗	Factoid
Diefenbach et al. [Die+17]	21957	✓	Encyclopedic	SPARQL	Wikidata	EN	✗	Simple

3.2 Scholarly Question Answering

Question Answering (QA) is an important research problem frequently tackled by research communities in different variations, applications, and directions. QA is an overlap between information retrieval, data mining, and language comprehension. Table 3.2 provides an overview of existing QA benchmarks and how they compare to each other. Multiple types and modalities of question answering systems exists and here we highlight two broad categories.

3.2.1 Open Domain

In open domain question answering, various systems and techniques have been proposed that rely on different forms of background knowledge. Pipeline-based systems, such as OpenQA [Mar+14], present a modular framework using standardized components for creating QA systems on structured knowledge graphs (e.g., DBpedia [Aue+07]). Frankenstein [Sin+18] creates the most suitable QA pipeline out of community created components based on the natural language input question. QAnswer [Die+19] is a multilingual QA system that queries different linked open data datasets to fetch correct answers. Diefenbach et al. [Die+18] discussed and compared other QA-over-KG systems (e.g., gAnswer [Zou+14], DEANNA [Yah+12], and SINA [She+15]) within the context of QALD "Question Answering over Linked Data" challanges [Lop+13].

Other types of QA systems rely on the raw unstructured text to produce the answers. Many of these systems are end-to-end systems that employ machine learning to mine the text and retrieve the answers. Deep learning models (e.g., Transformers) are trained and fine-tuned on certain QA datasets to find the answers from within the text. ALBERT [Lan+19] is a descendent of BERT [Dev+19] deep learning model. At the time of writing this thesis, ALBERT holds the fourth top position in answering the questions of SQuAD [Raj+16]. Such techniques model the linguistic knowledge from textual details and discard all the clutter in the text [Zin12]. Other similar approaches include SG-Net [Zha+19b], which uses syntax rules to guide the machine comprehension encoder-transformer models.

Tabular QA systems are also diverse and tackle the task with different techniques. TF-IDF [Ram+03] is used to extract features from the tables and the question, and to match them. Other models such as semantic parsers are used by Kwiatkowski et al. [Kwi+], and Krishnamurthy and Kollar [KK13]. Cheng et al. [Che+17] propose a neural semantic parser that uses predicate-argument structures to convert natural language text into intermediate structured representations, which are then mapped to different target domains (e.g., SQL).

Another category of table QA systems is neural systems. TableQA [VS17] uses end-to-end memory networks to find a suitable cell in the table to choose. Wang et al. [Wan+] propose to use a directional self-attention network to find candidate tables and then use BiGRUs to score the answers. Other table oriented QA systems include HILDB [DKV13] that converts natural language into SQL. Table 3.3 describes a collection of question answering systems across the different tasks that they perform.

3.2.2 Domain Specific

In the plethora of systems that the community has developed over the past decade, no system addresses the scholarly information domain, specifically. Throughout this thesis, we propose a system to fill this gap and address the issues of QA on scholarly tabular data in the context of digital libraries (specifically with the ORKG[2]) Infrastructure.

Though a variety of QA techniques exist, Digital Libraries (DL) primarily rely on standard information retrieval techniques [Sch97]. We briefly analyze and show when and how QA techniques can be used to improve information retrieval and search capabilities in the context of digital libraries. Since DLs have different needs [Hero6; Sch97]; QA systems can improve information retrieval availability [Blo+07]. We argue that, Knowledge Graph based QA systems (or KG-QA) can work nicely within a DL context (i.e., aggregate information, list candidate answers). Nevertheless, the majority of the existing scholarly KGs (such as MAG [Sin+15], OC [PS20]) focus on metadata (e.g., authors, venues, and citations), not the scholarly knowledge content. which is something a QA system can focus on but not different from a faceted search any digital library interface provides. Moreover, in the current state of the KGs addressing the content information, not the metadata, it would be rather useless to develop a QA system on that due to the lack of suitable data.

Another category of QA systems works on raw text, an important approach for digital libraries. However, such systems are not fine-tuned on scholarly data; rather, they are designed for open domain data. Furthermore, many of the end-to-end neural models have a built-in limitation [Yin+15] (i.e., model capacity) due to the architecture type, and as such cannot be used out of the box. Some systems circumvent the problem of capacity (i.e., the inability to feed the model large amounts of text) by having a component of indexing (e.g., inverted index, concept and entity recognition) that can narrow down the amount of text that the system needs to process as the context for questions. More recently, language models are created specifically to accommodate cases of larger input text such as Longformer [BPC20], and BigBird [Zah+20].

3.3 Knowledge Graph Population

The KG completion task utilizes different methods and techniques; we categorize these techniques into two main categories.

3.3.1 Knowledge Graph Embeddings

Knowledge Graph Embeddings (KGE) are classified into several categories: translation, semantic matching, and neural network-based models [CZC18]. The matrix factorization-based graph embedding learns the representations based on the statistics of global pairwise similarities. HSL [SJY08] and LGRM [Yan+10] are some examples of graph Laplacian eigenmaps-based graph embeddings. Deep learning-based graph embeddings such as DeepWalk [PAS14] uses truncated random walks to find a missing link. Other deep learning models like DeepCase use Markov chain-based random walk on top of a GRU [Cho+14]. Other non-random walk Deep learned GE models, such as GCN [KW17], rely on graph convolutional networks for learning the information embedded in input graphs.

[2] https://orkg.org/

Table 3.3: **comparison of question answering systems and the tasks that these systems address.** A comparison focusing on the main four tasks in some types of QA systems (Disambiguation, phrase mapping, question analysis, and query construction) [JO19].

	Disambiguation	**Phrase mapping**	**Query construction**	**Question analysis**
DENNA [Yah+12]	LIP Local disambiguation	Knowledge base labels	Using information from the question	Dependency parser NE n-gram strategy POS handmade
SemGraphQA [BGL15]	Local disambiguation	Knowledge base labels Lucene index or similar Redirects WordNet/Wiktionary	Using information from the question	Dependency parser NE n-gram strategy
CASIA [He+14]	Local disambiguation MLN	Distributional semantics Knowledge base labels Redirects Using extracted knowledge	Using machine learning	Dependency parser NE n-gram strategy POS learned
Xser [Xu+14]	Local disambiguation Structured perceptron	Knowledge base labels PATTY Wikipedia specific approaches	Using information from the question	Phrase dependencies and DAG POS learned
POMELO [Ham+14]	Local disambiguation	Knowledge base labels	Semantic information	NE n-gram strategy
ISOFT [PSL14]	Local disambiguation	Distributional Semantics Knowledge base labels Lucene index or similar PATTY	Using templates	Dependency parser EL tools POS handmade
Treo [FC14]	Graph search Local disambiguation	Distributional Semantics Knowledge base labels Lucene index or similar	Not generating SPARQL	Dependency parser POS handmade

ProjE [SW17] is another DL neural network model that deals with the graph as a ranking problem and optimizes ranking score vectors. Translational models use distance-based scoring functions to assess the plausibility of a triple (h, r, t). Such models evaluate the plausibility by the distance between the head entity (h) and the tail entity (t), which is typically done by performing a translation operation by the vector (r) [Wan+17]. Bordes et al. [Bor+13] proposed TransE as a representative model of translational graph embeddings; it uses a negative translational distance function for scoring. Work in [Nay+19] modified TransE for adapting it in scholarly domain. TransR [Lin+15] builds entities and relations embeddings by projecting them into different spaces, then building translations.

Other models rely on semantic matching, which internally uses similarity-based scoring functions. RESCAL [NTK11], DistMult [Yan+14], and all their extensions are representatives of this category. Such models employ different scoring functions. For instance, DistMult uses a bilinear function $f(h, r, t) = \langle h, r, t \rangle$. These methods conduct KG completion using the structural information from the triples disregarding other external information, in particular entity types, logical rules, or textual descriptions. For example, Socher et al. [Soc+13] represented entities by averaging word embeddings extracted from their labels. Other approaches such as [Wan+14a] chose to jointly embed entities and words into one vector space via aligning Wikipedia anchors and entity labels.

Convolutional neural networks (CNNs) are used by Xie et al. [Xie+16] to encode word sequences extracted from entity descriptions into embeddings. Semantic space projections (SSP) [Xia+17b] jointly learn embeddings by characterizing correlations between facts and textual descriptions. A common downfall of these techniques is that they learn the identical embeddings representation of entities and relations and not accounting for the different meanings that words might have in various text descriptions or different weights in different triples.

3.3.2 Language Models

Language Models (LM) that represent nodes and words can be split into two classes: feature-based and fine-tuning approaches.

Widely used word embedding techniques such as Word2Vec [Mik+13] and RDF2vec [RP16] aim at adopting features to learn context-independent word embeddings. GloVe [PSM14] is another technique that strives to find a global representation for words based on specific features. FastText [Boj+17; Jou+16] is a technique that uses Skipgram or Bag of Words to compute out-of-vocabulary word vector embeddings. More recently, Flair [ABV18] and ElMo [Pet+18] embeddings try to generalize embeddings to be more context-aware.

On the contrary, instead of relying on feature engineering, fine-tuning methods use the model architecture and parameters as a starting point for specific downstream NLP tasks. BERT [Dev+19] and Pegasus [Zha+20b] are a few examples of pre-trained models that can be fine-tuned on a variety of NLP sub-tasks (e.g., Classification, Sentiment analysis, or Summarization).

Such models capture the deep semantic patterns in the natural language text and handle text ambiguity and variations. Pre-trained language models have also been used on knowledge graphs, where graph triples are converted into sentences using random walks and then used to train language models. For example, DOLORIES [WKW18] initialized graph embeddings models such as TransE with contextualized embeddings generated from entity-relation chains gathered via KG random-walks. Furthermore, Bosselut et al. [Bos+19] use the generative model GPT [Rad+18] to generate tail phrase tokens given the

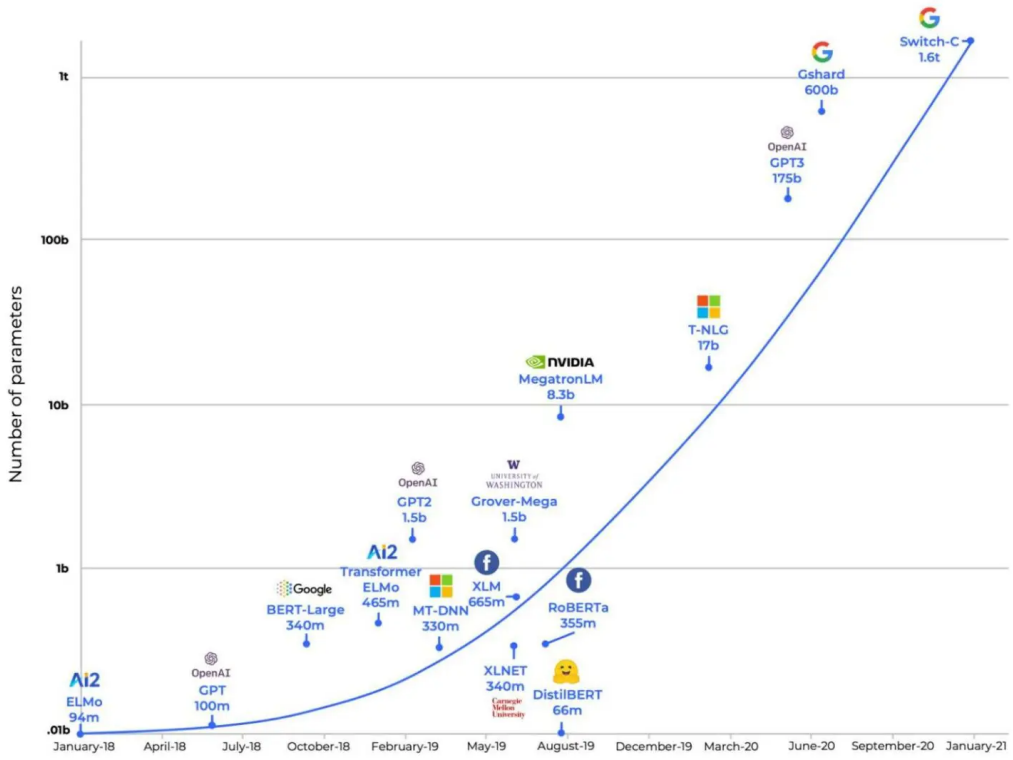


Figure 3.2: **Exponential growth of number of parameters in deep learning models**. Showing the trend of how new models are getting bigger in terms of size (i.e., number of parameters) [Use22].

head tokens and the relation tokens for a KG. Other approaches aim to enhance BERT's representations with entity semantics [Zha+19a]. As such, these approaches concentrate on generating new entities and relations. Finally, KG-BERT [YML19] is a novel approach that utilizes a BERT transformer model to compute plausibility scores of triples based on names or descriptions of entities and relations.

A growing demand for more sophisticated language models, tools, and hardware is being driven by the current requirement for better performance on more complex tasks. As a result, deep learning models have been pushed to generalize better across a variety of tasks. To attain state-of-the-art performance, it has also expanded the size of these models with bulky architectures. As shown Figure 3.2, language models are becoming increasingly complex as the number of factors increases. Although these language models produce outstanding results, they are typically very huge in size, increasing latency and making deployment and scaling challenging.

3.4 Information Extraction

Information Extraction (IE) is a wide term that subsumes numerous approaches and sub-tasks [Gri15]. In this thesis, we focus on two aspects of IE (component-based blind IE procedures, and end-to-end targeted IE processes).

3.4.1 Blind Information Extraction

In the last decade, many open source tools have been released by the research community to tackle IE tasks in the context of KG completion (see Table 5.2). These IE components are not only used for end-to-end KG triple extraction but also for various other tasks, such as:

Text Triple Extraction: The task of open information extraction is a well studied researched task in the NLP community [AJM15]. It relies on NER (Named Entity Recognition) and RE (Relation Extraction). MinIE [GGC17] extracts relation surface forms. While SalIE [PDW18] uses MinIE in combination with PageRank and clustering to find facts in the input text. Furthermore, OpenIE [AJM15] leverages linguistic structures to extract self-contained clauses from the text. For a detailed survey on open information extraction, we point readers to a comprehensive survey by Niklaus et al. [Nik+18].

Another system Graphene [Cet+18] employs two layered transformations of clausal and phrasal embedding to simplify text and extract linguistic triples.

Entity and Relation Linking: Entity and relation linking is a widely studied researched topic in the NLP, Web, and Information Retrieval research communities [Bal18; Bas+21; Del19]. Often, entity and relation linking is performed independently. DBpedia Spotlight [Dai+13] is one of the first approaches for entity recognition and disambiguation over DBpedia [Aue+07]. TagMe [FS10a] links entities to DBpedia using in-link matching to disambiguate candidates entities. Open Tapioca [Del19] uses semantic matching of entity candidates for Wikidata [VK14].

Others tools such as RelMatch [Sin+17] do not perform entity linking and only focus on linking the relation in the text to the corresponding KG relation. Recon [Bas+21] assumes entities are already linked in the text and aims to map relations between the entities to the KG using a graph neural network.

EARL [Dub+18a] is a joint linking tool over DBpedia and models the task as a generalized traveling salesperson problem. Sakor et al. [Sak+19] proposed Falcon, a linguistic rules based tool for joint entity and relation linking over DBpedia. Falcon 2.0 [Sak+20] performs joint entity and relation linking on Wikidata.

Coreference Resolution: This task is used in conjunction with other tasks in NLP pipelines to disambiguate text and resolve syntactic complexities. The Stanford Coreference Resolver [Rag+10] uses a multi pass sieve of deterministic coreference models. Clark and Manning [CM16] use reinforcement learning to fine-tune a neural mention-ranking model for coreference resolution. More recently, Sanh, Wolf, and Ruder [SWR19] introduced a hierarchical model that is capable of multi task learning including coreference resolution.

End-to-End Extraction Systems: More recently, end-to-end systems are gaining more attention due to the boom of deep learning techniques. Such systems draw on the strengths of deep models and transformers [Dev+19; Liu+19a]. Kertkeidkachorn and Ichise [KI17] present an end-to-end system to extract triples and link them to DBpedia. Other attempts such as KG-Bert [YML19] leverage deep transformers (i.e., BERT [Dev+19]) for the triple classification task, given the entity and relation descriptions of a triple. KG-Bert does not attempt end-to-end alignment of KG triples from a given input text. Liu et al. [Liu+18] design an encoder-decoder framework with an attention mechanism to extract and align triples to a KG.

Table 3.4 shows a comparison between some selected NLP components that are discussed here and analyze across multiple dimensions.

Table 3.4: **Comparison of Open Information Extraction components** . A list of various components that can be used for information extraction [Jar22a].

	Task	Method	Relies on	Operates on
Falcon [Sak+20]	Entity and Relation linking	Syntactic rules	Synonyms and string similarity	Wikidata
Spotlight [Dai+13]	Entity linking	Spotting, selection, and disambiguation stages	KG ontology and configuration	DBpedia
EARL [Dub+18a]	Entity and Relation linking	Generalized traveling salesman	KG ontology and label synonyms	DBpedia
SalIE [PDW18]	Triple extraction	Clustering and PageRanking relation surface forms	Fact salience system	Text
ClausIE [DG13]	Triple extraction	Cluster-based relation-arguments extraction	Dependency parsing	Text
NeuralCoref [CM16]	Coreference resoultion	Neural mention ranking	Training data	Text
HMTL [SWR19]	Coreference resoultion	Hierarchical multi-task learning	Training data	Text
RelMatch [Sin+17]	Relation linking	Bipartite graph matching	PATTY	DBpedia

Table 3.5: **Information Extraction systems and their approaches.** Collection of information extraction systems on various domains with details about how they operate and what kind of information do they target [Jar22b].

	Domain	Method	Targets	Operates on
Ji et al. [Ji+20]	Legal	end-to-end sentence classification	Evidence sentences	NL text
TRIE [Zha+20c]	Business	Unified end-to-end text reading and IE network	Form values	Visually rich document
DeepCPCFG [CD21]	Business	RNN learning parsing grammar	Form values	Scanned documents
exBERT [Jar+21b]	Scholarly	Textual triple classification	Knowledge graph triples	KG triples
QAsper [Das+21]	Scholarly	QA on evidence sentences	Targeted values	Full article text
FNG-IE [Tah+21]	Scholarly	Graph-based node ranking	Keywords	NL text
TableSeer [Liu+07]	Digital library	Table detection and ranking with TableRank	Table metadata	Tables
AxCell [Kar+20]	Scholarly	Key element classification	Evaluation metric-results pairs	Latex source

3.4.2 Targeted Information Extraction

Several information extraction methods have been proposed by the community, each with their own advantages and disadvantages. Ji et al. [Ji+20] proposed an end-to-end system that uses a multi-task model to perform sentence classification and information extraction on legal documents. TRIE [Zha+20c] uses end-to-end system to jointly perform document reading and information extraction on everyday documents such as invoices, tickets, and resumes. Chua and Duffy [CD21] proposes a method for finding the suitable grammar set for the parsing and the extraction of information. Specifically for scholarly context, various systems has been created to extract and retrieve information from publications and scholarly articles. exBERT [Jar+21b] uses triple classification to perform knowledge graph completion. Dasigi et al. [Das+21] proposed a method to retrieve information from papers to answer natural language questions. FNG-IE [Tah+21] is an improved graph-based approach for the extraction of keywords from scholarly big-data. Furthermore, Liu et al. [Liu+07] presents the TableSeer system that is capable of metadata extraction from tables of scholarly nature.

With regards to automated text summarization as a tool for information extraction, various language models relying on attention mechanisms [Vas+17] displayed state-of-the-art results superseding human performance. BERT [Dev+19] is one of the most commonly used transformer models capable to automatically summarize text. Similarly, RoBERTa [Liu+19b] is an optimized approach to represent language and is capable of producing summarizations of text. BART [Lew+19] is a sequence-to-sequence model trained as a denoising autoencoder, which improves on the pre-training phase. Zhang et al. [Zha+20b] presents the PEGASUS model trained for abstractive summary generation of text. These and other models are usually built to handle "short" input sequences, e.g. 512-1K tokens. Other attempts address the issue of processing longer inputs. BigBird [Zah+20] and Longformer [BPC20] present models that are capable of handling a much larger input, e.g. 4K-16K tokens. Other generational models aren't created specifically for one task; instead, they are capable of performing multiple tasks depending on the input text. For instance, GPT2 [Rad+19] supports unconditional text generation. Raffel et al. [Raf+19] describe T5, a model that can perform summarization, translation, and question answering based on keywords in the input text. Some of these language models have been either pre-trained on scholarly data such as PubMed[3] and arXiv [Cle+19] datasets, or have been fine-tuned on such data for empirical evaluation in their original publication.

In Table 3.5 we show a collection of information extraction systems compared across the domain they operate on and the methodology employed by them. Though all of these components perform the task of information extractions, each of them addresses the issue differently and targets various types of information elements. Our contribution in Chapter 6 leverages the capabilities of automatic summarization for the objective of information extraction form scholarly documents, and can be seen along the same lines as QAsper system in this table.

[3] https://www.nlm.nih.gov/databases/download/pubmed_medline.html

CHAPTER 4

Knowledge Graph Infrastructure for Scholarly Contributions

Documents are central to scholarly communication. In fact, nowadays almost all research findings are communicated by means of digital scholarly articles. However, it is difficult to automatically process scholarly knowledge communicated in this form. The content of articles can be indexed and exploited for search and mining, but knowledge represented in form of text, images, diagrams, tables, or mathematical formulas cannot be easily processed and analysed automatically. The primary machine-supported tasks are largely limited to classic information retrieval. Current scholarly knowledge curation, publishing and processing does not exploit modern information systems and technologies to their full potential [Har01]. As a consequence, the global scholarly knowledge base continues to be little more than a distributed digital document repository.

The key issue is that digital scholarly articles are mere analogues of their print relatives. In the words of Van de Sompel and Lagoze [SL09] dating back to 2009 and earlier: "The current scholarly communication system is nothing but a scanned copy of the paper-based system." A further decade has gone by and this observation continues to be true. Print and digital media suffer from similar challenges. However, given the dramatic increase in output volume and travel velocity of modern research [BM15a] as well as the advancements in information technologies achieved over the past decades, the challenges are more obvious and urgent today than at any time in the past.

Scholarly knowledge remains as ambiguous and difficult to reproduce [Bos+] in digital as it used to be in print. Moreover, addressing modern societal problems relies on interdisciplinary research. Answers to such problems are debated in scholarly discourse spanning often dozens and sometimes hundreds of articles [AN14]. While citation does link articles, their contents are hardly interlinked and generally not machine actionable. Therefore, processing scholarly knowledge remains a manual, and tedious task. Furthermore, document-based scholarly communication stands in stark contrast to the digital transformation seen in recent years in other information rich publishing and communication services. Examples include encyclopedia, mail order catalogs, street maps or phone books. For these services, traditional document-based publication was not just digitized but has seen the development of completely new means of information organization and access. The striking difference with scholarly communication is that these digital services are not mere digital versions of their print analogues but entirely novel approaches to information organization, access, sharing, collaborative generation.

There is an urgent need for a more flexible, fine-grained, context sensitive and machine actionable representation of scholarly knowledge and corresponding infrastructure for knowledge curation, publishing and processing [Gro19; Ten+19]. We suggest that representing scholarly knowledge as structured, interlinked, and semantically rich knowledge graphs is a key element of a technical infrastructure [Aue+18]. While the technology for representing, managing and processing scholarly knowledge in such form is largely in place, we argue that one of the most pressing concerns is how scholarly knowledge can be acquired as it is generated along the research lifecycle, primarily during the *Conducting* step [Vau+13].

In this chapter, we introduce the Open Research Knowledge Graph (ORKG)[1] as an infrastructure for the acquisition, curation, publication and processing of semantic scholarly knowledge. The content of this chapter is based the initial publications [Jar+19b; Jar+19a], as well as these publications [Oel+21; Aue+20; Oel+20] that reflect current changes to the ORKG. The result of this chapter aims to answer the following research question, research challenge at the level of semantic scholarly infrastructure:

> RQ1: What are the requirements and possible implementation options for a semantic scholarly knowledge graph representing scientific contributions?
>
> RC1: Representing, Modeling, and Querying Semantic Annotations of Scholarly Knowledge.

To address this research question, we present, evaluate and discuss ORKG based scholarly knowledge acquisition using crowdsourcing and text mining techniques as well as knowledge curation, publication and processing. The contributions of this chapter can be summarized as follows:

- Bases of a semantic scholarly infrastructure - The Open Research Knowledge Graph (ORKG) - capable of representing, storing, and manipulating structured scholarly communication with crowdsourcing and automated techniques.
- Description of universal data model for data representation in the ORKG, and the formalization of two main features of the ORKG "SOTA Comparison", and "Contribution Similarity".
- Assessment of avenues of scholarly knowledge acquisition, both manual and automated.
- User study on the user interface, to check the willingness of users to contribute structured descriptions and their overall acceptance of the proposed method, as well empirical evaluation of some automated aspects of the infrastructure.

This chapter is structured as it follows: Section 4.1 describes the basic research problem we are trying to tackle here. In Section 4.2, we outline how the ORKG operates, its architecture, features, knowledge acquisition methods, and its implementation. Section 4.3 presents the evaluation results of both the user study and the empirical evaluation. Finally Section 4.4 puts forward discussion points about the evaluation results and the overall contribution and summarizes the content of this chapter.

[1] The contribution described in this chapter is based on the work of the entire ORKG team. Our contribution is mainly in the backend development of the service to enable storing, modeling, and ingestion of the scholarly data. As well as the foundation of automated approaches for automated contribution data acquisition.

4.1 Problem Statement

We illustrate the problem with an example from life sciences. When searching for publications on the popular Genome editing method CRISPR/Cas in scholarly search engines we obtain a vast amount of search results. Google Scholar, for example, currently returns more than 50,000 results, when searching for the search string "CRISPR Cas". Furthermore, research questions often require complex queries relating numerous concepts. Examples for CRISPR/Cas include: How good is CRISPR/Cas w.r.t. precision, safety, cost? How is genome editing applied to insects? Who has applied CRISPR/Cas to butterflies? Even if we include the word "butterfly" to the search query we still obtain more than 600 results, many of which are not relevant. Furthermore, relevant results might not be included (e.g., due to the fact that the technical term for butterfly is Lepidoptera, which combined with "CRISPR Cas" returns over 1000 results). Finally, virtually nothing about the returned scholarly knowledge in the returned documents is machine actionable: human experts are required to further process the results.

We argue that keyword-based information retrieval and document-based results no longer adequately meet the requirements of research communities in modern scholarly knowledge infrastructures [Edw+13] and processing of scholarly knowledge in the digital age. Furthermore, we suggest that automated techniques to identify concepts, relations and instances in text [Sin+18], despite decades of research, do not and unlikely will reach a sufficiently high granularity and accuracy for useful applications. Automated techniques applied on published legacy documents need to be complemented with techniques that acquire scholarly knowledge in machine actionable form as knowledge is generated along the research lifecycle. As Mons suggested [Mon05], we may fundamentally question "Why bury [information in plain text] first and then mine it again". Hence, we break the original research question **RQ1** into two sub-questions and we study each of them:

- RQ1.1: Are authors willing to contribute structured descriptions of the key research contribution(s) published in their articles using a fit-for-purpose infrastructure, and what is the user acceptance of the infrastructure?
- RQ1.2: Can the infrastructure effectively integrate crowdsourcing and automated techniques for multi-modal scholarly knowledge acquisition?

4.2 Open Research Knowledge Graph

We propose to leverage knowledge graphs to represent scholarly knowledge communicated in the literature. We call this knowledge graph the Open Research Knowledge Graph (ORKG). Crucially, the proposed knowledge graph does not merely contain (bibliographic) metadata (e.g., about articles, authors, institutions) but semantic (i.e., machine actionable) descriptions of scholarly knowledge.

4.2.1 Architecture

The infrastructure design follows a classical layered architecture. As depicted in Figure 4.1, a persistence layer abstracts data storage implemented by labeled property graph (LPG), triple store, and relational database storage technology, each serving specific purposes. Versioning and provenance handles tracking changes to stored data.

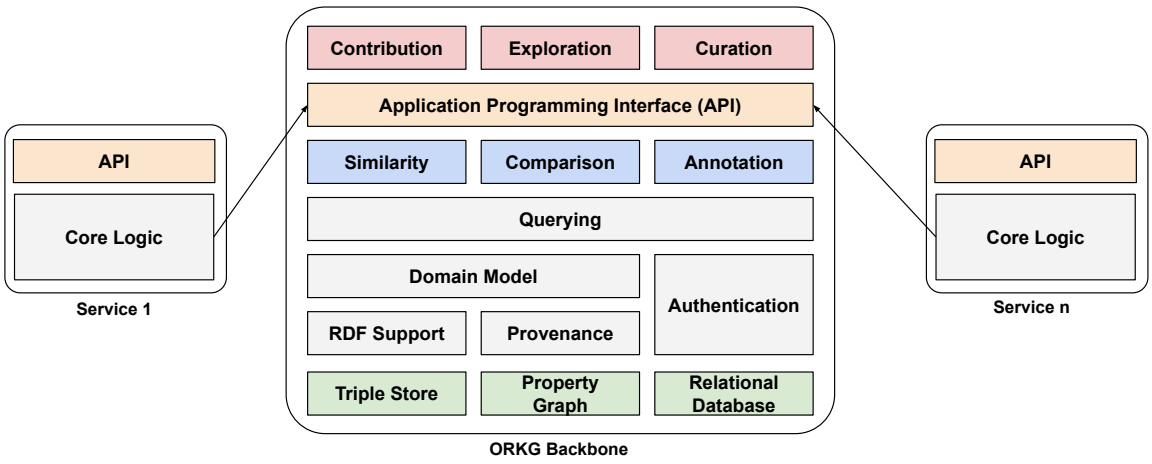


Figure 4.1: **ORKG layered architecture and micro service overview.** The base of the architecture is the ORKG backbone. The green components ○ indicate the persistence layer. ○ components describe higher level application that are still part of the backend but aggregate information from lower levels. ○ is the API connector that connects other microservices with the backbone and the graph. Frontend components ○ run on top of the API and leverage other services APIs as well.

The domain model specifies `ResearchContribution`, the core information object of the ORKG. A `ResearchContribution` relates the `ResearchProblem` addressed by the contribution with the `ResearchMethod` and (at least one) `ResearchResult`. Users can adopt arbitrary third-party vocabularies to describe problems, methods, and results. For instance, users could use the ontology for Biomedical Investigations as a vocabulary to describe statistical hypothesis tests.

Research contributions are represented by means of a graph data model. Similarly to the Research Description Framework [BG14] (RDF), the data model is thus centered around the concept of a statement, a triple consisting of two nodes (resources) connected by a directed edge. In contrast to RDF, the data model allows to uniquely identify instances of edges and to qualify these instances (i.e., annotate edges and statements). As metadata of statements, provenance information is a concrete and relevant application of such annotation (see Figure 4.2).

RDF import and export enables data synchronization between LPG and triple store, which enables SPARQL and reasoning. Querying handles the requests by services for reading, updating, and creating content in databases. The following layer is for modules that implement infrastructure features such as authentication or comparison and similarity computation. The REST API acts as the connector between features and services for scholarly knowledge contribution, curation and exploration.

ORKG users in author, researcher, reviewer or curator roles interact differently with its services. Exploration services such as State-of-the-Art comparisons are useful in particular for researchers and reviewers. Contribution services are primarily for authors who intend to contribute content. Curation services are designed for domain specialists more broadly to include for instance subject librarians who support quality control, enrichment and other content organization activities. The modular structure of the system allows to plug in multiple micro services, each performs their own custom functionality such as, GraphQL integration [Har+21] or Faceted search [Hei+21], each with their own API that exposes the functionality to the user interface and to any other client.

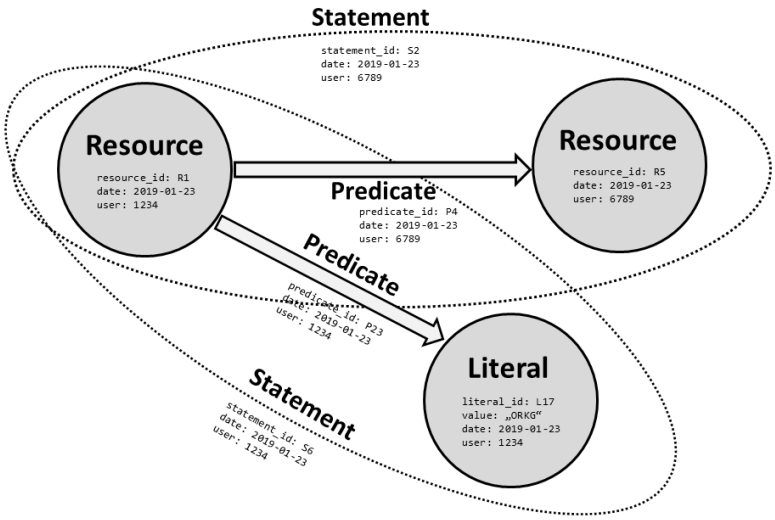


Figure 4.2: **Simplified view of the ORKG domain model.** Schematic representation of the domain model. All entities can be identified by their ID and are stored as nodes in the graph. Connected nodes form statements.

4.2.2 Features

The ORKG services are underpinned by numerous features that, individually or in combination, enable services. We present the most important current features next.

State-of-the-Art (SOTA) Comparison. SOTA comparison extracts similar information shared by user selected research contributions and presents comparisons in tabular form. Such comparisons rely on extracting the set of semantically similar predicates among compared contributions.

We use FastText [Boj+17] word embeddings in combination with Fuzzy string similarity to generate a similarity matrix γ

$$\gamma = \left[cos(\overrightarrow{p_i}, \overrightarrow{p_j}) \oplus \xi(L^{p_i}, L^{p_j}) \right] \tag{4.1}$$

with the cosine similarity of vector embeddings for predicate pairs $(p_i, p_j) \in \mathscr{R}$, whereby $\mathscr{R}$ is the set of all research contributions. The operator $\oplus$ is a combination operator for the similarities, ξ is the fuzzy string similarity function, and L^p is the label string of a predicate p.

Furthermore, we create a mask matrix $\boldsymbol{\Phi}$ that selects predicates of contributions $c_i \in \mathscr{C}$, whereby $\mathscr{C}$ is the set of research contributions to be compared. Formally,

$$\boldsymbol{\Phi}_{i,j} = \begin{cases} 1 & \text{if } p_j \in c_i \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

Next, for each selected predicate p we create the matrix φ that slices $\boldsymbol{\Phi}$ to include only similar predicates. Formally,

$$\varphi_{i,j} = (\Phi_{i,j})_{\substack{c_i \in \mathscr{C} \\ p_j \in sim(p)}} \tag{4.3}$$

where $sim(p)$ is the set of predicates with similarity values $\gamma[p] \geq T = 0.9$ with predicate p. The threshold T is computed empirically. Finally, φ is used to efficiently compute the common set of predicates and their frequency.

This method is also extend to merge not only on the predicate p rather to locate and collect the set of paths $P_d = \{r_1, p_1, ..., r_w, p_w\}$, where d refers to a resource in the graph (usually a leaf node), r & p refer to resources and predicates linked to each other in the subgraph leading to the resource d, and w is the length of the path (i.e., the number of hops from a contribution c to a resource d.

Contribution Similarity. Contribution similarity is a feature used to explore related work, find or recommend comparable research contributions. The sub-graphs $G(r_i)$ for each research contribution $r_i \in \mathscr{R}$ are converted into document D by concatenating the labels of subject s, predicate p, and object o, of all statements $(s, p, o) \in G(r_i)$. We then use $BM25$ [Ram+03] to index and retrieve the most similar contributions with respect to some query q. Queries are constructed in the same manner as documents D.

The way the document is index plays a crucial role in how the contribution similarity service finds similar contributions (and with extension papers). With the work of [Ara22], the representation of the document is analyzed and shown that the most suitable textual representation of the structured annotations in the indexing engine should follow this format:

```
Contribution 1
    has research problem
        Ambiguity in QA
    has approach
        corpus-based disambiguation
            has methodology
                rely on counts
```

Excerpt 4.1: **Indexed document sample for contribution similarity.** Indentation in the text describes the hierarchy between concepts. The description alternates resource followed by property until the end of the contribution sub-graph.

Automated Information Extraction. The ORKG uses machine learning for automated extraction of scientific knowledge from literature. Of particular interest are the NLP tasks named entity recognition as well as named entity classification and linking.

As a first step, we trained a neural network based machine learning model for named entity recognition using in-house developed annotations on the Elsevier Labs corpus of Science, Technology, and Medicine[2] (STM) for the following generic concepts: process, method, material and data. We use the Beltagy, Cohan, and Lo [BCL19] Named Entity Recognition task-specific neural architecture atop pretrained SciBERT embeddings with a CRF-based sequence tag decoder [MH16].

[2] https://github.com/elsevierlabs/OA-STM-Corpus

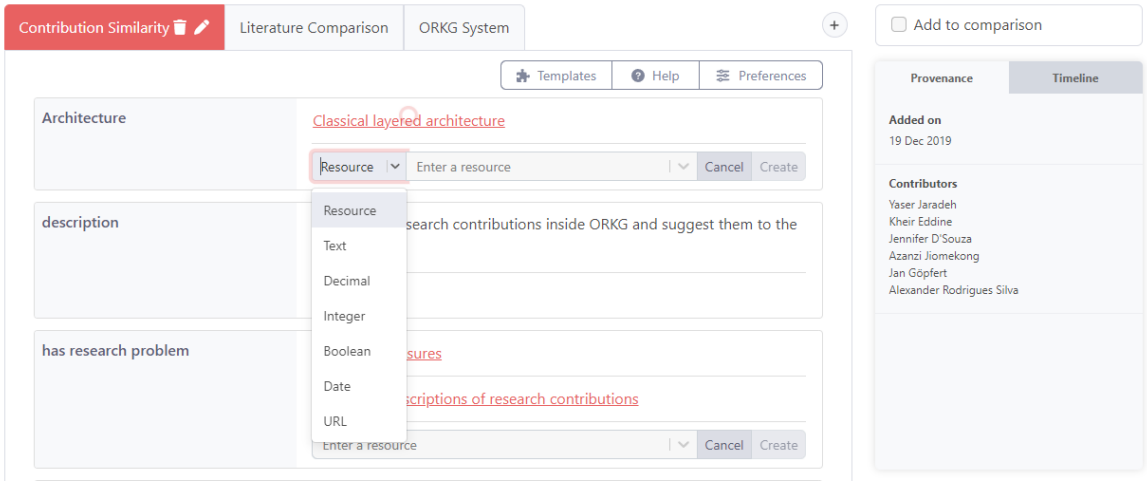


Figure 4.3: **Paper wizard** in the frontend of the ORKG. The wizard allows for navigation of data, creation of new statements, and updating existing ones.

Linking scientific knowledge to existing knowledge graphs including those from the open domain such as DBpedia [Aue+07] as well as domain specific graphs such as ULMS [Bod04] is another important feature. Most importantly, such linking enables semi-automated enrichment of research contributions.

4.2.3 Knowledge Acquisition Methods

The ORKG developed multiple avenues of knowledge acquisition, some focus on on visualizing the information and searching it, other methods relies on crowdsourcing as the main input dimensions, and some employ automated techniques to import at scale large amounts of knowledge with the support of NLP techniques. Here we give an overview of these approach and describe to which are the intended stakeholders of the approach, to which group it belongs, and what it is used for.

Paper Wizard: The most user-friendly method of data entry, and manipulation. From its name, the wizard guides the user step by step from adding a paper (the central concept of the ORKG), then its meta data, classifies it into a research area, and then adds the semantically annotated statements. Though this method is easy to use, it does not provide any sort of guidance in terms of what structure to use or which properties to link to. Making the intended stakeholder group to be expert researchers and curators that know how to model the data and what external anthologies to link to. The wizard abstracts away technical details that lowers the data entry threshold for users, such as typing, Provence assertion, and data validation. Figure 4.3 shows a snapshot of how the paper wizard looks like.

We note that a user can explore, create, and edit information about a paper from using this approach. As a means to assist users to fill in data correctly, auto complete functionalities and intelligent lookup components guide the user when creating new or when linking to existing resource. Furthermore, "Templates" which essentially are schemas with a more-friendly name, constrains certain input mechanisms which forces the entry process to follow a procedure or more formally ensures that the newly created data with this method adhere to certain structure and uses pre-defined set of types.

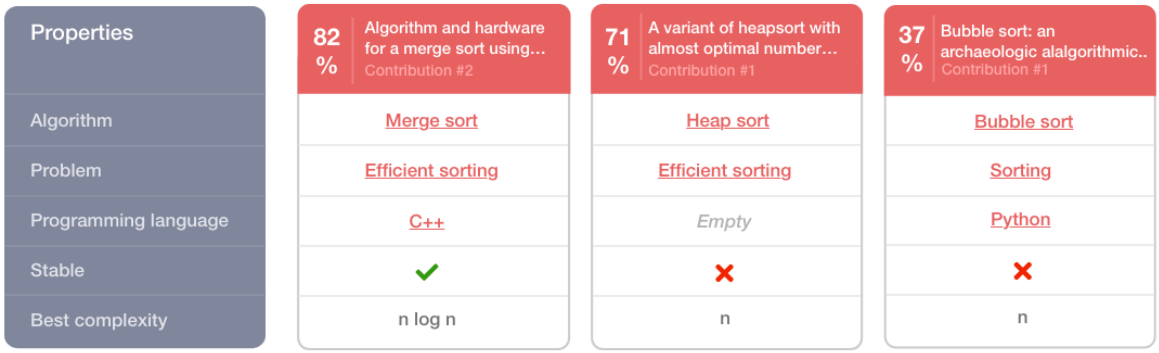


Figure 4.4: **Contribution comparison explorer.** A tabular view comparison multiple papers (contributions) with each other, generated automatically by the ORKG from the subgraphs of the individual contributions.

Comparison explorer: Creating state-of-the-art comparison dynamically is one of the main features and data exploration methods of the ORKG. The comparison (as described in Subsection 4.2.2) seeks to find the connection points in between contributions automatically. To do, it would be need to expand the subgraph of each of the contributions meant to be compared, and perform complex operations to find the joints to connect multiple distinct subgraphs together. The user of this approach has no saying in how the comparison tables are created, due to their dependence on the modeling of the scholarly contributions. The comparison explorer is intended for users that want to get a quick overview of multiple papers or even a research problem without going into the individual details of each publication. Consumption of the comparison does not require any technical or domain experience due to the simplicity of the tabular representation and the existence of tool-tips that explain any foreign notation.

Figure 4.4 depicts a view of the tabular comparison. Each column in the comparison represent a contribution in the ORKG, and each row represent a property and their corresponding value in each contribution subgraph. The comparison merges similar properties with each other. For instance, "problem" and "addresses" can be merged under one property because they both mean the same thing in the context of the comparison. As described by [Oel+20], the comparison can be then exported into a RDF, CSV, or even LaTeXrepresentation for wider interoperability and usability.

Contribution editor: Similarly to the comparison explorer, the contribution editor provides an easy method of creating comparisons directly by abstracting the way of creating individual papers that share the same structure hence preserving an envisioned comparison view and forcing the comparison creation to create a pre-defined view of the comparison. The contribution editor targets users that do not have much modeling experience and their main goal is to add papers into the ORKG that end up in a comparison.

Graph visualization: Having means to condense information in a visual way always lower consumption threshold for various users. Various visualization techniques are integrated into the ORKG ecosystem enable the easy exploration and the representation of information in visual manner. Most importantly, since the proposed scholarly graph revolves around the concept of a research paper and research contribution. One can visualize this information as a set of interconnected nodes and edges

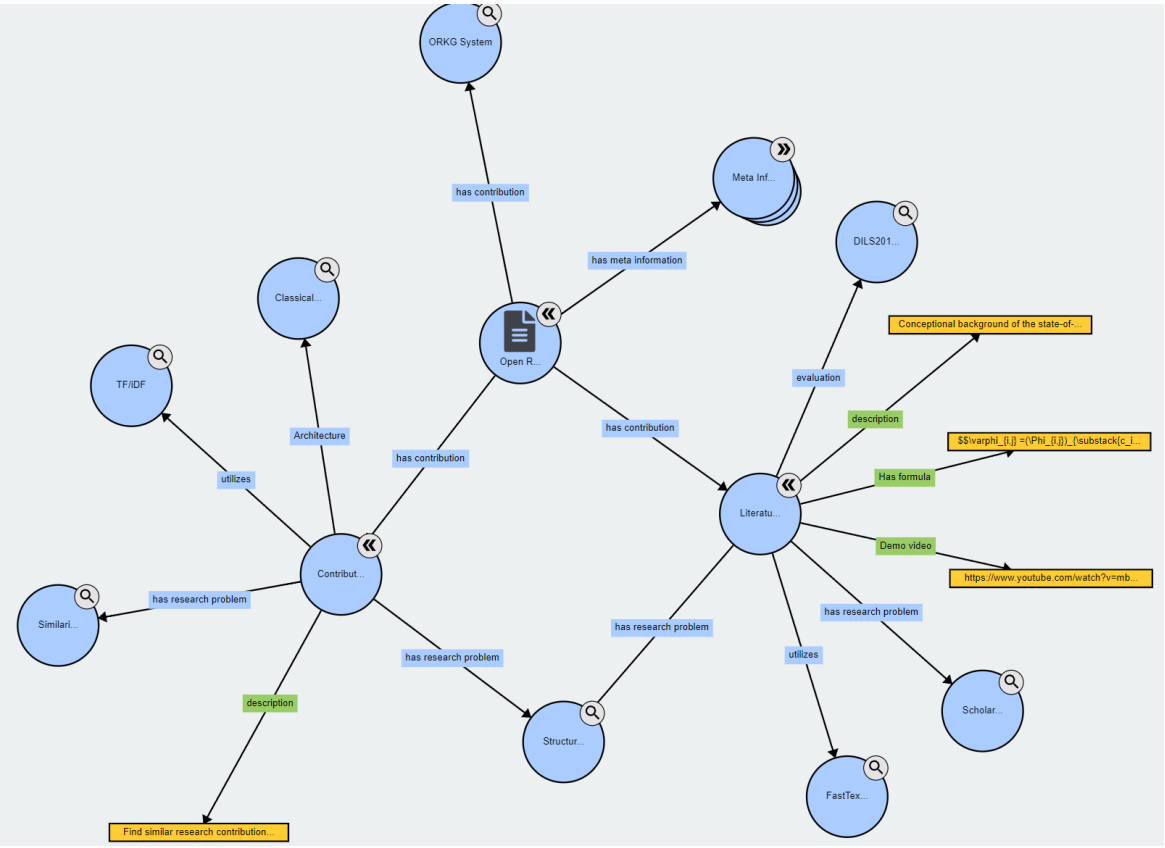


Figure 4.5: **Graph view visualization of a structured paper descriptions in the ORKG.** A visualization of the underlying graph representation of a subset of statements from a single paper. Oval nodes indicate resource, and yellow rectangles indicate literal values.

(i.e., a directed graph). Figure 4.5 depicts the visualization of a subgraph from one paper description in the ORKG. The visualization shows the resources and the properties interconnected with each other, and each node is interactive (i.e., can be further navigated and explored). The graph view enable users to get a better understanding of what is a knowledge graph without knowing any technical jargons or terminology, as well as providing the ability to search through the graph view, finding certain nodes, and then going to their individual structured description for further manipulation. Graph visualization falls into the exploratory method category, and does not need any technical background and targets the least knowledgeable class of users of the ORKG.

Abstract annotator: All the tools mentioned previously put the user in the center of the workflow, unlike the abstract annotator that supports the user by extracting concepts and entities from the abstract of the input articles and suggesting them to be added under specific relations to the newly created contribution. Integrated into the input flow of the paper wizard, the abstract annotator relies on pre-trained scholarly deep language models on the STEM corpus [DS0+20] to recognize and classify entities into four main categories (or classes), which can be overwritten by users.

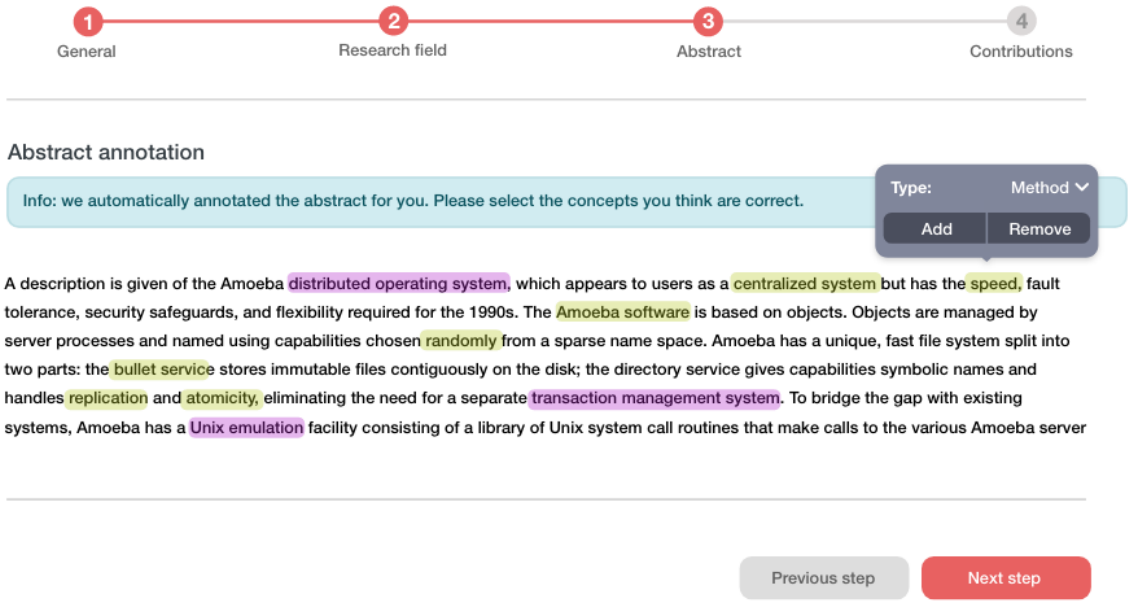


Figure 4.6: **Abstract annotator**: automated information extraction feature integrated into the research contribution workflow of the infrastructure.

Figure 4.6 illustrates how the abstract annotator integrates within the paper entry journey. The interface provides a certainty threshold that is used to limit more debatable choices of the model, which in turn helps the user with the concept selection and the data injection into the paper. The annotator tries to fetch the abstract automatically from available sources if permanent identifiers are induced when adding the paper into the ORKG, if not the user still has the opportunity to add the abstract themselves. This method of data acquisition targets the same stakeholders as the paper wizard due to its tight integration in the same workflow as the wizard.

CSV importer: Another method of data acquisition is to leverage existing data in legacy formats that researchers and non-researchers use. CSV (comma separated values) are one of the most common formats due to its compact representation and cross-interoperability. The CSV importer tool leverages this particular aspect. If a CSV data is represented in a similar manner to how an ORKG looks like - each row represents a property and their values, and each column is a contribution in a paper - then the CSV importer can in a single swoop import the bulk of the papers and creates counterpart resources for each value and reuses them if the need arises. The importance of the CSV imports is the ease at which any researcher can use and the backward compatibility with existing legacy research findings that is stored in CSV or CSV-compatible representations.

Benchmarks: Since the ORKG is not the only existing source of structured annotation of scholarly nature. The ORKG collects semi-structured data from other sources (such as the PwC[3]) and semantically annotates the data and exposes custom views on it. Benchmarks are an aggregation view on

[3] https://paperswithcode.com/

Figure 4.7: **Benchmarks in the ORKG.** In this examples, a benchmark of text summarization models on the "X-Sum" dataset, showing the performance trend of the Rouge-2 metric for a set of research papers.

evaluation results of machine learning papers. Each paper used in the benchmarks describes a machine learning model running a natural language processing task against other baselines and models on various datasets. The benchmark view query the structured imported data and bundles them under common datasets and evaluation metrics.

Figure 4.7 shows how the benchmarks are viewed from a particular dataset in the ORKG. The benchmark is unique across a research field, what research problem it describes, what are the datasets being evaluated, on what machine learning models, and using which evaluation metrics. Informative trends are depicted as line charts or other suitable interactive graphical representations, to show how different models (and inherently different papers) are performing on a specific dataset in a certain research problem that is being addressed in a particular research area.

Down-stream data science: The scientific method requires testing and analysis of hypotheses and results. This takes place down stream where individual researchers validates their assumptions and look at raw data. The ORKG recognizes that leveraging this step is crucial for knowledge acquisition and requires supporting. Thus, we provide a set of tools (i.e., APIs, libraries, packages, and interfaces) to make it easier for researchers to bring their findings directly into the graph without leaving their preferred working environment. For instance, a python package[4] is provided to be used with various data science scenarios and usecases.

Figure 4.8 depicts how easy it is to bring all the raw functionalities of the ORKG into a researchers workspace. Users can use the ORKG during their analysis, data curation, or even evaluation. The ORKG can be used as the source of the data or the final destination of certain conclusions made from the analysis which can be added into the graph using a few simple lines of code. This method

[4] https://orkg.readthedocs.io/en/latest/

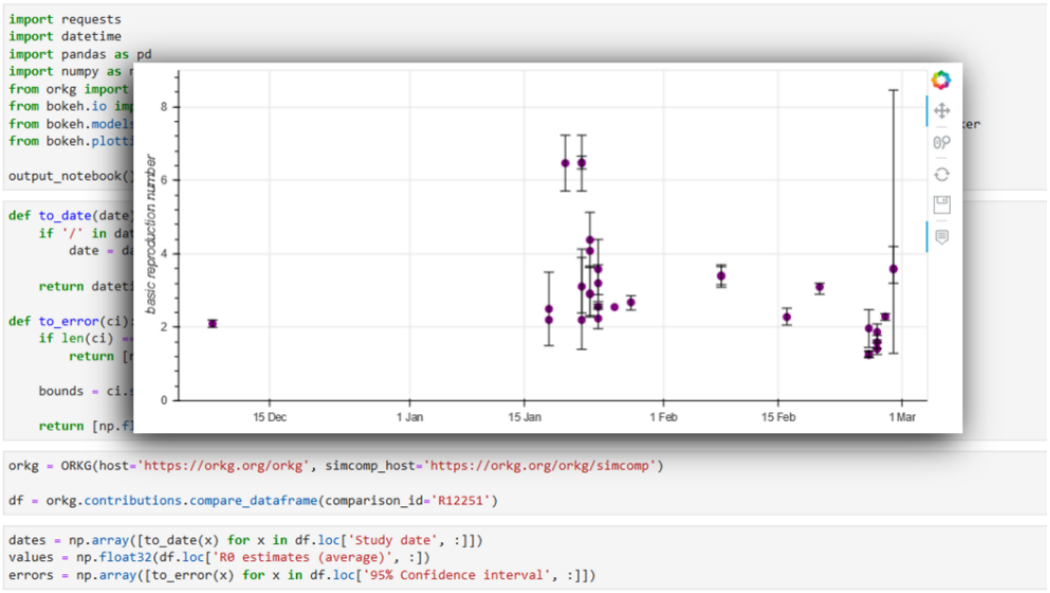


Figure 4.8: **Down stream application integration with the ORKG.** A sample of a python notebook creating scatter plots of data collected from the ORKG, and can be then procced and inserted back into the graph

of interaction with the ecosystem requires some technical background to understand the technical constraints and the interfaces that can be used to implement a certain functionality. So the targeted group of this approach is technical researchers and research engineers, though such tools are created with ease of access in mind.

4.2.4 Implementation

The infrastructure consists mainly of three building blocks: the *backend* with the business logic, system features and a set of APIs used by services and third party apps; the *frontend*, i.e., the User Interface (UI); and the *peripheral services* which covers all supporting and additional functionality that merits creating a seperate micro service for.

Backend: The back end is written in Kotlin [Jet11], within the Spring Boot framework. The data is stored in a Neo4j Labeled Property Graph (LPG) database accessed via the Spring Data Neo4j's Object Graph Mapper (OGM). Data can be queried using Cypher, Neo4j's native query language (shares similarities with SQL and SPARQL). The backend is exposed via a JSON RESTful API accessed by applications, including the ORKG frontend and the other micro services. A technical documentation of the current API specification is available online[5].

Data can be exported to RDF via custom mappings following the Neo4j Semantics extension[6] notations. Due to the differences between our graph model and RDF, a "semantification" step needs to occur always befor import or export. Most importantly, the ORKG backend auto-generates URIs to all

[5] https://tibhannover.gitlab.io/orkg/orkg-backend
[6] https://github.com/jbarrasa/neosemantics

the graphs entities. Mapping (or changing) these URIs to an existing ontology must be done manually. The Semantics extension also allows importing RDF data and ontologies. The Web Ontology Language is, however, not fully supported.

In order to enrich ORKG data, we support linking to data from other external sources. Of particular interest are systems that collect, curate and publish bibliographic metadata or data about entities relevant to scholarly communication, such as people and organizations. Rather than collecting such metadata ourselves we thus link and integrate with relevant systems (e.g., DataCite, Crossref) and their data via identifiers such as DOI, ORCID or ISNI.

Frontend: The ORKG frontend[7] is a Web-based tool for, among other users, researchers and librarians and supports searching, exploring, creating and modifying research contributions. The frontend is implemented according to the ES6 standard of JavaScript using the React framework[8] and relies on client side rendering. The Bootstrap[9] framework is used for responsive interface components. The user interface is the main aggregator of information from the backend, other micro-services, and other metadata repositories. All the knowledge acquisition methods discussed in this chapter and more are implemented in the frontend to make it easier for wide range of users to get familiar with the ecosystem and start applying it in associated research areas.

The frontend was built with the following two key requirements in mind: (1) *Usability* to enable a broad range of users, in particular researchers across disciplines, to contribute, curate and explore research contributions; and (2) *Flexibility* to enable maximum degree of freedom in describing research contributions. The design of the frontend takes great care to deliver the best possible overall experience for a broad range of users, in particular researchers not familiar with knowledge graph technology. User evaluations are a key instrument to continually validate the development and understand requirements.

Peripheral Services: Due to the dynamic and experimental nature of tools and approaches that need to be tried and evaluated to be become a part of the ORKG. Other peripheral services that perform specific actions like NLP processing pipelines, data import, GraphQL integration, or particular usecases are created as their own services with their own separate APIs. These services are usually written in python programming language[10], and would make use of the API of the backend directly or indirectly (through packages). Different services requires different requirements which vary from specialized databases to indexing services. At the time of writing this chapter, the ORKG ecosystem contains more than 20+ micro-services that are working in harmony to elevate structured scholarly description to the next level.

[7] http://orkg.org
[8] https://reactjs.org/
[9] https://getbootstrap.com
[10] https://www.python.org/
[11] Meaning that the participant added information following some organization scheme, including distinct and possibly further refined problem, method and result.

Table 4.1: **Overview of answers** to the key aspects covered by the evaluation questionnaire and other metrics recorded during the interviews (See Appendix B)

Participant Nr	Navigation	Terminology	Auto Complete	Guidance Needed	Suggest To Others	UI Likeness	Time	Number of Triples	Properly Structured[11]
	5 = Very intuitive	5 = Easy to understand	5 = Very helpful	5 = All the time	9 = Very likely	9 = Very much	in mins		
1	4	4	5	3	2	66	16	56	✓
2	2	3	5	4	8	7	19	35	✗
3	4	5	5	3	9	7	15	81	✓
4	3	3	5	3	6	7	13	27	✗
5	4	3	5	3	6	8	14	48	✗
6	4	3	5	3	8	9	13	23	✗
7	3	4	5	3	7	6	19	57	✓
8	3	2	4	3	8	6	13	56	✓
9	4	5	3	3	7	5	14	55	✗
10	4	5	5	1	8	8	22	68	✓
11	4	5	5	1	8	8	20	55	✓
12	-	-	-	-	-	-	21	71	✗
Average	**4**	**4**	**5**	**3**	**7**	**7**	**17**	**53**	-

4.3 System Analysis

The ORKG infrastructure, its services, features, performance and usability are continually evaluated to inform the next iteration and future developments. Among other preliminary evaluations and results, we present here the *first* frontend user evaluation, which informed the second iteration of frontend development, presented in this chapter.

4.3.1 Early User Study

Following a qualitative approach, the evaluation of the first iteration of frontend development aimed to determine user performance, identify major (positive and negative) aspects, and user acceptance/perception of the system. The evaluation process had two components: (1) instructed interaction sessions and (2) a short evaluation questionnaire. This evaluation resulted in data relevant to our first research question.

We conducted instructed interaction sessions with 12 authors of articles presented at the DILS2018[12] conference. The aim of these sessions was to get first-hand observations and feedback. The sessions were conducted with the support of two instructors. At the start of each session, the instructor briefly explained the underlying principles of the infrastructure, including how it works and what is required from authors to complete the task, i.e., create a structured description of the key research contribution in their article presented at the conference. Then, participants engaged with the system without further guidance from the instructor. However, at any time they could ask the instructor for assistance. For each participant, we recorded the time required to complete the task (to determine the mean duration of a session), the instructor's notes and the participant's comments.

In addition to the instructed interaction sessions, participants were invited to complete a short evaluation questionnaire (see Appendix B). The questionnaire is available online[13]. Its aim was to collect further insights into user experience. Since the quantity of collected data was insufficient to

[12] https://www.springer.com/us/book/9783030060152
[13] https://doi.org/10.5281/zenodo.2549918

Table 4.2: **Time (in seconds) needed to perform comparisons** with 2-8 research contributions using the baseline and ORKG approaches.

	Number of compared research contributions						
	2	**3**	**4**	**5**	**6**	**7**	**8**
Baseline	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

establish any correlational or causal relationship [Jan10], the questionnaire was treated as a qualitative instrument. The paper-based questionnaire consisted of 11 questions. These were designed to capture participant thoughts regarding the positive and negative aspects of the system following their instructed interaction session. Participants completed their questionnaire after the instructed interaction session. All 12 participants answered the questionnaire. The interaction notes, participant comments and the time recordings were collected together with questionnaire responses and analysed in light of our research questions.

A dataset summarizing the research contributions collected in the experiment is available online[14]. The data is grouped into four main categories. *Research Problem* describes the main question or issue addressed by the research contribution. Participants used a variety of properties to describe the problem, e.g., problem, addresses, subject, proposes and topic. *Approach* describes the solution taken by the authors. Properties used included approach, uses, prospective work, method, focus and algorithm. *Implementation & Evaluation* were the most comprehensively described aspects, arguably because it was easier for participants to describe technical details compared to describing the problem or the approach.

In summary, 75% of the participants found the frontend developed in the first iteration fairly intuitive and easy to use. Among the participants, 80% needed guidance only at the beginning while 10% did not need guidance. The time required to complete the task was 17 minutes on average, with a minimum of 13 minutes and a maximum of 22 minutes. Five out of twelve participants suggested to make the frontend more keyboard-friendly to ease the creation of research contributions. As participant #3 stated: "More description in advance can be helpful." Two participants commented that the navigation process throughout the system is complicated for first-time users and suggested alternative approaches. As an example, participant #5 suggested to "Use breadcrumbs to navigate."

Four participants wished for a visualization (i.e., , graph chart) to be available when creating new research contributions. For instance, participant #1 commented that "It could be helpful to show a local view of the graph while editing." This type of visualization could facilitate comprehension and ease curation. Another participant suggested to integrate a document (PDF) viewer and highlight relevant passages or phrases. Participant #4 noted that "If I could highlight the passages directly in the paper and add predicates there, it would be more intuitive and save time."

Further details of the questionnaire, including participant ratings on main issues, are summarized in Table 4.1. While the cohort of participants was too small for statistically significant conclusions, these results provided a number of important suggestions that informed the second iteration of frontend development, which is presented in this chapter and will be evaluated in the future.

[14] https://doi.org/10.5281/zenodo.3340954

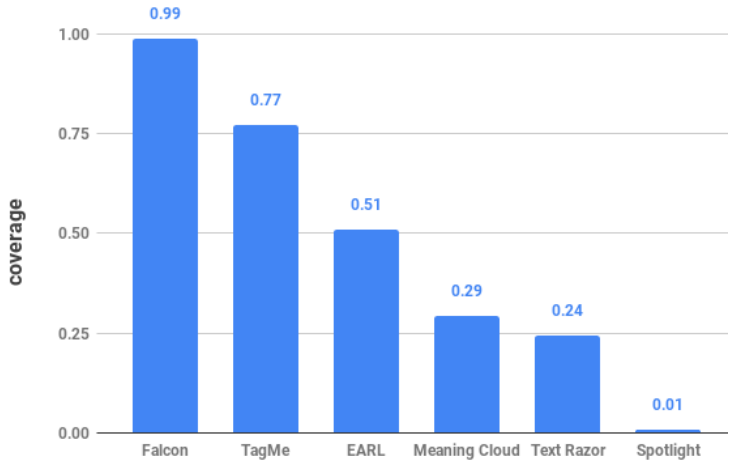


Figure 4.9: **Coverage values of different NED systems** over the annotated entities of the STM corpus.

4.3.2 Empirical Analysis

We have performed preliminary evaluations also of other components of the ORKG infrastructure. The experimental setup for these evaluations is an Ubuntu 18.04 machine with Intel Xeon CPUs 12 × 3.60 GHz and 64 GB memory.

Comparison feature. With respect to the State-of-the-Art comparison feature, we compared our approach in ORKG with the baseline approach, which uses brute force to find the most similar predicates and thus checks every possible predicate combination. Table 4.1 shows the time needed to perform the comparison for the baseline approach and for the approach we implemented and presented above. As the results suggest, our approach clearly outperforms the baseline and the performance gain can be attributed to more efficient retrieval. The experiment is limited to 8 contributions because the baseline approach does not scale to larger sets.

Scalability. For horizontal scalability, the infrastructure containerizes applications. We also tested the vertical scalability in terms of response time. For this, we created a synthetic dataset of papers. Each paper includes one research contribution described by three statements. The generated dataset contains 10 million papers or 100 million nodes. We tested the system with variable numbers of papers and the average response time to fetch a single paper with its related research contribution is 60 ms. This suggests that the infrastructure can handle large amounts of scholarly knowledge.

NED performance. We evaluated the performance of a number of existing NED tools on scholarly knowledge, specifically Falcon [Sak+19], DBpedia Spotlight [MRN14], TagME [FS10b], EARL [Dub+18b], TextRazor[15] and MeaningCloud[16]. These tools were used to link to entities from Wikidata and DBpedia. We used the annotated entities from the STM corpus as the experimental data. However, since there is no gold standard for the dataset, we only computed the coverage metric ζ = *# of linked entities* divided

[15] https://www.textrazor.com/docs/rest
[16] https://www.meaningcloud.com/developer

by *# of all entities*. Figure 4.9 summarizes the coverage percentage for the evaluated tools. The results suggest that Falcon is most promising tool to be Incorporated for automatic named entity recognition and alignment.

4.4 Discussion

We model scholarly knowledge communicated in the scholarly literature following the abstract concept of `ResearchContribution` with a simplistic concept description. This is especially true if compared to some conceptual models of scholarly knowledge that can be found in the literature, e.g., the one by Hars [Har01]. While comprehensive models are surely appealing to information systems, e.g., for the advanced applications they can enable, we think that populating a database with a complex conceptual model is a significant challenge, one that remains unaddressed. Conscious of this challenge, for the time being we opted for a simplistic model with lower barriers for content creation.

A further and more fundamental concern is the granularity with which scholarly knowledge can realistically be acquired, beyond which the problem becomes intractable. How graph data models and management systems can be employed to represent and manage granular and large amounts of interconnected scholarly knowledge as well as knowledge evolution is another open issue.

With the evaluation of the first iteration of frontend development we were able to scrutinize various aspects of the infrastructure and obtain feedback on user interaction, experience and system acceptance. Our results show that the infrastructure meets key requirements: it is easy to use and users can flexibly create and curate research contributions. The results of the questionnaire (Table 4.1) show that with the exception of "Guidance Needed", all aspects were evaluated above average (i.e., , positively). The results suggest that guidance from an external instructor is not needed, reinforcing the usability requirement of the frontend. All case study participants displayed an interest in the ORKG and provided valuable input on what should be changed, added or removed. Furthermore, participants suggested to integrate the infrastructure with (digital) libraries, universities, and other institutions.

Based on these preliminary findings, we suggest that authors are willing to provide structured descriptions of the key contribution published in their articles, answering our first sub-research question **RQ1.1**. The practicability of crowdsourcing such descriptions at scale and in heterogeneous research communities needs further research and development.

In support of our second sub-research question **RQ1.2**, we argue that the presented infrastructure prototypes the integration of both crowdsourced and automated modes of semantic scholarly knowledge acquisition and curation. With the NLP task models and experiments as well as designs for their integration in the frontend, we suggest that the building blocks for integrating automated techniques in user interfaces for manual knowledge acquisition and curation are in place. In addition to these two modes, [Sto+18] have proposed a third mode whereby scholarly knowledge is acquired as it is generated during the research lifecycle, specifically during data analysis by integrating with computational environments such as Jupyter.

4.5 Summary

This chapter described the first steps of a larger research and development agenda that aims to enhance document-based scholarly communication with semantic representations of communicated scholarly knowledge. In other words, we aim to bring scholarly communication to the technical standards of

the 21st century and the age of modern digital libraries. We presented the architecture of the proposed infrastructure as well as an implementation. The frontend has seen substantial development, driven by earlier user feedback. We have reported here the results of the user evaluation to underpin the development of the current frontend. By integrating crowdsourcing and automated techniques in natural language processing, initial steps were also taken and evaluated that advance multi-modal scholarly knowledge acquisition using the ORKG. Our work here build the bases that is needed to create different NLP extraction techniques to add information into the knowledge graph, as well as intelligent applications capable of using the structured descriptions for exploration and retrieval purposes.

CHAPTER 5

Dynamic Blind Information Extraction Pipelines

Since the early 21st century [BHL01b], there has been continuous effort to extend the Web with a global data graph using the Resource Data Framework (RDF) to publish structured data on the Web. One of the pivotal steps in this effort was the emergence of publicly available Knowledge Graphs (KG) such as DBpedia [Aue+07] and Yago [FGG+07] as large sources of structured data. Since then, these KGs have become a rich sources of structured content used in various applications, including Question Answering (QA), fact checking, and dialog systems [Bas+21]. The research community developed numerous approaches to extract triple statements [YML19], keywords/topics [Cui+11; Cui+14], tables [Yu+; Ibr+19; Hou+19], or entities [Sak+19; Sak+20] from unstructured text to complement KGs. Despite extensive research, public KGs are not exhaustive and require continuous effort to align newly emerging unstructured information to the concepts of the KGs.

This work was motivated by an observation with recent approaches [Sak+19; Del19; Yu+; Dub+18a; Sak+20] that automatically align unstructured text to structured data on the Web. Such approaches are not viable in practice for extracting and structuring information because they only address very specific subtasks of the overall information extraction problem. If we consider the exemplary sentence *Rembrandt painted The Storm on the Sea of Galilee. It was painted in 1633.* (cf. Figure 5.2). To extract statements aligned with the DBpedia KG from the given sentences, a system must first recognize the entities and relation surface forms in the first sentence. The second sentence requires an additional step of the coreference resolution, where *It* must be mapped to the correct entity surface form (namely, *The Storm on the Sea of Galilee*). The last step requires mapping of entity and relation surface forms to the respective DBpedia entities and predicates.

There has been extensive research in aligning concepts in unstructured text to KG, including entity linking [Dub+18a; Hof+11; FS10a], relation linking [Sak+20; Sin+17; Bas+21], and triple classification [Don+19]. However, these efforts are disjoint, and little has been done to align unstructured text to the complete KG triples (i.e., represented as subject, predicate, object) [KI17; Wei+20]. Furthermore, many entity and relation linking tools have been reused in pipelines of QA systems [Sin+18; Kim+17]. The literature suggests that once different approaches put forward by the the research community are combined, the resulting pipeline-oriented integrated systems can outperform monolithic end-to-end systems. For example, Liang et al. [Lia+20a] propose a modular QA system built reusing a variety of existing NLP components that outperforms all existing end-to-end methods on the DBpedia-based QA task. For the blind (or open) information extraction task, however, to the best of our knowledge,

approaches aiming at dynamically integrating and orchestrating various existing components do not exist.

Based on these observations, we build a framework that enables the integration of previously disjoint efforts on the information extraction task under a single umbrella. In this chapter we present the Plumber framework (cf. Figure 5.1) for creating Information Extraction (IE) pipelines. Plumber integrates 40 reusable components released by the research community for the subtasks entity linking, relation linking, text triple extraction (subject, predicate, object), and coreference resolution. Researchers can also use the framework for running IE components independently for specific subtasks such as triple extraction and entity linking. Overall, there are 433 different composable information extraction pipelines (generated by the possible combination of the available 40 components). Plumber implements a transformer-based classification algorithm that intelligently chooses a suitable pipeline based on the unstructured input text.

The content of this chapter is based on the following publications [Jar+21c; Jar+21a]. The results and contributions of this chapter aims to answer the following research question and tackles the subsequent research challenge.

> RQ2: How does the dynamic selection of pipelines based on the input text affect the end-to-end information extraction task?
>
> RC2: Data Mining and Information Extraction from Scholarly Text.

To answer this research question fully, we present, describe, evaluate, and discuss Plumber. The contributions of this chapter are the following:

- The Plumber framework is the first of its kind for dynamically assembling and evaluating information extraction pipelines based on sequence classification techniques and for a given input text. Plumber is easily extensible and configurable, thus enabling the rapid creation and adjustment of new information extraction components and pipelines.
- A collection of 40 reusable IE components that can be combined to create 433 distinct IE pipelines.
- We perform an exhaustive evaluation of Plumber on the three knowledge graphs DBpedia, Wikidata, and ORKG to investigate the efficacy of Plumber in creating KG triples from unstructured text.
- We demonstrate that independent of the underlying KG; Plumber can find and assemble different extraction components to produce optimized KG triple extraction pipelines, outperforming existing baselines and approaches.

The rest of the chapter is structured as it follows; We start with a motivating example in Section 5.1 and highlight the problem. Section 5.2 formalizes the solution proposed by Plumber. In Section 5.3 we discuss the architecture and details of the Plumber framework. Section 5.4 presents copious evaluation about the framework, error propagation, and ablation studies. Followed by Section 5.5 with discussion about the results and insights gained from this research. Finally, a summary of all the work presented in the chapter in Section 5.6.

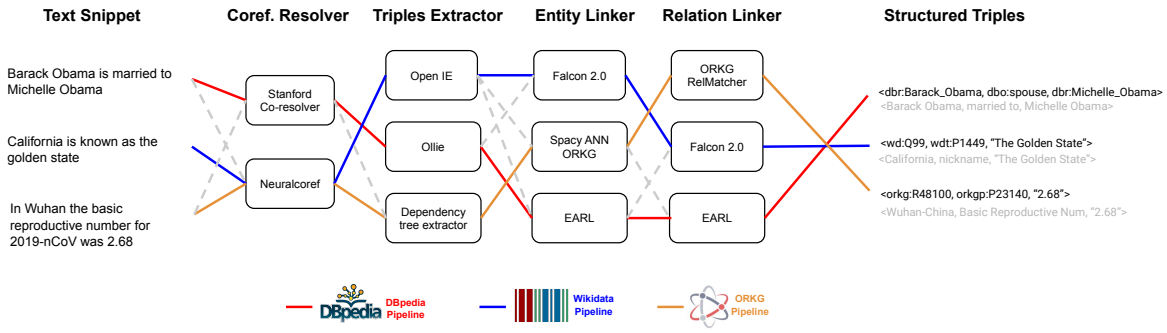


Figure 5.1: **Plumber in action**: Three information extraction pipelines that convert natural language text into structured triples aligned with respective knowledge graphs. The optimal pipeline for each text snippet and corresponding KG is highlighted.

5.1 Motivating Example

Let us consider as a running example the sentence *Rembrandt painted The Storm on the Sea of Galilee. It was painted in 1633*. The sentence can be represented using the DBpedia vocabulary as follows:

```
@prefix dbr: <http://dbpedia.org/resource/>.
@prefix dbp: <http://dbpedia.org/property/>.

dbr:Rembrandt dbp:artist dbr:The_Storm_on_the_Sea_of_Galilee.
    dbr:The_Storm_on_the_Sea_of_Galilee dbp:year "1633".
```

Multiple steps are required to extract these formally represented statements from the given text. First, the pronoun *it* in the second sentence should be replaced by *The Storm on the Sea of Galilee* using a coreference resolver. Next, a triple extractor should extract the correct text triples from the natural language text, i.e., `<Rembrandt, painted, The Storm on the Sea of Galilee>`, and `<The Storm on the Sea of Galilee, painted in, 1633>`. In the next step, the entity and relation linking component aligns the entity and relation surface forms extracted in the previous step to the DBpedia entities: `dbr:Rembrandt` for *Rembrandt van Rijn*, and `dbr:The_Storm_on_the_Sea_of_Galilee` for *The Storm on the Sea of Galilee*, and for relations: `dbp:artist` for *painted*, and `dbp:year` for *painted in*.

There exists a plethora of techniques and components for extracting such statements from a given text. However, the performance of the tools varies widely and depends strongly on the input text (cf. [Sin+18]). Figure 5.2 illustrates our running example and shows three Plumber IE pipelines with different results. In Pipeline 1, the coreference resolver is unable to map the pronoun *it* to the respective entity in the previous sentence. Moreover, the triple extractor generates incomplete triples, which also hinders the task of the entity and relation linker in the last step. Pipeline 2 uses a different set of components, and its output differs from the first pipeline. Here, the coreference resolution component is able to correctly co-relate the pronoun *it* to *The Storm on the Sea of Galilee*, and extract the text triple correctly. However, the overall result is only partially correct because the second triple is not extracted. Also, the entity linking component is not able to spot the second entity. It is important

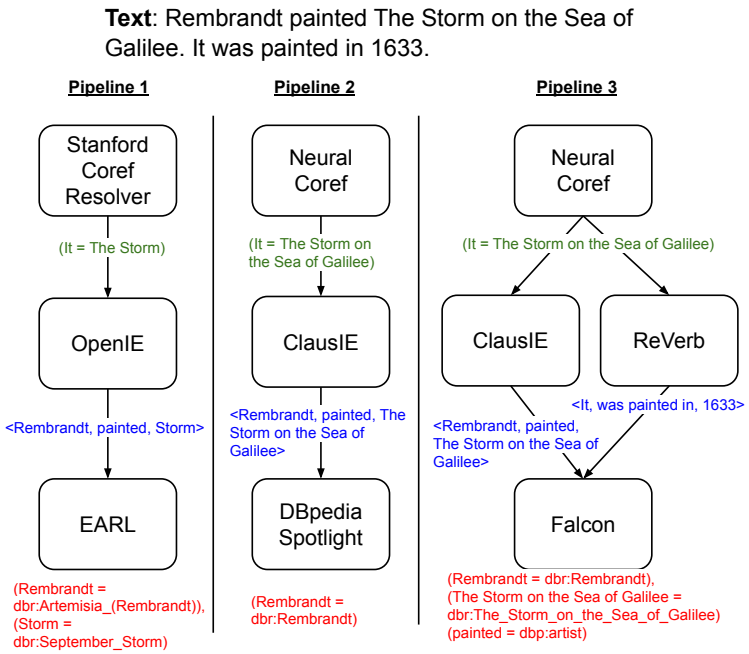


Figure 5.2: **Three example information extraction pipelines** showing different results for the same text snippet. Each pipeline consists of coreference resolution, triple extractors, and entity/relation linking components. For the sake of readability, we hide some intermediate triples and mappings. DBpedia was chosen over other KGs due to its human readable URIs.

to note that the entity linking component in the second pipeline (i.e., DBpedia Spotlight [Dai+13]) does not perform relation linking. Hence, even if the information extraction step produces the correct results, triples could not be mapped correctly.

Pipeline 3 correctly extracts both triples. This pipeline employs the same component as the second pipeline for coreference resolution but also includes an additional information extraction component (i.e., ReVerb [FSE11]) and a joint entity and relation linking component, namely Falcon [Sak+19]. With this combination of components, the text triple extractors were able to compensate for the loss of information in the second pipeline by adding one more component. Using the extracted text triples, the last component of the pipeline, a joint entity and relation linking tool, can map both triple components correctly to the corresponding KG entities.

With the availability of a large pool of components, such as those employed in PLUMBER, a suitable pipeline for a given text can be identified experimentally by executing all possible pipelines. However, this brute force approach is impractical. Therefore, we suggest a heuristic-based approach for identifying a suitable candidate pipeline for a given input text.

5.2 Approach Formalization

An end-to-end information extraction pipeline is composed of all IE tasks needed to transform a sequence of natural language text into a set of structured triples in the form of (*subject, predicate,*

Table 5.1: **Symbol notation** used to formalize PLUMBER pipelines interfaces.

P	IE extraction pipeline P composed of several IE components
Θ, θ	Set of coreference resolution (CR) components Θ and individual CR components $\theta \in \Theta$
Z, ζ	Set of triple extractor (TE) components Z and individual TE components $\zeta \in Z$
Ψ, ψ	Set of entity linking/relation linking components Ψ and individual EL/RL components ψ
Ω, ω	Set of end-to-end (E2E) components Ω and individual E2E components ω
Υ, υ	Set of KG triples Υ and individual KG triple υ
$\Lambda(.)$	Transformation function that enriches a text T
c	Coreference chain $c = \{m, a\}, m := Mention, a := Alias$
γ	Text triple $\gamma =< s, p, o >, s := subject, p := predicate, o := object$
λ	Mapping pair $\lambda = (m, u), m := Mention, a :=$ KG uri
K	Knowledge Graph K also referred to as Knowledge Base

object). However, since each component of the IE pipeline performs different tasks, we first formalize the interfaces of the IE tasks. We then define the problem statement, a formal approach implemented in PLUMBER, and how pipelines are generated.

5.2.1 Defining Various IE Tasks Interfaces

We formally define a pipeline P as a triple extraction and alignment function, from text T to a set of aligned KG triples Υ. Figure 5.3 illustrates a running example with the formalization shown here, and few instances of the intermediate results.

$$P : T \rightarrow \Upsilon \tag{5.1}$$

Text element T is a white-spaced separated sequence of words and sentences. Let $\oplus$ be the composition operator (i.e., the input of a function is the output of the previous one). We define P as a composition of four sub-functions, each corresponding to an IE task in the pipeline:

$$P := \Theta \oplus Z \oplus \Psi \oplus \Omega \tag{5.2}$$

An overview of the notation used in the problem formalization can be found in Table 5.1.

(i) Coreference Resolution (CR): The first step is to disambiguate the input text and replace pronouns and acronyms with its associated entity mention. This step is formally defined as:

$$\theta := T \rightarrow T', T' := \Lambda(T, c), c := \{(m, a) \subseteq T | m, a \neq \emptyset\} \tag{5.3}$$

where T' is a text resulting from the transformation function Λ and the coreference chain c. m is the mention in the text T and a is the pronoun or other alias that refers to the mention m. The resulting text T' is a text without ambiguities in mentions and pronouns.

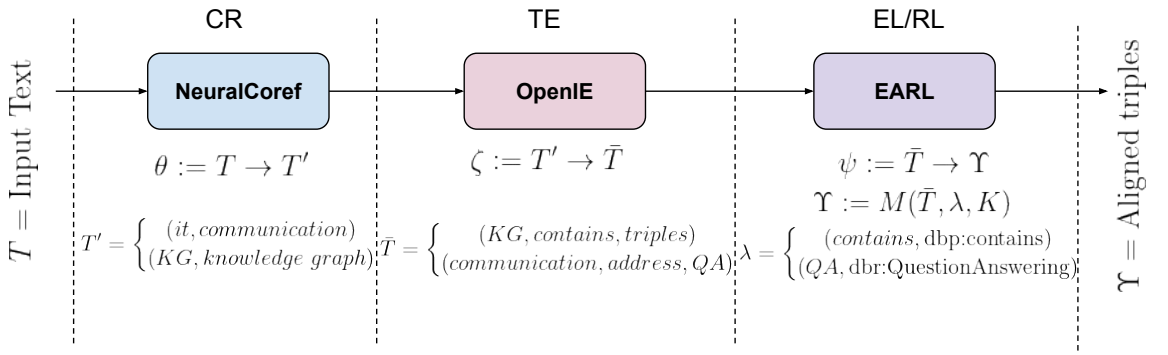


Figure 5.3: **Running example of IE tasks formalization**. Sample pipeline indicating which part of the pipeline correlates to the formalization.

(ii) Text Triple Extraction (TE): For the second step we define a text triple as combination of three keyphrases or text snippets usually in the form of (subject, verb, object). TE components extract such textual triples from the disambiguated text T'. Formally:

$$\zeta := T' \rightarrow \bar{T}, \bar{T} := \Lambda(T', \gamma), \gamma := \{(s,p,o) \in T' | s,p,o \neq \emptyset\} \tag{5.4}$$

Each triple set γ is formed of triplets (i.e., $\{s,p,o\}$). The transformation function Λ in this step enriches the disambiguated text T' with set of triples producing $\bar{T}$.

(iii) Entity and Relation Linking (EL/RL): The third and concluding step in the pipeline is the alignment of triples in $\bar{T}$ to a knowledge graph K. We define a KG triple $\upsilon \in \Upsilon$ similarly to a text triple γ whereby, they have the same structure alike $\{s,p,o\}$. However, a KG triple assumes that the subject s and predicate p components are URIs referring to a KG K, and the object o is either a URI as well or a literal value (i.e., string snippet). Formally, it is defined as:

$$\begin{aligned} \psi &:= \bar{T} \rightarrow \Upsilon, \Upsilon := M(\bar{T}, \lambda, K), \\ \lambda &:= \{(m,u), m \in \bar{T}, u \subset K | m,u \neq \emptyset\} \end{aligned} \tag{5.5}$$

Here $M(\bar{T}, \lambda, K)$ denotes a mapping function that takes the enriched text $\bar{T}$, a knowledge graph K (which is constant), and a set of mappings λ to construct a set of aligned KG triples Υ.

(iv) End-to-End Extraction (E2E): resembles the composed pipelines P and is defined as $\omega := T \rightarrow \Upsilon$. End-to-End pipelines produces results that are concatenated to results of the Ψ components.

5.2.2 Generating Candidate IE Pipelines

Generating candidate pipelines relies on the interfaces of the IE tasks and the set of requirements r. The set of pipelines $\xi(r)$ is populated with candidate pipelines ϱ_i following a composition

$$\varrho_i := \oplus_{\tau \in \Delta} \{\chi_r^\tau\} \tag{5.6}$$

where Δ the list of IE tasks following the specifications of the interfaces formalized previously. χ_r^τ a set of possible IE components that perform the IE task τ and comply with the requirements r (e.g., the knowledge graph K). It is created by concatenating IE components carrying out a task τ

(i.e., concatenating components is running them in parallel and concatenating the results). If $\|$ denotes the concatenation of IE components, then the set of IE components χ is defined as follows:

$$\chi_r^\tau := \|_i \{C(\tau)\}, \text{for } i = 1...n \tag{5.7}$$

whereby $C(\tau)$ retrieves a component from the set of IE components addressing task τ and n is the number of needed components per task. The n parameter is introduced to limit the space of candidates generation. Hence, pipelines can be generated and added to the pool of IE pipeline selection mechanism.

5.2.3 Determining Suitable IE Pipeline

Problem: here we tackle the pipeline selection problem. The pipeline selection problem deals with finding a suitable pipeline of IE components ρ_s^r, for a sequence of text s and a set of requirements r. Formally, we define the optimization problem as follows:

$$\rho_s^r := \arg \max_{\varrho \in \xi(r)} \{\mathcal{Q}(\varrho, s)\} \tag{5.8}$$

where $\xi(r)$ constitute the set of IE extraction pipelines that conform to the requirements r and $\mathcal{Q}(\varrho, s)$ corresponds to the estimated performance of a pipeline ϱ on a text sequence s.
Solution: We model the problem at hand as a $\mathcal{K}$-class classification problem [LI97]. In order to be able to solve this problem we decompose it into a series of smaller and simpler two-class problems. Suppose we have $\mathcal{K}$ pipelines (i.e., classes). Let $\mathcal{W}$ be the training set for a $\mathcal{K}$-class problem:

$$\mathcal{W} := \{(X_l, Y_l)\}_{l=1}^L \tag{5.9}$$

where X_l is an input text sequence, $Y_l \in \mathbb{R}^{\mathcal{K}}$ is the desired output, and L is the number of training data. Following the class decomposition methods [Ana+95; CY93; FT95], a $\mathcal{K}$-class problem can be divided into $\mathcal{K}$ two-class problems. The training set for each of the two-class problems is as follows:

$$\mathcal{W}_i := \left\{(X_l, y_l^i)\right\}_{l=1}^L, \text{for } i = 1, ..., \mathcal{K} \tag{5.10}$$

where $y_l^i \in \mathbb{R}^1$ is the desired output defined as:

$$y_l^i = \begin{cases} 1 - \epsilon & , X_l \in \mathcal{C}_i \\ \epsilon & , X_l \in \bar{\mathcal{C}}_i \end{cases} \tag{5.11}$$

whereby ϵ is a small positive real number and $\bar{\mathcal{C}}_i$ denotes all classes except $\mathcal{C}_i$. A sequence classifier Γ can now be trained on the decomposed training dataset $\mathcal{W}$ and is able to classify the performance of a pipeline $\mathcal{Q}(\varrho, s)$ into a class $\kappa \in \mathcal{K}$ that maps to a pipeline configuration (i.e., a set of IE components). This is the best pipeline to run on the input sequence s. Hence, we rewrite Equation 5.8 as follows:

$$\rho_s^r := \arg \max_{\kappa \in \mathcal{K}} \{\Gamma(s, \xi(r))\} \tag{5.12}$$

which stands for a problem of classifying a sequence of text s based on a set of candidate pipelines $\xi(r)$.

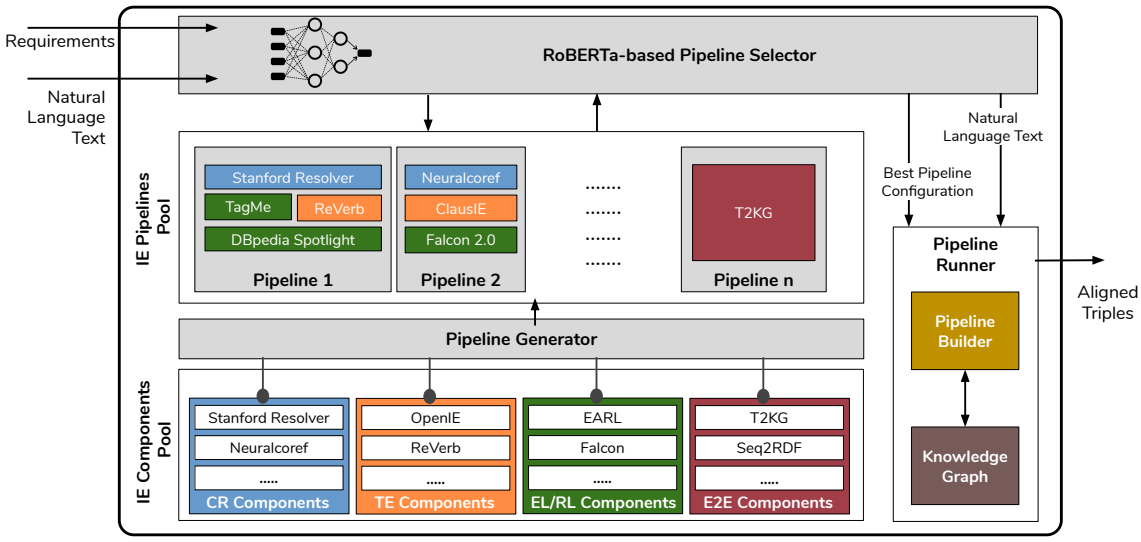


Figure 5.4: **Overview of Plumber's architecture** highlighting the components for pipeline generation, selection, and execution. Plumber receives an input sentence and requirement (underlying KG) from the user. The framework intelligently selects the suitable pipeline based on the contextual features captured from the input sentence.

5.3 Dynamic Pipelining Framework

Plumber orchestrates and evaluates IE components to select the most suitable pipeline configuration based on the input text for the information extraction task. We now detail its architecture.

5.3.1 Architecture Overview

Plumber has a modular design (see Figure 5.4) where each component is integrated as a microservice. To ensure a consistent data exchange between components, the framework maps the output of each component to a homogeneous data representation using the Qanary [Bot+16] methodology. Qanary follows the linked data principles [Ber] employing an ontology to systematize the process of connecting components. Plumber follows three design principles: i) *Isolation*, the IE components are independent from each other and they can be accessed through exchangeable interfaces; ii) *Reusability*, the framework is open source and can be reused in different contexts and variations; iii) *Extensibility*, Plumber provides common interfaces to expand and add further IE components in such a way that new components and tools are directly integrated in the framework and operate within the pipelines. The design of the framework is inspired by the work of [Bot+16; Sin+18; URa15].

Plumber uses a RoBERTa [Liu+19a] based classifier that given a text and a set of requirements, Plumber predicts the most suitable pipeline to extract KG triples.

Plumber includes the following modules:

- **IE Components Pool**. All information extraction components that are integrated within the framework are parts of the pool. The components are divided based on their respective tasks, i.e., coreference resolution, text triple extraction, as well as entity and relation linking.

- **Pipeline Generator**. This module creates possible pipelines depending on the requirements of the components (i.e., the underlying KG). Users can manually select the underlying KG and, using the metadata associated with each component, PLUMBER aggregates the components for the concerned KG.
- **IE Pipelines Pool**. PLUMBER stores the configurations of the possible pipelines in the pool of pipelines for faster retrieval and easier interaction with other modules.
- **Pipeline Selector**. Based on the requirements (i.e., underlying KG) and the input text, a RoBERTa based model extracts contextual embeddings from the text and classifies the input into one of the possible classes. Each class corresponds to one pipeline configuration that is held in the IE pipelines pool.
- **Pipeline Runner**. Given the input text, and the generated pipeline configuration, the module executes the pipeline to add the extracted triples to the KG.

5.4 Evaluation

In this section, we detail the empirical evaluation of the framework in comparison to baselines on different datasets and knowledge graphs.

5.4.1 Experimental Setup

Knowledge Graphs. To study the effectiveness of PLUMBER in building dynamic information extraction pipelines, we use the following KGs during our evaluation:
DBpedia [Aue+07] (DBP) is containing information extracted automatically from Wikipedia info boxes. DBP consists of approximately 11.5B triples [Sak+19].
Wikidata [VK14] (WD) is a crowd-sourced knowledge base providing structured data for integration in Wikipedia. In contrast to DBP, WD also allows user created entities and predicates. WD consists of over 4.9B triples [Mal+].
Open Research Knowledge Graph [Jar+19b] (ORKG) collects structured scholarly knowledge published in research articles, using crowd sourcing and automated techniques. In total, ORKG consists of approximately 1.8M triples.

Datasets. Throughout our evaluation, we employed a set of existing and newly created datasets for structured triple extraction and alignment to knowledge graphs: the WebNLG [Gar+17] dataset for DBP, the T-Rex [ElS+18] dataset for WD, and COV-triples for ORKG.
WebNLG[1] is the Web Natural Language Generation Challenge. The challenge introduced the task of aligning unstructured text to DBpedia. In total, the dataset contains 46K triples with 9K triples in the testing and 37K in the training set.
T-Rex[2] is a dataset of a large scale alignment of Wikipedia text with the Wikidata. It comprising approximately 11M triples aligned to the WD knowledge graph. The data was split in 80/20 ratio for training and testing.

[1] https://webnlg-challenge.loria.fr/
[2] https://hadyelsahar.github.io/t-rex/

COV-triples is a handcrafted dataset that focuses on COVID-19 related scholarly articles. The COV-triples dataset consists of 21 abstracts from peer-reviewed articles and aligns the natural language text to the corresponding KG triples into the ORKG. Three Semantic Web researchers verified annotation quality, and triples approved by all three researchers are part of the dataset. The dataset contains only 75 triples. Hence, we use the WebNLG dataset for training, and 75 triples are used as a test set.

Components and Implementation. The Plumber framework integrates 40 components, shown in Table 5.2 along with the associated subtasks and underlying KG. Our framework is implemented in Python and we adapt a pre-trained version of RoBERTa from its public GitHub[3] and fine-tuned it on the employed datasets. All experiments were performed on a system with 768GB RAM, 96 CPUs, and one GPU (NVIDIA GeForce 1080 Ti). The implementation code of Plumber and all related resources are publicly available online[4].

Baselines. We include the following baselines:

- *T2KG* [KI17] is an end-to-end static system aligns a given natural language text to DBpedia KG triples.
- *Frankenstein* [Sin+18] dynamically composes Question Answering pipelines over the DBpedia KG. It employs logistic regression based classifiers for each component for predicting the accuracy and greedily composes a dynamic pipeline of the best components per task. We adapted Frankenstein for the information extraction over DBpedia and Wikidata since some of its components also perform entity and relation linking.
- *KnowledgeNet* [Mes+19] represents a benchmarking dataset for information extraction alongside a baseline model. The KnowledgeNet baseline model performs information extraction and population on the WD knowledge graph.

Training the model. Plumber relies on a classification model to find the suitable pipeline. Each pipeline is represented as a class, making this a multi-class classification problem. To train the model; for every entry in the datasets, every possible pipeline is composed and ran on the input snippet. The results in terms of f1-scores are used to decide which pipeline performed better than the others. Next step is for our underlying model (RoBERTa) is to create contextualized embeddings from the input text and learn to classify it onto its corresponding class (i.e., the best performing pipeline from the IE pipelines pool). Adding new IE components to the pool of Plumber requires the retraining of the model over all possible pipelines which is costly. We choose a transformer-based architecture due to its ability to encode the contextual knowledge from the input text, providing more accurate classification.

5.4.2 Experiments

This section summarizes a variety of experiments to compare the Plumber framework against other baselines. Note, that evaluating the performance of individual components or their combination is out of this evaluation's scope, since they were already used, benchmarked, and evaluated in the respective

[3] https://github.com/pytorch/fairseq/
[4] https://github.com/YaserJaradeh/ThePlumber

Table 5.2: **IE components implemented and integrated** within the PLUMBER framework along with their respective publications, API links, underlying KGs, and respective task.

Pipeline Component	IE Task	Knowledge Graph	Open Source	Custom Built	Rest API
Ollie [Mau+12]	TE	-	✓	✗	✗
OpenIE [AJM15]	TE	-	✓	✗	✗
ClausIE [DG13]	TE	-	✓	✗	✗
MinIE [GGC17]	TE	-	✓	✗	✗
POS Extractor[i]	TE	-	✓	✓	✗
Dependency Extractor[ii]	TE	-	✓	✓	✗
Graphene [Cet+18]	TE	-	✓	✗	✗
ReVerb [FSE11]	TE	-	✓	✗	✗
Ro Extractor	TE	ORKG	✓	✓	✗
Stanford KBP Extractor [CPB]	TE	KBP[viii]	✓	✗	✗
Falcon [Sak+19]	EL+RL	DBP	✓	✗	✓
Falcon 2.0 [Sak+20]	EL+RL	WD	✓	✗	✓
EARL [Dub+18a]	EL+RL	DBP	✓	✗	✓
Spacy ANN[iii]	EL+RL	DBP+WD	✓	✓	✗
Spacy ANN[iii]	EL+RL	ORKG	✓	✓	✗
Falcon NER [Sak+19] + ES[iv]	EL+RL	DBP+WD	✓	✓	✗
Falcon 2.0 NER + ES[iv]	EL+RL	WD	✓	✓	✗
EARL NER [Dub+18a] + ES[iv]	EL+RL	DBP+WD	✓	✓	✗
Meaning Cloud[v]	EL	DBP	✗	✗	✓
Text Razor[vi]	EL	DBP+WD	✗	✗	✓
DBpedia Spotlight [Dai+13]	EL	DBP	✓	✗	✓
TagMe [FS10a]	EL	DBP	✓	✗	✓
OpenTapioca [Del19]	EL	WD	✓	✗	✓
TagMe NER [FS10a] + ES[iv]	EL	DBP+WD	✓	✓	✗
Ambiverse-nlu [Hof+11]	EL	WD	✓	✗	✗
RelMatch [Sin+17]	RL	DBP	✓	✗	✗
Stanford Coref Resolver [Rag+10]	CR	-	✓	✗	✗
NeuralCoref [CM16]	CR	-	✓	✗	✗
PyCobalt[vii]	CR	-	✓	✗	✗
HMTL [SWR19]	CR	-	✓	✗	✗

[i] Based on https://github.com/tdpetrou/RDF-Triple-API/
[ii] Adapted from https://github.com/anutammewar/extract_triplets/
[iii] https://github.com/microsoft/spacy-ann-linker/
[iv] https://www.elastic.co/elasticsearch/
[v] https://www.meaningcloud.com/
[vi] https://www.textrazor.com/technology/
[vii] https://github.com/Lambda-3/PyCobalt
[viii] https://tac.nist.gov/2017/KBP/

Table 5.3: **Performance comparison of the Plumber static pipeline against the baselines on different KGs**. The total number of mapped triples in test set (Extracted/Expected) is given in the last column to indicate how many triples the systems produce regardless of correctness.

System	Dataset	Knowledge Graph	P	R	F1	# Mapped Triples
T2KG [KI17]	WebNLG	DBP	0.133	0.140	0.135	1.26K/9.0K
KnowledgeNet [Mes+19]	T-Rex	WD	0.243	0.254	0.247	0.56M/2.2M
Frankenstein [Sin+18]	WebNLG	DBP	0.177	0.189	0.181	1.70K/9.0K
	T-Rex	WD	0.228	0.249	0.238	0.55M/2.2M
Plumber	WebNLG	DBP	**0.210**	**0.225**	**0.215**	2.02K/9.0K
	T-Rex	WD	**0.282**	**0.296**	**0.289**	0.65M/2.2M
	COV-triples	ORKG	**0.403**	**0.423**	**0.413**	32/75

publications. We report values of the standard metrics Precision (P), Recall (R), and F1 score (F1) adapted from [CHA20]. In all experiments, end-to-end components (e.g., T2KG) are not part of Plumber.

We note that Plumber is able to use 433 pipelines; For DBpedia: 3 CRs, 8 TEs, and 10 EL/RLs, hence, 3 * 8 * 10=240. And the same for Wikidata: 3CRs * 8TEs * 7EL/RL = 168. For the ORKG: 4CRs * 3TEs * 2EL/RL=24 pipeline. And one single pipeline for the KBP mappings. In total: 240+168+24+1=433 pipeline.

Performance of Static Pipelines. In this experiment, we report results of the static pipelines, i.e., no dynamic selection of a pipeline based on the input text is considered. We ran all 433 pipelines and Table 5.3 reports the performance of the best Plumber pipeline against the baselines. Plumber static pipeline for DBpedia comprises of NeuralCoref [CM16] for coreference resolution, OpenIE [AJM15] for text triple extraction, TagMe [FS10a] for EL, and Falcon [Sak+19] for RL tasks. For Wikidata, the static pipeline contains NeuralCoref [CM16] for coreference resolution, Graphene [Cet+18] for text triple extraction, Falcon 2.0 [Sak+20] jointly for EL and RL tasks. Also, in case of Frankenstein, we choose its best performing static pipeline. Results illustrated in the Table 5.3 confirm that the static pipeline composed by the components integrated in Plumber outperforms all baselines on DBpedia and Wikidata. We observe that the performance of pipeline approaches is better than an end-to-end monolithic information extraction approaches. Although the Plumber pipeline outperforms the baselines, the overall performance is relatively low. All our components have been trained on distinct corpora in their respective publications and our aim was to put them together to understand their collective strengths and weaknesses. Note, Frankenstein addresses the QA pipeline problem and not all components are comparable and can be applied in the context of information extraction. Thus, we integrated NeuralCoref coreference resolution component and OpenIE triple extraction component used in Plumber static pipeline into Frankenstein for providing the same experimental settings.

Static Pipeline for Scholarly KG. In order to assess how Plumber performs on domain-specific use cases, we evaluate the static pipelines' performance on a scholarly knowledge graph. We use the COV-triples dataset for ORKG. To the best of our knowledge, no baseline exists on completing KGs

Table 5.4: **10-fold cross-validation of pipeline selection classifiers** wrt. Precision, Recall, and F1 score.

Pipeline Selection Approach	Dataset	Knowledge Graph	Classification		
			P	R	F1
Random	WebNLG	DBP	0.081	0.092	0.086
	T-Rex	WD	0.090	0.103	0.096
	COV-triples	ORKG	0.092	0.114	0.102
Frankenstein [Sin+18]	WebNLG	DBP	0.732	0.751	0.741
	T-Rex	WD	0.770	0.791	0.780
	COV-triples	ORKG	0.832	0.858	0.845
Plumber	WebNLG	DBP	**0.877**	**0.900**	**0.888**
	T-Rex	WD	**0.891**	**0.912**	**0.901**
	COV-triples	ORKG	**0.901**	**0.917**	**0.909**

of research contribution descriptions over ORKG. Hence, we execute all static pipelines in Plumber tailored to ORKG to select the best one as shown in Table 5.3 (last line). Plumber pipelines over ORKG extract statements determining the reproductive number estimates for the COVID-19 infectious disease from scientific articles as shown below.

```
@prefix orkg: <http://orkg.org/orkg/resource/>.
@prefix orkgp: <http://orkg.org/orkg/property/>.

orkg:R48100   orkgp:P16022   "2.68" .
```

In this example, *orkg:R48100* refers to the city of Wuhan in China in the ORKG and *orkgp:P16022* is the property "has Ro estimate (average)". The number "2.68" is the reproductive number estimate.

Comparison of the Classification Approaches for Dynamic Pipeline Selection. In this experiment, we study the effect of the transformer-based pipeline selection approach implemented in Plumber against the pipeline selection approach of Frankenstein. For a comparable experimental setting, we re-use Frankenstein's classification approach in Plumber, keeping the underlying components precisely the same. We perform a 10-fold cross-validation for the classification performance of the employed approach. Table 5.4 demonstrates that the Plumber pipeline selection significantly outperforms Frankenstein across all knowledge graphs.

Performance Comparison for Open Information Extraction Task. Our third experiment focuses on comparing the performance of Plumber against previous baselines for an end-to-end information extraction task. We also report the values of best performing static pipelines from Table 5.3. The results in Table 5.5 illustrate that the dynamic pipelines built using Plumber for information extraction outperform the best static pipelines of Plumber as well as the dynamically selected pipelines by Frankenstein. The end-to-end baselines, such as [Mes+19; KI17], significantly underperform compared to dynamic pipelines. We also observe that in cross-domain experiments for COV-triples datasets, dynamically selected pipelines perform better than the static pipeline. In the cross-domain experiment, the static and dynamic Plumber pipelines are relatively better performing than the other two KGs. Unlike components for DBpedia and Wikidata, components integrated into Plumber for ORKG

Table 5.5: **Overall performance comparison** of static and dynamic pipelines for the KG completion task.

System	Dataset	Knowledge Graph	Performance		
			P	R	F1
T2KG [KI17]	WebNLG	DBP	0.133	0.140	0.135
KnowledgeNet [Mes+19]	T-Rex	WD	0.243	0.254	0.247
Frankenstein (Static) [Sin+18]	WebNLG	DBP	0.177	0.189	0.181
	T-Rex	WD	0.228	0.249	0.238
PLUMBER (Static)	WebNLG	DBP	0.210	0.225	0.215
	T-Rex	WD	0.282	0.296	0.289
	COV-triples	ORKG	0.403	0.423	0.413
Frankenstein (Dynamic) [Sin+18]	WebNLG	DBP	0.199	0.208	0.203
	T-Rex	WD	0.244	0.263	0.253
	COV-triples	ORKG	0.403	0.424	0.413
PLUMBER (Dynamic)	WebNLG	DBP	**0.287**	**0.307**	**0.297**
	T-Rex	WD	**0.361**	**0.397**	**0.378**
	COV-triples	ORKG	**0.411**	**0.437**	**0.424**

are customized for KG triple extraction. We conclude that when components are integrated into a framework such as PLUMBER aiming for the information extraction task, it is crucial to select the pipeline based on the input text dynamically. The fine performance of PLUMBER shows that the dynamic pipeline selection for information extraction has a positive impact agnostic of the underlying knowledge graph and dataset. This also answers our research question **RQ2**.

5.4.3 Ablation Studies (Error Analysis)

PLUMBER and baselines render relatively low performance on all the employed datasets. Hence, in the ablation studies our aim is to provide a holistic picture of underlying errors, collective success, and failures of the integrated components.

In the first study, we calculate the proportion of errors in PLUMBER. The modular architecture of the proposed framework allows us to benchmark each component independently. We consider the erroneous cases of PLUMBER on the test set of the WebNLG dataset. We calculate the performance (F1 score) of the PLUMBER dynamic pipeline (cf. Table 5.5) at each step in the pipeline. Figure 5.5 presents the results of the error evaluation. Each box in the figure corresponds to an IE task. The results show that the coreference resolution components caused 21.54% of the errors, 33.71% are caused by text triple extractors, 18.17% by the entity linking components, and 26.58% are caused by the relation linking components.

We conclude that the text triple extractor components contribute to the largest chunk of the errors over DBpedia. One possible reason for their limited performance is that open-domain information extracting components were not created for a complete workflow of extraction to alignment in the graph. Also, these components do not incorporate any schema or prior knowledge to guide the extraction. We observe that the errors mainly occur when the sentence is complex (with more than one entity and

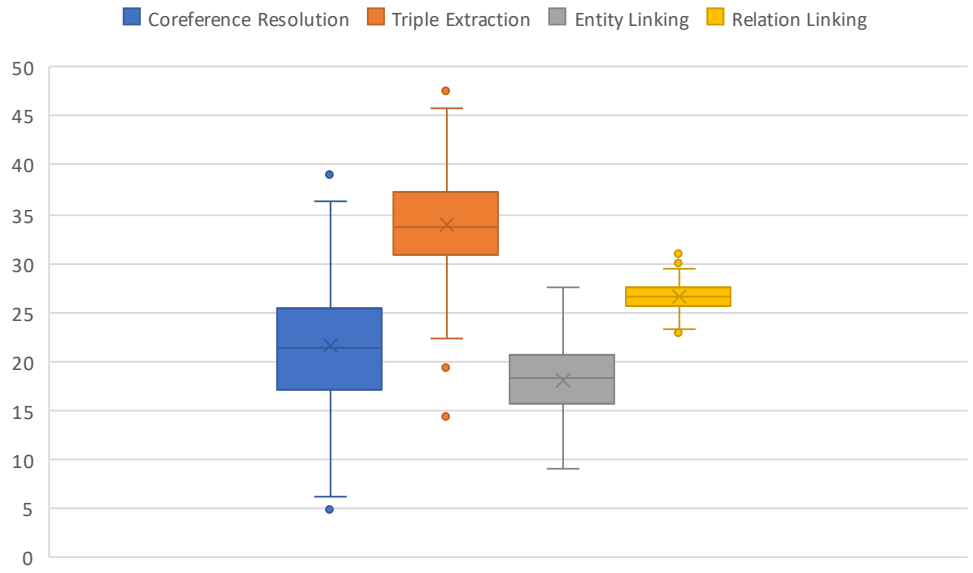


Figure 5.5: **Box plot of error percentage per IE task**. The Y axis shows the error percentage. Each box shows the error percentage by all components, the average error, and some of the outliers. Higher values means greater error rate. The figure shows that text triple extraction is the highest impacting component followed by relation linking, coreference resolution, and the least impacting is the entity linking.

predicate), or relations are not explicitly mentioned in the sentence. We further analyze the text triple extractor errors. The error analysis at the level of the triple subject, predicate, and object showed that most errors are in predicates (40.17%) followed by objects (35.98%) and subjects (23.85%).

5.4.4 Ablation Studies (Component Performance)

Aiming to understand why IE pipelines perform with low accuracy, we conduct a more in-depth analysis per IE task. In the first analysis, we evaluated each component independently on the WebNLG dataset. Researchers [Der+15; Sin+19] proposed several criterion for micro-benchmarking tools/components for KG tasks (entity linking, relation linking, etc.) based on the linguistic features of a sentence. We motivate our analysis based on the following criteria per task:

Text Triple Extraction. We consider the number of words (wc) in the input sentence (a sentence is termed by "simple" with average word length of 7.41 [Sin+18]. Sentences with higher number of words than seven are complex sentences). Furthermore, having a comma in a sentence (sub-clause) to separate clauses is another factor. Atomic sentences (e.g., "cats have tails") are a type of sentence that also affects triples extractors' behavior. Moreover, nominal relation as in "Durin, son of Thorin" is another impacting factor on the performance. Uppercase and lowercase mentions of the words (i.e., correct capitalization of the first character and not the entire word) in a sentence are standard errors for entity linking components. We consider this as a micro-benchmarking criteria.

Coreference Resolution. We focus on the length of the coreference chain (i.e., the number of aliases for a single mention). Additionally, the number of clusters is another criterion in the analysis. A cluster refers to the groups of mentions that require disambiguation (e.g., "mother bought a new phone, she is so happy about it" where the first cluster is *mother* → *she* and the second is *phone* → *it*). The presence of proper nouns in the sentence is studied as well as acronyms. Furthermore, the demonstrative nature of the sentence is also observed as a factor. Demonstrative sentences are the ones that contain demonstrative pronouns (this, that, etc.).

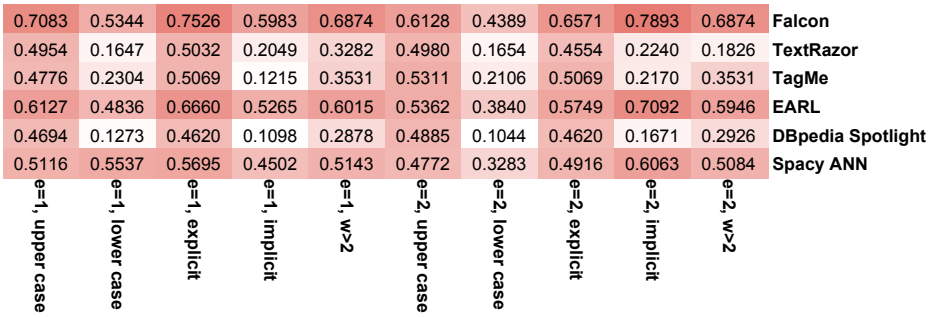
Entity Linking. The number of entities in a sentence (e=1,2) is a crucial observation for the entity linking task. Capitalization of the surface form is another criterion for micro-benchmarking entity linking tools. An entity is termed as an explicit entity when the entity's surface form in a sentence matches the KG label. An entity is implicit when there is a vocabulary mismatch. For example, in the sentence "The wife of Obama is Michelle Obama.", the surface form *Obama* is expected to be linked to `dbr:Barack_Obama` and considered as an implicit entity [Sin+19]. The last linguistic feature is the number of words (w) in an entity label (e.g., "The Storm on the Sea of Galilee" has seven words).

Relation Linking. Similar to the entity linking criteria, we focus on the number of relations in a sentence (rel=1,2). The type of relation (i.e., explicit, or implicit) is another parameter. Covered relation (sentences without a predicate surface form) is also used as a feature for micro-benchmarking: "Which companies have launched a rocket from Cape Canaveral Air Force station?" where the `dbo:manufacturing` relation is not mentioned in the sentence. Covered relations highly depend on common sense knowledge (i.e., reasoning) and the structure of the KG [Sin+19]. Lastly, the number of words (w>=N) in a predicate surface form is also considered.

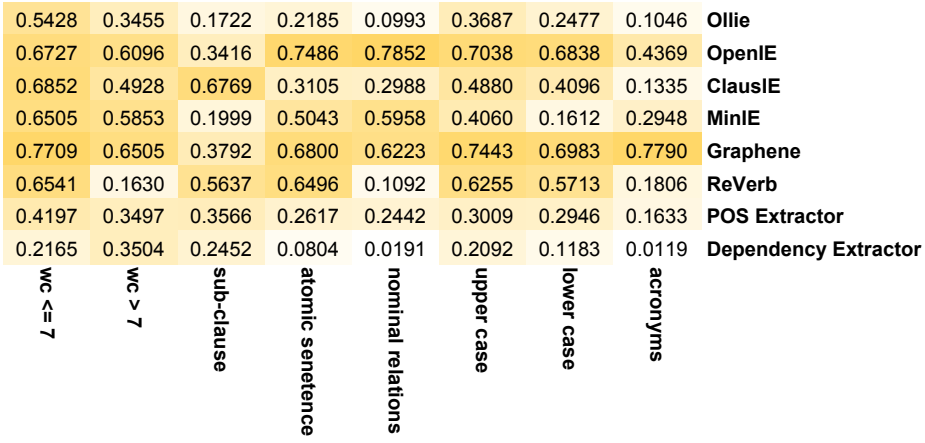
Figure 5.6 illustrates micro-benchmarking of various Plumber components per task. We observe that across IE tasks, the F1 score of the components varies significantly based on the sentence's linguistic features. In fact, there exist no single component which performs equally well on all the micro-benchmarking criteria. This observation further validates our hypothesis to design Plumber for building dynamic information extraction pipelines based on the strengths and weaknesses of the integrated components.

In Figure 5.6, all the CR components report limited performance for the demonstrative sentences (*demonstratives*). When there is more than one coreference cluster in a sentence, all other CR components observe a discernible drop in F1 score. The NeuralCoref [CM16] component performs best for *proper nouns*, whereas PyCobalt [FBH] performs best for the *acronyms* feature (almost being tied by NeuralCoref). In the TE task, Graphene [Cet+18] shows the most stable performance across all categories. However, the performance of all components (except Dependency Parser) drops significantly when the number of words in a sentence exceeds seven (wc>7). Case sensitivity also affects the performance and all components observe a noticeable drop in F1 score for lowercase entity mentions in the sentence. Similar behavior is observed for entity linking components where case sensitivity is a significant cause of poor performance. When the sentence has one entity and it is implicit (e=1, implicit); all entity linking components face challenges in correctly linking the entities to the underlying KG. Relation linking components also report lower performance for implicit relations.

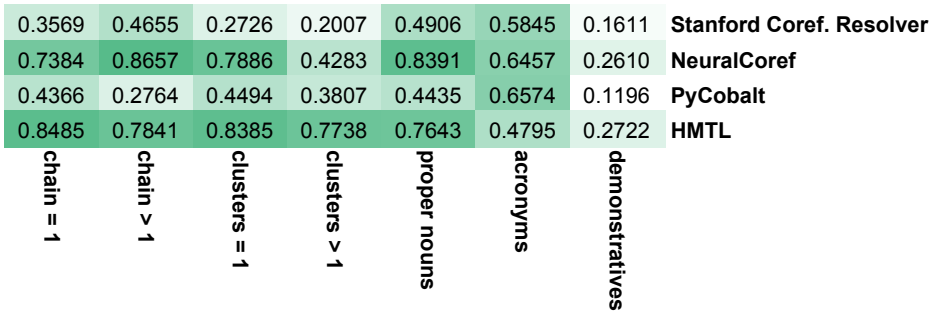
We then extended micro-benchmarking of the components over Wikidata and report their performance in isolation. We considered all the sentences present in the T-Rex test set (approx 1.2M sentences). Figure 5.7 illustrates the findings per linguistic feature for all IE subtasks. Similarly as



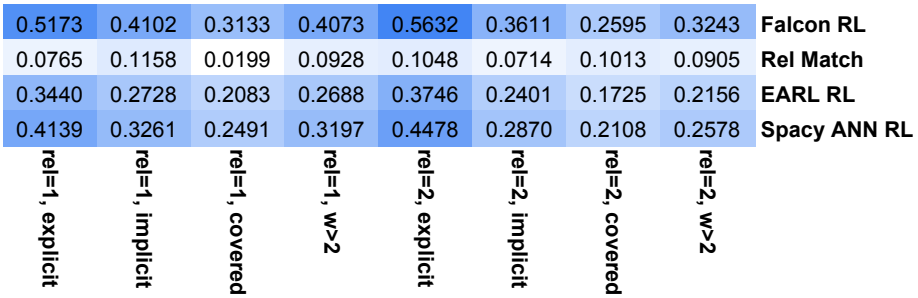
(a) F1 score heatmap of the EL task



(b) F1 score heatmap of the Text TE task



(c) F1 score heatmap of the CR task



(d) F1 score heatmap of the RL task

Figure 5.6: **Comparison of F1 scores per component for different IE tasks** based on the various linguistic features of an input sentence (number of entities, word count in a sentence, implicit vs. explicit relation, etc.). Darker colors indicate a higher F1 score.

Table 5.6: **Average runtime on all datasets**. The dynamic pipelines on WebNLG report the slowest runtime because few components have a high avg. runtime (up to 65 sec.).

System	WebNLG	T-Rex	COV-Triples
T2KG [KI17]	2.8	-	-
KnowledgeNet [Mes+19]	-	3.4	-
Frankenstein (Static) [Sin+18]	2.4	2.5	-
Frankenstein (Dynamic) [Sin+18]	10.1	3.9	2.9
PLUMBER (Static)	1.8	1.9	1.2
PLUMBER (Dynamic)	12.3	3.9	2.7

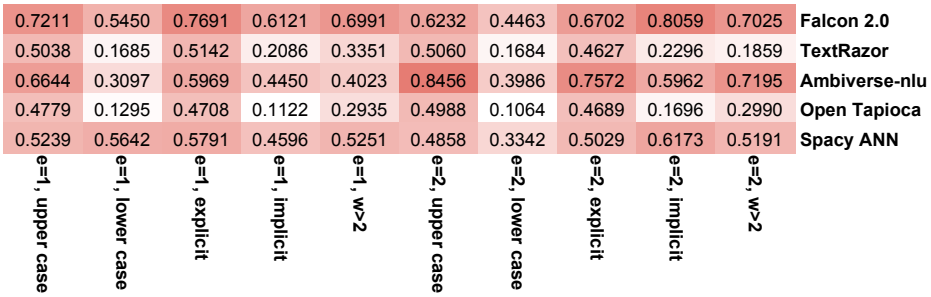
for DBpedia, we observe that no single component is superior to all micro-benchmarking criteria. Issues such as capitalization of entity surface forms continue to impact EL and TE components' overall performance negatively. Relation linking components on Wikidata inherit a similar trend as DBpedia components, where the implicit and hidden nature of relation surface forms has the highest impact on their performance.

5.5 Discussion and Use-cases

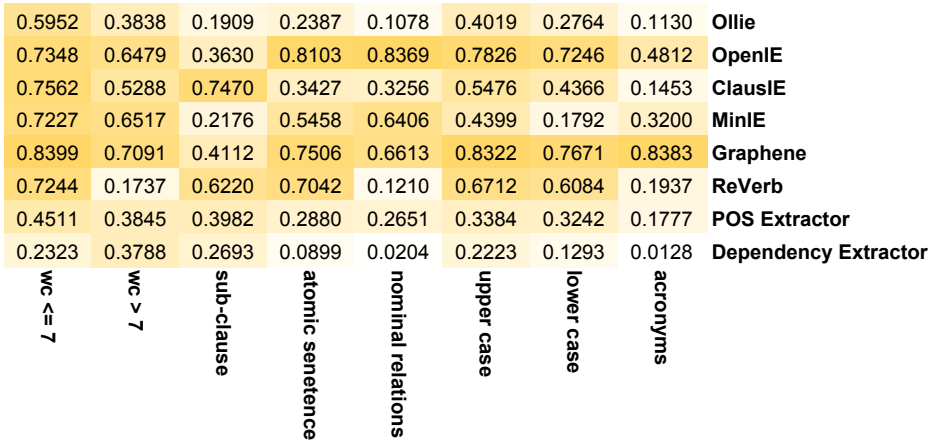
Even though the dynamic pipelines of PLUMBER outperforms static pipelines, the overall performance of PLUMBER and baselines for the information extraction task remains low. Our detailed and exhaustive ablation studies suggest that when individual components are plugged together, their individual performance is a major error source. However, this behavior is expected, considering earlier research works in other domains also observe a similar trend. For example, since its first release in 2015, the research community performed over 50,000 experiments[5] to improve EL components using the Gerbil framework [URa15]. Similarly, in 2018, Frankenstein reported the best dynamic question answering pipeline with F1 score 0.20. Within two years, the Semantic Web research community has released several components dedicated to solving entity linking and relation linking [Sak+19; Dub+18a; Mih+20], which were two loopholes identified by [Sin+18] for the QA task. At present, the QA system [Lia+20a] reuses components from Frankenstein and is a new SotA on standard complex QA dataset [Tri+17] with an F1 score 0.68. We also calculated the average runtime of PLUMBER and baselines on all three datasets (cf. Table 5.6). PLUMBER static pipeline was the fastest; however, the dynamic pipelines on DBpedia were the slowest. The main reason for the slow dynamic pipeline was the high runtime of DBpedia based relation linking components. For example, Relmatch [Sin+17] has an average runtime of 65 seconds, thus negatively impacting the overall dynamic pipeline runtime. Due to the direct impact of a component's runtime on the overall efficiency (runtime, and memory consumption).

We observe that state of the art components for information extraction still have much potential to improve their performance (both in terms of runtime and F1 score). It is essential to highlight that some of the issues observed in our ablation study are very basic and repeatedly pointed out by researchers in the community. For instance, Derczynski et al. [Der+15] in 2015, followed by Singh et al. [Sin+18]

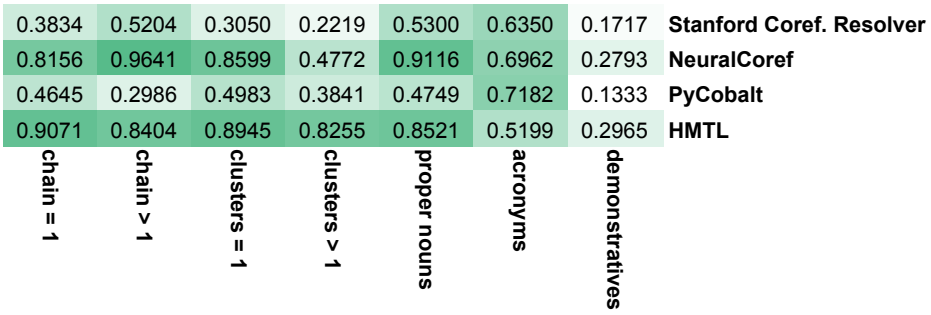
[5] http://aksw.org/Projects/GERBIL.html



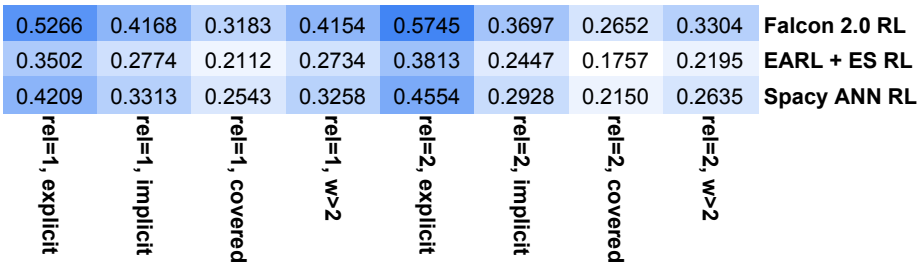
(a) F1 score heatmap of the EL task



(b) F1 score heatmap of the Text TE task



(c) F1 score heatmap of the CR task



(d) F1 score heatmap of the RL task

Figure 5.7: **Comparison of F1 scores per component for different IE tasks (on the T-Rex dataset)**. Darker colors indicate a higher F1 score. We observe that components for Wikidata show a similar trend in the performance as in Figure 5.6 where test sentences having linguistic features such as implicit entities, word count in a sentence, capitalization of entity surface form, etc. negatively impact performance.

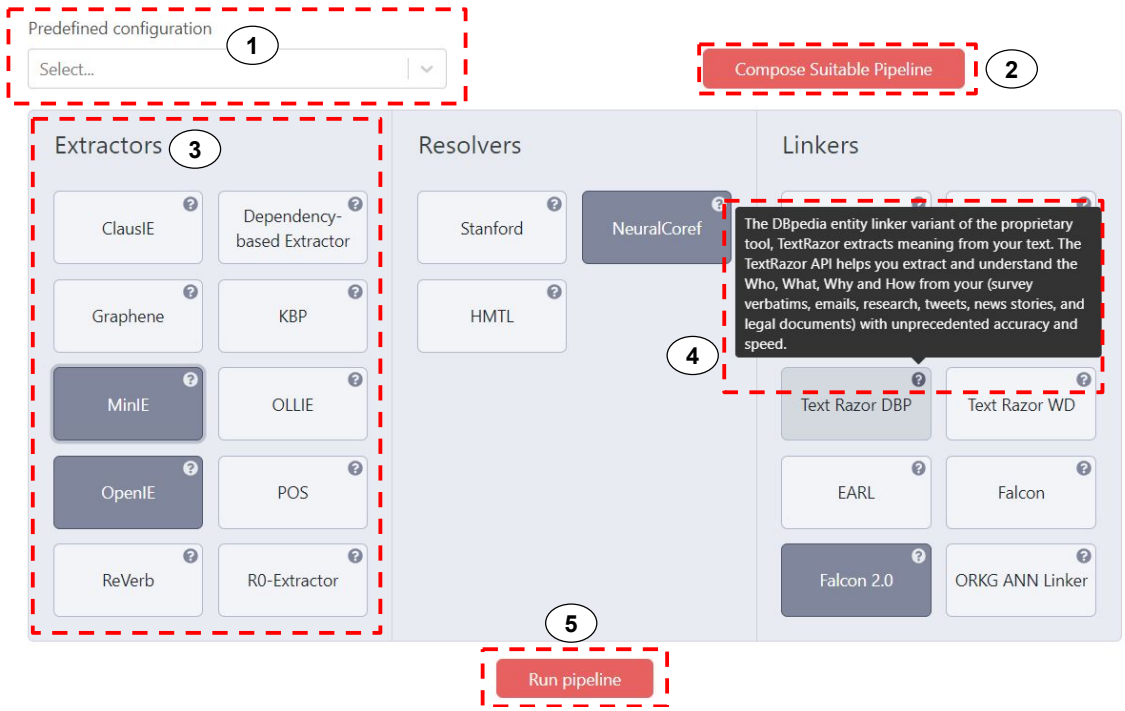


Figure 5.8: **Overview of the user interface of PLUMBER in the ORKG infrastructure**. 1) Predefined pipeline selector: used for easy access to generally stable information extraction pipeline. 2) invoke the framework to create a dynamic pipeline on-the-fly based on the input. 3) collection of IE components that can be used in conjunction manually or automatically. 4) additional information from components to better help the user interact with the system. 5) pipeline runner to display the results and get feedback.

in 2018, showed that case sensitivity is a main challenge for EL tools. Our observation in Figure 5.6 and Figure 5.7 again confirms that case sensitivity of entity surface forms remains an open issue even for newly released components. In contrast, on specific datasets such as CoNLL-AIDA, several EL approaches reported F1 scores higher than 0.90 [Yan+19], showing that EL tools are highly customized to particular datasets. In a real-world scenario like ours, the underlying limitations of approaches are uncovered. We also found relation linking and text triple extractor components contributed caused significant errors in PLUMBER's performance.

Practical Use-case: Our dynamic pipelining framework can be used in conjunction with other knowledge management systems to allow users the possibility to leverage automated extraction from natural language text. In this use case we integrate PLUMBER within the ORKG infrastructure[6] providing an access point to researchers to convert textual descriptions into structured and linked triples. Figure 5.8 depicts how the PLUMBER can be integrated within other systems (here the ORKG). Moreover, such integration allow for user feedback and comments to be fed into the system in a machine-in-the-loop way. Hence, allowing constant active-learning style improvement to the underlying model of PLUMBER.

[6] Demo video: http://www.youtube.com/watch?v=XC9rJNIUv8g

5.6 Summary

In this chapter, we presented the PLUMBER approach and framework for open information extraction. PLUMBER effectively selects the most suitable pipeline for a given input sentence using the sentential contextual features and a state-of-the-art transformer-based classification model. PLUMBER has a service-oriented architecture which is scalable, extensible, reusable, and agnostic of the underlying KG. The core idea of PLUMBER is to combine the strengths of already existing disjoint research for information extraction and build a foundation for a platform to promote reusability for the construction of large-scale and semantically structured KGs. Our empirical results suggest that the performance of the individual components directly impacts the end-to-end information extraction accuracy.

This contribution does not focus on internal system architecture or employed algorithms in a particular IE component to analyze the failures. The focus of the ablation studies is to holistically study the collective success and failure cases for the various tasks. Our studies provide the research community with insightful results over three knowledge graphs, 40 components, 433 pipelines, and test datasets collectively containing over 2.2M triples extracted from approximately 1.2M sentences. Our work is a step in the larger research agenda of offering the research community an effective way for synergistically combining and orchestrating various focused IE approaches balancing their strengths and weaknesses taking different application domains into account. applying their research to a domain driven by many different fields, consequently requiring a collaborative approach to achieve significant progress.

CHAPTER 6

Targeted Scholarly Information Extraction

Information extraction (IE) from scholarly articles is a challenging task due to the sizable document length and implicit information hidden in text, figures, and citations. By their very nature, scholarly articles tend to be dense with information and knowledge [PHH18]. The task of information extraction has been widely researched by the community in a variety of contexts [Cha+06; PY13; Jia12], including the scholarly domain [Wil+16b; NJM18]. However, information extraction from scholarly articles continues to suffer from low accuracy. Reasons include ambiguity of scholarly text, information representation in scholarly articles, and lack of training datasets [Sin+16].

Other than retrospective information extraction, initiatives such as the Open Research Knowledge Graph (ORKG) [Jar+19b], Hi-Knowledge [Jes+20], and Coda [Spa+20] collect structured scholarly information by engaging researchers in the knowledge curation process. In ORKG, information is collected by experts that extract and structure the essential information from articles. However, experts might not use the exact wording from the original article or might put forward a novel segment of text that did not exist before in the original text.

Information extraction techniques [Sar08] could play a supporting role through automated extraction, suggestions to experts or autonomously adding extracted information to a a data source (e.g., database or knowledge graph). However, blindly extracting information (i.e., factual extractions) is not suitable for scholarly data due to the large amount of information condensed into little text. Blind extraction refers to Open Information Extraction [Etz+08] that relies on propositions and facts as well as common entities and relations between them. For scholarly articles, a more targeted approach is required, whereby a system is able to extract a set of predefined properties and their corresponding values while ignoring others.

We propose MORTY, a method that leverages summarization tasks conducted by deep-learning language models to create structured summaries that can be parsed into extracted information, stored in a knowledge graph.

The content of this chapter are based on publication [JSA22] for the main part. Chapter 5 and Chapter 6 are two side to one coin addressing the same research challenge. The results and contributions of this chapter aim to answer the following research question and tackles the subsequent research challenge.

RQ3: How can we guide information extraction processes on scholarly documents?

RC2: Data Mining and Information Extraction from Scholarly Text.

In order to answer our research question we present, evaluate, and discuss MORTY. Here is a list of the contributions put farward by work presented in this chapter:

- Dataset of paper full texts with a list of property-value pairs of human-expert annotated information[1].
- An approach (MORTY) for information extraction from scholarly articles using structured summarization.
- Detailed evaluation of various baselines to study the efficacy of the proposed approach.

The reminder of the chapter is as it follows: Section 6.1 describes the research problem of this chapter. Followed by our approach to tackle the problem in Section 6.2. In Section 6.3 we present the details of the conducted evaluation, and discuss the results and what do they mean. Section 6.5 concludes the work and summarizes the key findings.

6.1 Problem Statement

Scholarly text is ambiguous and information dense. We illustrate the problem by taking a look at some textual content from this chapter (see Figure 6.1). If we want to extract a single piece of information (i.e., a property) such as the "research problem" addressed by the article, it is necessary to comprehend the text and look even behind the textual representation. In this example, the research problem is "information extraction from scholarly articles". The method of looking up certain properties such as "research problem" in the text, proves insufficient because the phrase may not exist as is or is spelled differently. This can be extend by looking up synonyms for the property or by finding verbs that represent the same intent (e.g., addresses, tackles, etc.). Other times, regardless of how the property is represented, the value itself is implicit or not represented as expected, which requires more abstractive answers than extractive ones [TK07].

For instance, in the example presented in Figure 6.1, we note that the wording of problem or any other synonym is not mentioned. Rather the closest thing to that property is "challenging task" which refers to something that is hard to achieve. Moreover, the text contains other pieces of information that are valuable but not really related to the value in question here, which should be ignored by the information extraction system. Sometimes the text includes multiple variations of a certain value like with the exact research problem "information extraction from scholarly articles" and the more generic problem "targeted information extraction", where any suitable approach would need to distinguish among them and extract the suitable value for the requested property.

We argue that these cases barely scratch the surface of the problem. Certain properties could require values placed throughout the text, combined together, and even morphed into dissimilar wording.

[1] https://github.com/YaserJaradeh/MORTY/

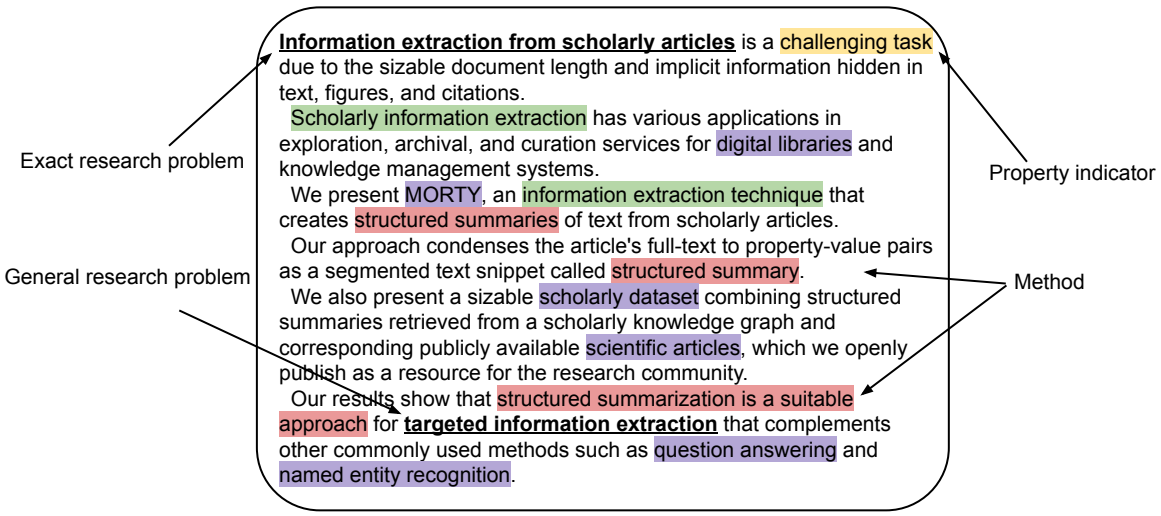


Figure 6.1: **Example of how information is hidden in the text.** Underlined and bold terms refer to the research problem. ○ show the keyword that helps locating the research problem. ○ though valuable information, they are not relevant to the extraction of the research problem. ○ refer to the method used to solve the research problem. and ○ are more indicators to the problem the text is about.

Others, cannot be found in the text, but are included in figures, tables, or even in citations [RMG15; Xia+17a]. Furthermore, some properties could be of annotation-nature, i.e., the property and the value are not in the original text, but tacit knowledge of an expert annotating an article.

Hence, out-of-the-box tools and methods for automated information extraction are unlikely to yield accurate information. We argue that blind non-tailored methods are not suitable for scholarly information extraction. Due to the heterogeneity of knowledge representation, and the concept drift, and knowledge evolution along with the scientific discourse. Rather, there is a strong need for more tailored guided approaches.

6.2 Approach: MORTY

Deep-learning language models have achieved strides on various natural language processing tasks, such as Question Answering [SS20], Text Summarization [Ma+21], and Machine Translation [Xia+19]. MORTY leverages the capabilities of deep learning language models to comprehend the semantics of scholarly text and perform targeted information extraction via text summarization. Scholarly articles typically follow a certain structure. IMRaD [SP04] refers to Introduction, Methods, Results, Discussion. The concept has been applied to abstracts for a high-level overview of the four essential aspects of the work. Structured abstracts [Nak+05] follows the IMRaD principles by including the same points in the abstract. This motivated us to incorporate structure into automatic textual summaries, which can be easily parsed for the sake of information extraction (a.k.a. structured summary).

Figure 6.2 depicts a high-level view of the MORTY approach to information extraction on scholarly articles comprising several workflow phases. It starts with pre-processing of the article text (i.e., the conversion from traditional PDF into text as well as cleaning and removing some needless segments of the text). A summarization model is then capable of rendering a large text snippet into a much

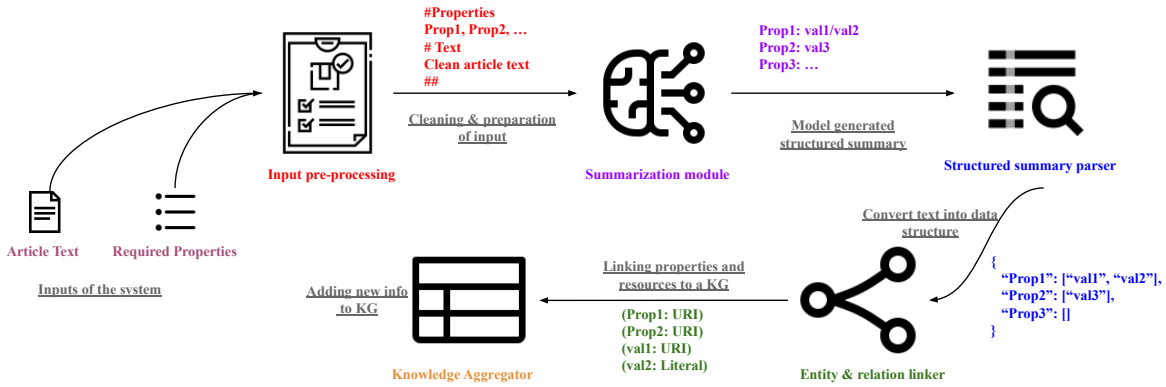


Figure 6.2: **Bird's eye view on the complete workflow of employing structured summarization** in the context of information extraction from scholarly documents and articles (MORTY).

shorter structured summarization that contains pairs of properties and their corresponding values. Later stages take care of parsing the produced summary via finding pre-defined syntactical patterns in the produced text. Then interlinking extracted values to knowledge graph entities via exact lookup functionalities. Lastly the newly extracted and aligned data gets added it to a destination knowledge graph. The fundamental component of the approach is the summarization module due to the fact that all other components of MORTY are self-consistent.

6.2.1 Formalization

One way of looking at MORTY is to see it as a pipeline of tasks each takes an input which is the output of a previous one. Formally we define the approach as a multi-layer transformation function $\Xi : T \rightarrow PV$, where T is the input text of an article, and PV is the set of property values that the transformation function finds. $\Xi(t)$ can be described as it follows:

$$\Xi(t) = \Gamma(\Pi(S(C(t)))) \tag{6.1}$$

whereas, t is the input text sample. $C(.)$ is a heuristic function that cleans up the input text and converts it to T'. The heuristics it follows are deterministic, which are a set of predefined roles and transformations. $S(.)$ is the core function in the transformation steps and it converts the clean text T' to structured summary $\bar{T}$.

$$\begin{aligned} C &: T \rightarrow T' \\ S &: T' \rightarrow \bar{T} \\ \Pi &: \bar{T} \rightarrow \tilde{PV} \\ \Gamma &: \tilde{PV} \rightarrow PV \end{aligned} \tag{6.2}$$

Moreover, $\Pi(.)$ is a simplistic syntax parsing function that converts the structured summary by finding pre-defined patterns into an unaligned property value set $\tilde{PV}$. Finally, the last transformation is to find exact mappings of extracted values into a destination knowledge graph, transforming the unaligned set into a fully aligned set of property value pairs PV.

The aligned set of property value pairs PV is defined as it follows:

$$PV = \{(p_1, v_1), (p_2, v_2), ..., (p_i, v_j)\} \tag{6.3}$$

whereas, p is the property, and v is a corresponding value. There could be i properties and j values, where $i \neq j$. A property p_i could have multiple values at any given times.

6.2.2 Transformation Example

In order to clarify what steps are taken in the transformation from plain text to needed information pieces that can be added to the knowledge graph. We will run through a real-life example from our dataset step by step. MORTY expects plain text as its input, alongside a set of properties that should be extracted from this text.

```
Automatic summarization techniques on meeting conversations developed so
 ↪ far have been primarily extractive [22], resulting in poor summaries.
 ↪ To improve this, Jang et al. [3] proposes an approach to generate
 ↪ abstractive summaries by fusing important content ......
```

Excerpt 6.1: **Snippet of plain text** from a research article.

First step, is to pre-process the plain text in order to remove a series of predefined patterns and constellations of terms. In this case, the citations are removed (both author names, and numerical references). The resulting text is exactly the same, minus all blacklist elements, like shown in Excerpt 6.2.

```
Automatic summarization techniques on meeting conversations developed so
 ↪ far have been primarily extractive, resulting in poor summaries. To
 ↪ improve this, proposes an approach to generate abstractive summaries
 ↪ by fusing important content ......
```

Excerpt 6.2: **Snippet of "clean" text** from a research article, pre-processed by one of MORTY's components.

Next step, is to convert the text and the list of requested properties into one piece of text to feed to the summarization module.

```
#Properties:
Summarization Type, Machine Learning Paradigm, dataset, Future work, Has
 ↪ preprocessing steps, Data Size ....
#Text:
Automatic summarization techniques on meeting conversations developed so
 ↪ far have been primarily extractive ....
```

Excerpt 6.3: **Snippet of summarization input text** generated by combining the clean text and the list of requested properties.

Excerpt 6.3 showcases how the two elements are merged into one coherent text, that is then feed into the summarizing model in order to get a structured summary. The structured summary is a way of combining semantically meaningful information in plain text by using certain patterns to distinguish various elements.

```
Summarization Type:: Abstractive, Machine Learning Paradigm::
 ↪ Unsupervised, Has preprocessing steps:: Topic segmentation/anaphora
 ↪ resolution/pronoun resolution/ambiguity resolver, ...
```

Excerpt 6.4: **MORTY's structured summary**. Which encodes all the extracted information within the resulting summary of the summarization model.

Excerpt 6.4 shows what a structured summary looks like. Certain syntactic patterns are used to separate elements and values. For instance, the double colon "::" separate the property label (left hand side) with the values (right hand side). The comma "," indicates that one property-values pair is done, and a new one is starting. Last but not least, the forward slash "/" separates multiple values per property. After this step, the structured summary is parsed by a simple syntactic parser with these simple rules to get a machine-readable and actionable structure, such as:

```
(Summarization Type, Abstractive)
(Machine Learning Paradigm, Unsupervised)
(Has preprocessing steps, Topic segmentation)
(Has preprocessing steps, naphora resolution)
......
```

Excerpt 6.5: **Parsed summary**. The result of parsing the structured summary by MORTY as pairs of (property, value).

Final transformation step involves the alignment of resources and properties extracted from the text to the knowledge graph. The most simple approach to do so, is to perform an exact lookup in the KG entities for each extracted value, and if found it can be replaced with it. Same goes for properties, however, since the properties are less (reused by nature) thus the properties are guaranteed to exist in the knowledge graph.

6.3 Evaluation

Since the main component of MORTY is the structured summary generation, **we focus our evaluation on that component solely**. Other components of the approach are deterministic in behavior and can be disregarded for the sake of this evaluation. We created a dataset using the ORKG infrastructure, and empirically evaluated the feasibility of the summarization task with various models and approaches based on this dataset.

Table 6.1: **Structured summarization dataset characteristics.** Showing the count of each distinct element in the dataset.

Concept	Count
Papers	485
Properties	844
Properties w/ multi value	391
Values	12,492
Distinct values	5,145
Total samples	539

6.3.1 Dataset Collection

We require a source for human-curated annotations of scholarly articles. ORKG is a knowledge graph that contains this sort of information. Hence, we leverage the ORKG to create a dataset of scholarly articles' texts with a set of property-value pairs. First, we took a snapshot of the ORKG data[2] and we filtered on papers that are open access or have pre-prints on arXiv. This ensures restriction-free access to the PDF files of articles. Second, we parse the PDF files using GROBID [Lop09] into text. Furthermore, we employ a heuristic to clean the text. The heuristic involves the following steps: i) Remove a set of pre-defined sections (such as abstract, related work, background, acknowledgments, and references); ii) Remove all URLs from the text, as well as all Unicode characters; iii) Remove tables, figures, footnotes, and citation texts. Lastly, we collect all annotations from the ORKG excluding some properties that contain values of URIs and other structural properties[3]. Afterwards that data was collected in a format that the summarization model is trained on and can process. We split the data in 80-10-10 training-validation-testing split in favor of the testing set. Table 6.1 shows important numbers about the dataset with regards to main concepts of the dataset that can affect the model being trained or the approach as a whole.

6.3.2 Baselines

Throughout the evaluation, multiple baselines were investigated (see Table 6.2). Various language models were used that are capable and pre-trained on summarization tasks. The maximum input size for each model varies depending on its architecture. In our created dataset, the average entry contained around 5K tokens, with a maximum around 9K and a minimum around 1.5K. Some of the models we used (e.g. Pegasus) are capable of abstractive summarization, i.e. can create summaries with words that don't exist in the original input text. This is important when annotated properties and values are not present in the text, but are formulated differently.

Furthermore, the feasibility of the task is evaluated using two other categories of NLP tasks. Extractive question answering (similarly to [YV14]) language models are leveraged to try to extract values for certain questions. The questions are formulated as follows: "what is the <property-label>?". This type of baselines is inherently flawed because some of the properties and values from the datasets are

[2] Data snapshot was taken on 02.02.2022.
[3] Properties that are used solely for information organization and have no semantic value.

Table 6.2: **Overview of models used in MORTY evaluation**, categorized per task. With the number of parameters, the max input size they can handle, and what dataset they are fine-tuned on beforehand to our training.

Model	# of Params	Input Size	Finetuned on
Summarization			
ProphetNet-large [Yan+20]	391M	2K	CNN
BART-large [Lew+19]	460M	4K	CNN
GPT2-large [Rad+19]	774M	2K	-
Pegasus-large [Zha+20b]	568M	2K	Pubmed
BigBird-large [Zah+20]	576M	4K	Pubmed
T5-large [Raf+19]	770M	4K	-
Longformer-large [BPC20]	459M	8K	Pubmed
Question Answering (QA)			
BERT-large [Dev+19]	335M	1K	SQuAD2
Longformer-large [BPC20]	459M	8K	SQuAD2
Named Entity Recognition (NER)			
BERT-large [Dev+19]	335M	1K	CoNLL
RoBERTa-large [Liu+19b]	355M	2K	CoNLL

Table 6.3: **Rouge F1 scores** for 1-gram, 2-grams, and longest-gram variations of the summarization models. Top best results are indicated in bold, second best in italic.

	Rouge-1	Rouge-2	Rouge-L
ProphetNet	31.1	12.5	23.7
BART	**36.7**	*22.0*	*29.4*
GPT2	16.1	3.6	9.3
Pegasus	27.1	11.7	21.2
BigBird	17.9	5.9	12.5
T5	12.2	2.8	7.9
Longformer	*34.7*	**22.4**	**29.6**

not as is in the input text. Another method for evaluation is to perform named entity recognition by recognizing the individual values as entities of interest and then classifying them into one of the classes (properties).

6.3.3 Evaluation Results and Discussion

The evaluation took place on a machine with 2 GPUs RTX A6000 each with a 48GB vRAM. Training scripts where adapted from the fine-tuning scripts of each of model's code repositories with the help of the Transformers [Wol+20] library. The training used a batch size = 2 and epochs = 20 with early stopping enabled.

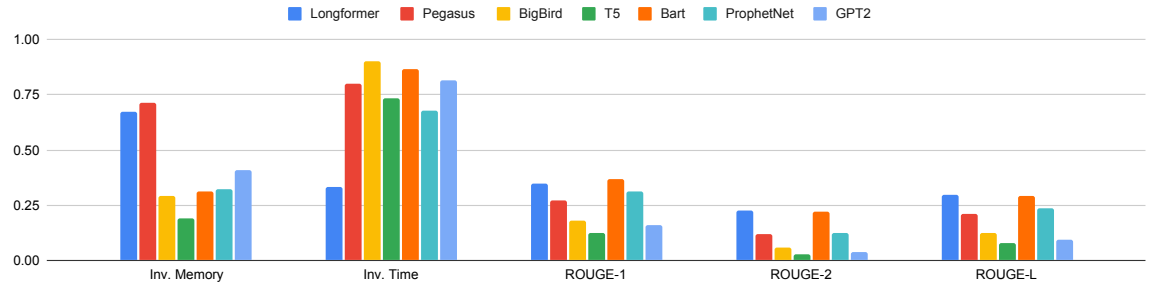


Figure 6.3: **Summarization metrics overview** of used models including the inverse time needed for training, and inverse memory consumption. Time and memory values are normalized and inversed. Higher values are better.

Table 6.4: **Precision, recall, f1-score results** of other baseline models on the question answering (QA) and named entity recognition (NER) tasks.

	Precision	**Recall**	**F1-score**
BERT (QA)	20.8	18.1	19.1
Longformer (QA)	23.7	22.8	23.2
BERT (NER)	17.2	17.0	17.0
RoBERTa (NER)	19.7	19.5	19.6

First, we evaluate the performance of various language models for structured summarization. Table 6.3 shows the results of the Rouge F1 metric (following [Zha+20b]) for all considered language models.

Second, we evaluate the feasibility of the task using techniques other than summarization, namely extractive question answering and using named entity recognition. Table 6.4 describes the performance of the two different approaches using two models for each case. Each model has different maximum input size and for the QA task the models were previously tuned on the SQuAD2 [RJL18] and the CoNLL [SD03] datasets for the NER task. For the QA metrics, the reported number are computed @1, meaning only candidate results at the first place.

The results show that the task is viable using summarization and that structured summaries are able to extract the required information out of the scholarly articles. When considering the normal summarization task, i.e., summarizing text into a coherent shorter text snippet, the top model [Pan+22] at the time of writing this article are performing with 51.05, 23.26, 46.47 for Rouge 1-2-L respectively[4]. This kind of summarization is far easier than structured summarization since the aim is merely coherent text creation, not structured summary of text fragments. Examining Table 6.3 and Table 6.2, we note that input size affects the performance of the model. The summarization model requires the processing of the complete input article text to extract values from it, and if the model can not handle the full article then it will suffer in performance metrics. We note that, BART and Longformer summarization

[4] https://paperswithcode.com/sota/text-summarization-on-pubmed-1

Table 6.5: **Examples**: Expected vs. model predicted values (for two summarization models).

Property	Expected	Longformer prediction	Pegasus prediction
Preprocessing steps	Topic segmentation Anaphora resolution Pronoun resolution	Anaphora resolution	Segmentation
Data size	139 meetings	20 meetings	20 meetings
Summarization type	Abstractive	Abstractive	Summarization
Evaluation metrics	ROUGE-2 ROUGE-SU4	F1	F1 SU4
Study location	Singapore	The City of Singapore	Middle

models performed best across all metrics. BART outperforms all other models for Rouge-1, and holds the second place for Rouge-2 and Rouge-L. Longformer is the best performing model for Rouge-2 and Rouge-L.

ProphetNet and Pegaus performed well compared to other models, but they were not able to beat Longformer in part due to limited maximum input size. GPT2 model suffered due to its nature as a generative model. It was not able to generate the structured summary rather generating more coherent text. Surprisingly, although big models with a 4K max input size BigBird and T5 did not perform comparatively to top models in the list.

Figure 6.3 depicts an overview of the Rouge metrics of the summarization models as well as time and space requirement of each. Though Longformer is the best performing model on average, it requires more time to train compared to BART. On the other hand, BART requires almost twice the memory compared to Longformer.

In order to empirically judge if the summarization method is suitable for the task of information extraction, we evaluate the approach against two categories of tasks: Question Answering (QA) and Named Entity Recognition (NER). Table 6.4 shows the precision, recall, and f1-score metrics for two models in each category. Due to the nature of the training data and the task itself, these two categories are inherently flawed because they are extractive and not abstractive, meaning that they aim at finding values from within the text, rather than compute with novel values. Thus, these tasks are only able to retrieve parts of the values that are in the text and the rest are unattainable to them. This explains why different models in both tasks preform poorly.

Table 6.5 shows some examples of five properties from three different articles with the expected values and the predicted values by two different summarization model. We observe that the models are able to extract partial values or similar values but with different wordings, as well as exact values, and completely different values. For instance, "Data size" is an annotation property, were the expected value is not in the text, rather it is a summation of other values. "Preprocessing steps" property aggregate values from multiple places in the text. The remarks made in this section answers our research question.

6.4 Usecase & Limitations

Targeted information extraction approaches like MORTY, are perfect candidates for integration within an automated approach to literature review completion usescase. When reviewing literature, researches create tabular comparison tables that describe common properties across publications and how they differ from each other. These common properties would serve as a guide MORTY's guided IE approach.

We envision the ORKG as a suitable host for such an application. The application would have to go thorough several steps and to involve the user (e.g., researcher) after the extraction of information. The workflow of this usecase would adhere to the following:

1. **Comparison Selection:** first step is to find a suitable comparison (literature review) that the user would like to auto fill. The selection of the source/destination comparison is out of the scope of the targeted IE task. However, it informs the approach with the set of properties that it would need to locate and extract from the input text.

2. **Recommend Suitable Articles:** next, based on the articles and the research problem of the literature review, a set of candidate articles should be recommended for the user to be added. The recommendation can be done via tailored tools, or generic similarity methods from existing corpora.

3. **PDF Upload:** after selecting the an article to be analyzed and used for extraction (this is on the user of the system and not automatic). A PDF - the most common format of scholarly publications - is then uploaded to the system. Due to the constraints that MORTY only accepts plain text as input, the PDF would have to be parsed by customized tools (e.g., GROBID) and produce the input as plain text.

4. **Running MORTY:** now that all the requirements to run the extraction approach are in place, the approach starts going through its phases one by one till the extraction of the information, as well as linking to KG entities.

5. **Human Intervention:** at the last stage, after the required information are located, extracted, and aligned. The user plays the role of the judge of what values and properties to add to the comparison. Other decisions like ambiguity in linking, or to choose to create new resources/properties lies solely on the user of the system.

For the recommendation of suitable articles (second step), we created a tailored approach of relying on the selected comparison and its abstracts. Since each comparison contains a list of papers and their corresponding properties and values. We collected the list of abstracts for each of those articles, and created a system capable of classifying whether an abstract belongs to the same cluster of other abstracts. i.e., whether a candidate paper (via its abstract) belong with the other compared papers (also via their collective abstracts) in the same cluster. To do so, we collected a list of all papers with their abstracts from the ORKG comparisons and trained a system on this task. The problem is modeled as a binary classification task, and the input to the model is formulated like this:

"[CLS] <*New Abs.*> [SEP] <*Abs. 1*> [SEP] ... <*Abs. n*> [SEP]"

We empirically evaluated the feasibility of this method and found that the classification model was able to perform well enough given a large dataset of similar papers along with their abstracts. The evaluation we performed resulted in 81.4 precision, 87.6 recall, and 84.3 f1-score.

MORTY's approach depends mainly on the capabilities of the automatic text summarization model it operates. We note here some of the observed and hypothesized limitation of the approach:

1. **Dataset distribution**, the bigger the dataset and the more representative it is, the better the model is able to learn the task.
2. **Pre-defined syntactic separators** of the structured summary, play a great role in how the model is able to condense the extracted information into a new representation. For instance, we use "comma" as a separator, which could be problematic if a value or a property label contain a comma.
3. **Abstractive vs Extractive**. Summarization models are usually trained to do one or the other of these summarization categories. If the values to be extracted are not existing in the text then abstractive is better, and vice versa.
4. **Pre-processing steps** affects greatly how the approach performs. In our approach we are using a set of heuristic patterns to be removed, which might remove required information.
5. **Document length** plays a crucial role with which summarization model it is used. Most of the language models accepts only small lengths, and scholarly articles exceed that length resulting in truncating or removing valuable content.
6. **Values that don't exist** at all in the text, cause an obvious issue of the approach. In an annotated dataset collected from the ORKG, we have statements that are reflected from the background knowledge or human intelligence and are not existing in the text. As such the model is not able to find such properties or values.

This is merely a listing of some of the major points that affect the performance and the overall capacity of the proposed approach of the chapter. We leave these points open for future research and investigation.

6.5 Summary

The objective of this work is to leverage structured summarization for the task of IE from scholarly articles. Automatic summarization is capable for condensing large amounts of text into a small snippet that contains only the important pieces. It is also better that other methods because it doesn't have to quote values from the text, rather generate completely new values. We evaluated various models on the summarization task, as well as compared against models performing question answering and named entity recognition. The results show that summarization is a viable and feasible approach for the IE task on scholarly articles. We also put MORTY in a bigger ageneda and describe how it should work in a use case of auto-completing literature reviews. Finally, we discuss the limitations of the approach and highlight open questions.

CHAPTER 7

Scholarly Knowledge Graph Completion

Researchers are confronted with an ever-increasing publication flood, deteriorating peer-review, and the reproducibility crisis. Thus, organizing scholarly knowledge is one of the most pressing tasks for solving current and upcoming societal challenges. Knowledge Graphs (KGs) are a perfect candidate for representing and managing scholarly knowledge. However, other than the typical usage of KGs for encyclopedic and common sense knowledge representation, managing scholarly knowledge is significantly more challenging due to the heterogeneity, in-homogeneity and evolution of scholarly communication [HTT+09].

Encyclopedic KGs (e.g., DBpedia [Aue+07] and Wikidata [VK14]) have become central sources of structured content in downstream information extraction tasks such as entity linking, relation extraction, and question answering [Zha+16; JSA20; Cui+17]. KGs store facts in the form of triples (*head entity*, *relation*, *tail entity*). For instance, in a triple <*Wuhan*, has Ro estimate (average), *2.68*>, where *Wuhan* is the head entity (*h* for short), *has Ro estimate (average)* is the relations (*r*), and *2.68* is the tail entity (*t*). Though large-scale knowledge graphs vary in terms of scope and coverage, KGs are often suffering from incompleteness and sparseness [Ji+16]. Therefore, KGs are required to be regularly updated and completed with missing information. The incompleteness of KGs motivates the knowledge graph completion task comprising several subtasks [YML19; CZC18]: i) *Head Prediction* (HP) aims to find missing head entities in triples. ii) *Relation Prediction* (RP) predicts missing relations between two entities. iii) *Tail Prediction* (TP) finds tails of partial triples comprising head entities and desired relations. In the literature, head/tail prediction is synonymous with link prediction [Sun+18]. KG completion methods calculate the plausibility of a triple via a scoring function to determine the validity of a knowledge graph triple [Lin+15; GBS19]. These approaches can be broadly categorized into Knowledge Graph Embeddings (KGE) and Language Models (LM) based models [Wan+17]. KGE techniques such as TransE [Bor+13] and ConvE [Det+18] learn entity and relation representation in a low dimensional vector space. However, due to their limitation of utilizing semantic information from text these methods produce different representations for the same entities in distinct triples [An+18; WL16]. Thus, LM techniques for KG completion emerged in an attempt to represent semantic information encoded in texts and produce contextualized embeddings [Dev+19; Bro+20; Cla+20].

The content of this chapter is based on publication [Jar+21b]. The results and contributions of this chapter aim to answer the following research question and tackles the subsequent research challenge.

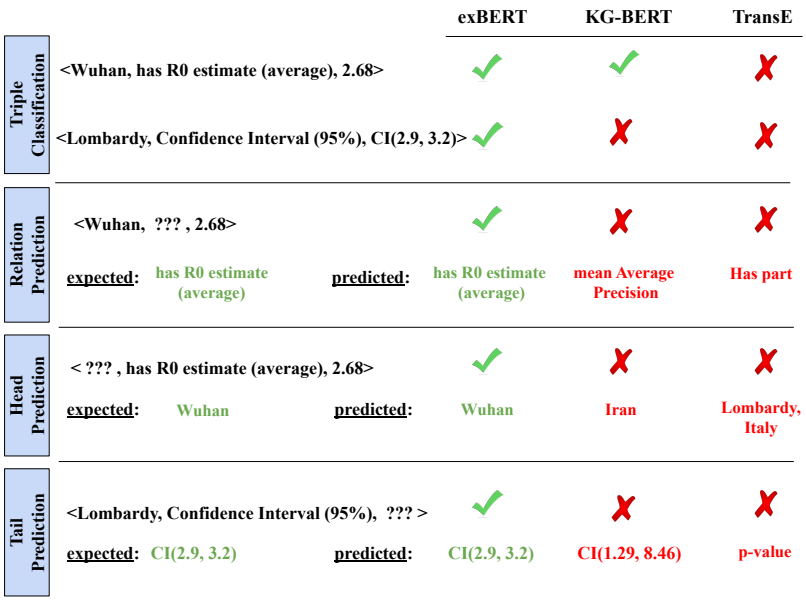


Figure 7.1: **Overview of KG-completion tasks with example results from our system exBERT** as well as the two baseline approaches KG-BERT [YML19] and TransE [Bor+13]. Green answers represent correctly extracted facts. Red answers indicate invalid facts. Entities and relations are part of a KG. These fact triples are extracted from scientific literature summarizing basic reproduction numbers [Del+19] of COVID-19.

RQ4: What is the impact of task-specific context on scholarly KG completion?

RC3: Automatic Population of Scholarly Knowledge Graphs.

To answer this research we propose exBERT, a system that uses triple classification to perform knowledge graph completion. In the following we summarize our key contributions of this chapter:

- exBERT a system that performs scholarly knowledge graph completion task, including the subtasks of triple classification, relation prediction, and link prediction.
- Extensive evaluation of several datasets against various baselines to show that our method achieves state-of-the-art results for the respective tasks. Our proposed datasets, code, and empirical results are publicly available[1].
- We release two publicly available scholarly datasets curated from scholarly KGs for the KG completion task.

The rest of the chapter is structured as it follows: Section 7.1 motivates the problem of scholarly knowledge graph completion. In Section 7.2 we describe the approach we take to tackle the issue. Section 7.3 presents an extensive empirically evaluation of approach against various baselines on three different datasets, and discusses the results. Finally, Section 7.4 concludes the work.

[1] https://github.com/YaserJaradeh/exBERT

7.1 Motivating Example

The proliferation of scientific literature creates new challenges in the research community, such as reproducibility crisis, duplication, inefficiency, and a lack of transparency [Whi+17; Bak16]. To mitigate such challenges and to adhere to the FAIR data principles [Wil+16a], researchers have shifted focus to build several scholarly KGs such as the Open Research Knowledge Graph [Jar+19b], MAKG [Fär19], and SciKGraph [TR21]. Figure 7.1 illustrates scholarly KG completion tasks using three systems on a few concrete examples. Scholarly KGs differ from encyclopedic purpose KGs (e.g., DBpedia and Wikidata) because the encoded information is derived from the scholarly literature [Des+20]. The scholarly KGs are ordinarily sparser than generic KGs because structured scholarly knowledge is more laborious to extract [Jar+19b]. Furthermore, these KGs are ambiguous due to a lack of standard terminology used across the literature and poses domain-specific challenges for KG completion task [Jar+19b]. Due to domain-specific challenges, the performance of existing KG completion methods such as KG-BERT and TransE [YML19] is limited when applied to scholarly KG completion tasks (cf. Figure 7.1 and Section 7.3). The observed behavior is not surprising and is due to the peculiar entity and relation types of scholarly KGs. For example, background knowledge is required to understand entities and relations of scholarly KGs in Figure 7.1. Hence, we suggest that existing KG completion approaches require additional task-specific context for the scholarly domain. We focus on the task of KG completion for the scholarly domain. Inspired by recent advancements in contextualizing language models [Mul+20; SK21], we argue that LMs can be utilized for scholarly KG completion tasks if fed with context derived from the scholarly literature. Our rationale for the choice is as follows: Language models such as SciBERT [BLC19] are already trained in an unsupervised manner on large corpora of scholarly literature. Hence, adapting SciBERT for KG completion will allow us to inherit task-specific context.

We model KG completion as a sequence classification problem by treating triples of the scholarly KG as sequences of natural language text. We propose exBERT, a contextualized approach for scholarly KG completion. In exBERT, we build on SciBERT [BLC19] and adapt the underlying pre-trained language model on these sequences for predicting the plausibility of a triple, relation, or an entity. Our empirical results on the standard scholarly KG completion datasets provide superior performance on several KG completion tasks against baseline approaches.

7.2 Approach

We model the KG completion task as a sequence classification task to harness the richness and power of transformer language models. We rely on a BERT model to perform the classification via transforming the input knowledge graph triples into sequences of text with some extra tokens following the convention of BERT fine-tuning. Furthermore, we leverage a SciBERT model rather than a plain BERT as our core transformer model for the scholarly context.

7.2.1 Language Model for Scientific Text

SciBERT [BLC19] is a state-of-the-art pre-trained language model that creates contextual representations using a multi-layer bidirectional transformer encoder architecture described by Devlin et al. [Dev+19]. SciBERT builds on the BERT [Dev+19] model and further trains it on scientific literature using 1.14 million scientific papers from Semantic Scholar [Amm+18]. The model also consists of 3.17 billion tokens (i.e., words). SciBERT is pre-trained on two tasks: masked language modeling and next sentence

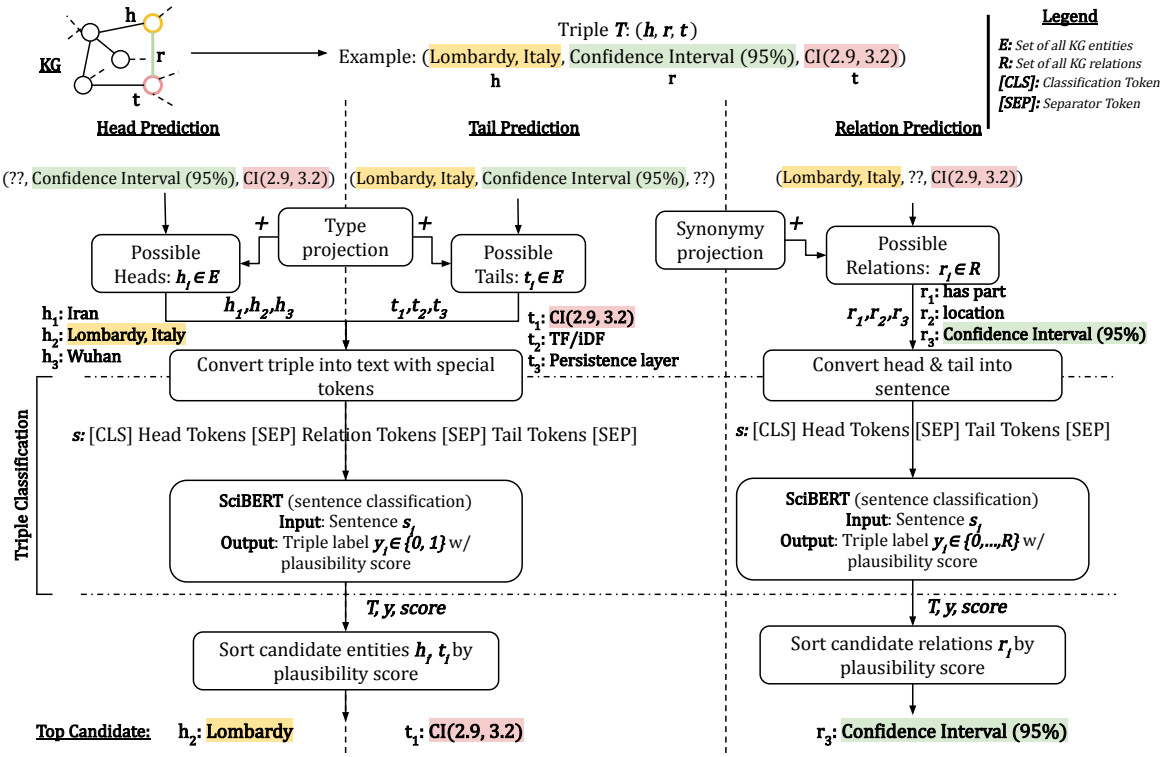


Figure 7.2: **Illustration of triple classification with ExBERT along with KG completion sub-tasks**. The figure shows the fine-tuning process that allows ExBERT to predict the plausibility score of a triple. In head/tail prediction tasks, the missing entity is substituted with all other entities in the KG, and subsequently, all plausibility scores (a.k.a. confidence scores) are estimated and sorted to provide the top candidates. In relation prediction, the relations are used as classes and a triple viewed in the form of a sentence is classified according to them.

prediction. For next sentence prediction, the language model predicts whether two sequences of input (i.e., sentences) are consecutive in the text. For masked language prediction, SciBERT predicts suitably masked input tokens. Furthermore, regarding the fine-tuning phase, SciBERT is initialized with the weights and the parameters from the pre-training phase. Thus, these parameters are fine-tuned using labeled data from downstream natural language processing tasks (e.g., question answering, summarization, and token classification) [Zha+20a].

7.2.2 Scholarly Model (ExBERT)

Building on what KG-BERT [YML19] proposed, and to take full advantage of contextual representation with rich language patterns, we use a pre-trained SciBERT transformer model to perform the task of knowledge graph completion. SciBERT contains rich scholarly context as it has been trained on sizeable scientific literature. We employ a SciBERT as our base model to take advantage of the scholarly context. In ExBERT, we represent entities and relations of the knowledge graphs using their respective text labels. Furthermore, these label sequences are given as an input sentence to our model for further

fine-tuning. In order to estimate the plausibility score of triples, we arranged the sentences of (*h*,*r*,*t*) as individual sequences. In our case, a sequence represents a BERT compatible token sequence combined from two entities (*head* and *tail*) or a complete triple (*head*, *relation*, *tail*).

The workflow of exBERT is shown in Figure 7.2. exBERT performs a triple classification task to determine if a triple belongs to a KG. Furthermore, other tasks can be performed by leveraging the triple classification task, e.g., head/tail or relation prediction. The first token of every input sequence for exBERT is always a unique classification token `[CLS]`. Each entity in the triple and the relation are represented as a sentence containing a list of tokens $v_1^{\{h,r,t\}}, ..., v_\alpha^{\{h,r,t\}}$, $\alpha \geq 1$. For instance, *Lombardy Italy has a confidence interval (95%) of CI(2.9, 3.2)* has a head entity *Lombardy, Italy* which comprises two tokens v_1^h = Lombardy, v_2^h = Italy; the relation *Confidence Interval (95%)* comprises three tokens v_1^r, v_2^r, v_3^r; and the tail *CI(2.9, 3.2)* comprises two tokens v_1^t, v_2^t. While constructing the sentences of entities and relations, a special token `[SEP]` is used to differentiate elements (i.e., components of the triple). The separation token indicates that the various elements of the sentence have different segment embeddings. However, the tokens of the head and tail entities share identical segment embeddings. Furthermore, when various tokens occupy the same position – for the case of SciBERT, a position $1 \leq i \leq 512$ – they have the same position embeddings.

Token sequences are used as an input to the SciBERT model architecture, which is a multi-layer bidirectional transformer encoder based on the native BERT architecture [Dev+19]. The final hidden vector of the special `[CLS]` token is denoted as $C \in \mathbb{R}^H$, where H is the hidden state size in the pre-trained SciBERT model. Furthermore, the i-th token of the model's input tokens is referred to as $v_i \in \mathbb{R}^H$. When fine-tuning the model to perform the triple classification task, a set of weights is created (i.e., classification layer weights) $W \in \mathbb{R}^{2H}$. A sigmoid function Ω is used to score a triple $\Upsilon = (h, r, t)$ and produce its class affiliation. Equation 7.1 shows the scoring function of a triple Υ. Where $\Omega^\Upsilon \in \mathbb{R}^2$ is a two-dimensional real-valued vector with $\Omega_0^\Upsilon, \Omega_1^\Upsilon \in [0, 1]$ and $\Omega_0^\Upsilon + \Omega_1^\Upsilon = 1$.

$$\Omega^\Upsilon = sigmoid(CW^v) \tag{7.1}$$

We are then able to compute the cross-entropy loss using the scoring function Ω^Υ and predicted labels y^Υ as follows:

$$\Phi = -\sum_{\Upsilon \in \mathbb{K}^+ \cup \mathbb{K}^-} \left(y^\Upsilon log(\Omega_0^\Upsilon) + (1 - y^\Upsilon) log(\Omega_1^\Upsilon) \right) \tag{7.2}$$

where $\mathbb{K}^+$ is the positive triple set and $\mathbb{K}^-$ is the negative triple set. y^Υ is the label of the triple (i.e., positive or negative label) $y^\Upsilon \in \{0, 1\}$. The positive triple set contains the correct triples within the KG. However, the negative set $\mathbb{K}^-$ is constructed by replacing the head entity h or tail entity t in a positive triple with a random entity $\bar{h}$ or $\bar{t}$, as indicated in Equation 7.3.

$$\mathbb{K}^- = \left\{ \mathbb{K}_{\bar{h}}^- \cup \mathbb{K}_{\bar{t}}^- \right\} \tag{7.3}$$

$$\mathbb{K}_{\bar{h}}^- = \left(\bar{h}, r, t \right) | \bar{h} \in \mathbb{E} \wedge \bar{h} \neq h \wedge \left(\bar{h}, r, t \right) \notin \mathbb{K}^+ \tag{7.4}$$

$$\mathbb{K}_{\bar{t}}^- = \left(h, r, \bar{t} \right) | \bar{t} \in \mathbb{E} \wedge \bar{t} \neq t \wedge \left(h, r, \bar{t} \right) \notin \mathbb{K}^+ \tag{7.5}$$

In Equation 7.4 and Equation 7.5, $\mathbb{E}$ is the set of KG entities. When corrupting the triple (i.e., creating the negative set), we make sure that the head or tail being replaced is not the correct one and ensure that

the resulting corrupt triple does not belong to the positive set. Both relation and link prediction tasks use triple classification as an underlying task. The difference is in the way these tasks compose the input sentences (cf. Figure 7.2 "Convert triple into text with special tokens" steps). Based on the findings of Yao, Mao, and Luo [YML19], we compose the input sequence with the two entities h and t to predict a relation r. KG-BERT empirical results suggest that predicting relations from the two entities using triple classification has higher performance. The other setting involving complete triples by curating negative samples with random relations $\bar{r}$ does not yield a performance gain. Similarly to link prediction, the final hidden state vector C corresponds to the special classification token `[CLS]`. We highlight that relation prediction exBERT differs from link prediction exBERT (i.e., head and tail prediction) with the classification layer weights. For relation prediction, the tasks fine-tune the weights $\bar{W} \in \mathbb{R}^{R \times H}$, whereby R is the number of all KG relations. The scoring function is $\bar{\Omega}^{\Upsilon} = softmax(C\bar{W}^{\nu})$, whereby $\bar{\Omega}^{\Upsilon} \in \mathbb{R}^{R}$ is a $\mathbf{R}$-dimensional real vector with $\bar{\Omega}_i^{\Upsilon} \in [0, 1]$ and $\sum_i^R \bar{\Omega}_i^{\Upsilon} = 1$. Similarly, we compute the cross-entropy loss with the help of the scoring function $\bar{\Omega}^{\Upsilon}$ and the relation labels $\bar{y}^{\Upsilon}$ as shown in Equation 7.6.

$$\boldsymbol{\Phi}_r = -\sum_{\Upsilon \in \mathbb{K}^+} \sum_{i=1}^{R} \bar{y}_i^{\Upsilon} log(\bar{\Omega}_i^{\Upsilon}) \tag{7.6}$$

where $\bar{y}_i^{\Upsilon}$ is the relation class (i.e., indicator) for a positive triple Υ and its value is conditional on the relation as described in Equation 7.7.

$$\bar{y}_i^{\Upsilon} = \begin{cases} 1 & \text{if } r = i \\ 0 & \text{o/w} \end{cases} \tag{7.7}$$

7.3 Evaluation

We conduct our experiments and analysis in response to the overall research question **RQ4**. As such, we also compare exBERT against approaches not in the scholarly context. To understand the efficacy of exBERT for scholarly KG completion, we further divide our overall research question into three sub-questions:

- RQ4.1: What is the performance of exBERT for scholarly relation prediction?
- RQ4.2: What is the performance of exBERT for link prediction in scholarly KGs?
- RQ4.3: What is the efficacy of exBERT in scholarly triple classification?

7.3.1 Datasets

The scholarly KG domain is relatively new. Hence, there is a scarcity of standard datasets to benchmark the performance of KG completion methods. With our focus on scholarly domain data and knowledge, the need arises for domain-specific datasets to benchmark the performance of language models and other systems. Therefore, we created two datasets collected from available knowledge graphs and online resources.

Table 7.1: **Summary statistics of datasets and the percentage of each type of relations**. We propose two new datasets (labeled with [*]) for the scholarly KG completion task.

Dataset	# Ent	# Rel	# Triples	# Train	# Dev	# Test	% N to N	% N to 1	% 1 to N	% 1 to 1
*ORKG21**	226,210	2634	249,682	149,808	49,937	49,937	11.80	6.92	12.36	68.92
*PWC21**	192,115	26	284,875	170,925	56,975	56,975	19.23	15.38	19.23	46.16
UMLS	135	46	6,529	5,216	652	661	56.52	6.53	23.91	13.04

- **ORKG21**: A dataset of scholarly contributions extracted from the ORKG infrastructure [Jar+19b]. The ORKG contains data about scientific contributions and publications. The data is (semi-)automatically curated. The ORKG provides a dump of data in RDF [MMM+04] format, which we used to create the dataset. The ORKG dataset has many relations because they rely on crowdsourcing input and do not automate the end-to-end extraction. Many relations are created to represent similar intent, resulting in a significant number of relations.

- **PWC21**: A dataset from the online resource Papers-with-Code [AI21] that describes papers in the field of machine learning, information extraction, and NLP along with their evaluation results. PWC data is represented in tabular rather than graph structure. We transformed the raw data into RDF for broader use. This dataset contains only a small set of relations because it focuses on certain aspects of research papers (i.e., evaluation results and metrics).

Since both datasets contain literals, we transformed them into entities by creating sequential ids in the form of "/literal_**num**", similar to the Yago3-10 dataset [Sun+18]. Additionally, we also used the following public scholarly dataset:

- **UMLS** [Bod04]: "Unified Medical Language System" is an ontology for the medical field that describes technical medical concepts and their interlinked relations.

Table 7.1 provides a summary of the employed datasets. If a dataset does not provide a class label (i.e., binary class) for training/testing triples, throughout our evaluation we considered every triple in the dataset as ground truth and we corrupted triples to generate negative samples as explained in Section 7.2.

7.3.2 Experimental Settings & Baselines

In this section, we list the baselines and other experimental details.

Baselines. To assess efficacy w.r.t. the tasks, we compare ExBERT with various state-of-the-art knowledge graph embedding methods, specifically:

- TransE [Bor+13] and its extensions TransH [Wan+14b], TransD [Ji+15], TransR [Lin+15], TransG [XHZ16], and DistMult [Yan+14], which only rely on structural information of the knowledge graph to compute the embeddings.

- The neural tensor network NTN [Soc+13] and its simplified version ProjE [SW17].

- CNN models, specifically ConvKB [Ngu+18] and ConvE [Det+18].

- Other knowledge graph embedding models, specifically: AATE [An+18] and TEKE [WL16] that leverage textual information; Contextualized graph embeddings with DOLORES [WKW18] and KG2E [He+15]; Complex-values KG embeddings techniques RotatE [Sun+18] and ComplEX [Tro+16]; Probabilistic embeddings with prior KG knowledge ERMLP [Don+14]; compositional vector-space embeddings HolE [NRP16]; Learning embeddings dependently with SimplE [KP18]; Hyperbolic embeddings to represent hierarchical data MuRE [BAH19b]; Tucker decomposition of the binary tensor representation of knowledge graph triples using TuckER [BAH19a]; BERT based approach KG-BERT by Yao, Mao, and Luo [YML19] which utilizes a BERT-Base model to perform KG completion.

We note that the baselines vary depending on the task considered because some of the KGE models do not perform tasks such as relation prediction.

Experimental Settings. SciBERT is the base transformer model used in exBERT. It has 12 layers, 12 self-attention heads, and $H = 768$ hidden layers. We used the Adam implementation for the optimizer. For fine-tuning the triple classification task, the batch size is 32, with a learning rate $5e-5$ and a dropout rate of 0.1. For triple classification, we sample one negative triple for a positive triple to assure class balance for binary classification tasks. Furthermore, the number of epochs for triple classification is 3. We found no discernible improvement by increasing this number. For the link prediction task, we used 5 epochs and we also sampled 5 negative samples for each positive triple (following the findings of Yao, Mao, and Luo [YML19]). Finally, for relation prediction the number of epochs chosen was 20.

For benchmarking exBERT and KG-BERT, we used a system running Ubuntu 20.04 with 128GB of RAM and 4x Nvidia A100 GPUs, each with 40GB vRAM. Other knowledge graph embeddings baselines are trained on 8x Nvidia RTX 3090 GPUs, each with 24GB vRAM. KG embeddings are ran using the benchmarking framework from PyKeen [Ali+21].

Graph embeddings are trained using stochastic local closed-world assumption. The TransE family is optimized using an SGD [Ket17] (Stochastic Gradient Descent) while other KGEs are optimized with an Adam [KB17] implementation. Lastly, the learned knowledge graph embedding has a dimension size of 200.

Evaluation Metrics. Following the widely adapted metrics and inheriting evaluation settings from KG-BERT [YML19], we report the Mean Rank (MR) and the cut-off hit ratio (Hits@N) metrics on all the datasets for the link prediction ($N = 10$) and the relation prediction ($N = 1$) tasks. MR reports the average rank of all correct entities. Hits@N evaluates the ratio of correct entity predictions at a top N predicted results. Similar to KG-BERT, for the triple classification task we report accuracy. Classification accuracy is the number of correctly classified triples of the testing set divided by the total number of test triples. For the relation prediction task, we rank candidate relations by the scoring function $f(h, r, t)$ (cf. $\bar{\Omega}^{\Upsilon}$ in our approach formalization). Each correct test triple (h, r, t) is corrupted by replacing the relation with every other relation in the KG, with the exception of the relation itself, i.e., $r' \in R | r' \neq r$. Then these candidates are ranked in descending order by their plausibility score we obtain from the triple classification task. Following Yao, Mao, and Luo [YML19] and Bordes et al. [Bor+13] we report the results under the filtered settings only, which means that all corrupted triples are removed from training, development, and testing sets before getting the ranking lists. In the link prediction task, we aim to predict the missing entity of the triple. Each correct test triple $T = (h, r, t)$ is corrupted by replacing either the head entity or the tail entity with every other entity in the KG

$e \in E | e \neq h \wedge e \neq t$. For bookmarking the results we report two metrics, Mean Rank (MR) of the correct entities and Hits@10 metric, which is the proportion of correct entities in the top 10 candidate results. As in relation prediction, lower MR is better as well as a high Hits@10 values.

7.3.3 Experiment 1: Relation Prediction

The task aims to find relation r given two entities $(h, ?, t)$. where ? refers to the missing relation in the triple. In this task we get a set of candidate relations and we rank each one by the scoring function $f(h, r, t)$ (cf. $\bar{\Omega}^{\Upsilon}$ in our approach formalization). Each correct test triple (h, r, t) is corrupted by replacing the relation with every other relation in the KG $r' \in R | r' \neq r$. Then these candidates are ranked in descending order by their plausibility score we obtain from the triple classification task. We measure two metrics in this task, the Mean Rank (MR) of correct relations and Hits@1, In these settings, a lower MR indicates a better result while a higher Hits@1 is better. Following Yao, Mao, and Luo [YML19] and Bordes et al. [Bor+13] we report the results under the filtered settings only, which means that all corrupted triples are removed from training, development, and testing sets before getting the ranking lists. Table 7.2 presents relation prediction results on all datasets. Structural embeddings such as TransE, TransH, and TransR report limited performance due to the lack of contextualized information. KG2E and NTN perform slightly better than TransE and extensions with their density-based embeddings and neural model, respectively. PairRE outperforms the other embedding-based models with its paired encoding of relations and its capability of encoding various types of relations (e.g., symmetric, inverse). A common downfall of embedding-based techniques is that they learn identical embedding representations of entities and relations and do not account for the different meanings that words might have in various contexts. The limited performance in the scholarly domain clearly validates the observation. In contrary, KG-BERT and exBERT significantly outperform embedding-based models due to their own contextualized embeddings learned using a large corpus of unstructured text. However, task-specific context enhanced the ability of exBERT in predicting scholarly relations compared to KG-BERT, which successfully answers the first research sub-question **RQ4.1**.

7.3.4 Experiment 2: Link Prediction

We report link prediction results in Table 7.3. Across all datasets, exBERT achieves significantly higher performance compared to all baselines. On the ORKG21 dataset, the majority of relations are symmetric relations. Translation-based methods report limited performance on ORKG21 due to its inability to infer symmetric connectivity patterns of a KG. Furthermore, the TransX family uses only the structure of the KG and does not induce context information or labels of entities, which results in limited performance. On PWC21, connectivity patterns are evenly distributed across all relation types (one-to-one, many-to-many, etc.), and the performance of exBERT does not drop. The majority of entities contain multiple relations on the UMLS dataset, and exBERT can successfully predict the missing links while reporting a slightly lower performance compared to best results. Embedding-based models continued to show limited performance for link prediction tasks due to non-standard characteristics of scholarly entities. From our empirical results, we conclude that the task-specific context fed into exBERT for the scholarly domain has positively impacted the performance across all KG completion tasks (successfully answering **RQ4.2**).

Table 7.2: **Relation prediction results on all three datasets**. Best results are indicated in bold font. The results listed in the table were all obtained by us. Techniques marked by [*] do not perform relation prediction by default. Hence, we employ triple classification to measure performance for relation prediction.

Method	ORKG21		PWC21		UMLS	
	MR	Hits@1	MR	Hits@1	MR	Hits@1
TransE [Bor+13]	1683	50.1	1232	49.5	166	23.3
TransH [Wan+14b]	1544	51.8	1169	51.1	141	25.7
TransR [Lin+15]	1378	53.3	897	54.4	78	27.3
NTN* [Soc+13]	1189	58.4	754	59.2	22	29.9
KG2E* [He+15]	1301	57.9	773	58.3	21	30.0
ProjE (pointwise) [SW17]	522	74.6	212	78.5	6.42	41.8
ProjE (listwise) [SW17]	503	75.5	198	79.1	6.31	41.8
PairRE [Cha+21]	206	82.1	59	88.6	4.25	63.5
KG-BERT [YML19]	15.37	92.8	1.51	96.7	1.21	87.2
exBERT	**12.98**	**95.5**	**1.02**	**98.3**	**1.11**	**88.8**

Table 7.3: **Link prediction results on all three datasets**. Best results are indicated in bold font. Results accompanied by asterisk* are reported by Yao et al. [YML19], other results were obtained by us and not from respective publications.

Method	ORKG21		PWC21		UMLS	
	MR	Hits@10	MR	Hits@10	MR	Hits@10
TransE [Bor+13]	2879	51.2	3176	60.8	1.84*	89.9*
TransH [Wan+14b]	2811	52.5	2994	61.3	1.80*	**99.5***
TransR [Lin+15]	2789	53.3	2642	65.6	1.81*	99.4*
TransD [Ji+15]	2791	53.2	2135	68.8	1.71*	99.3*
DistMult [Yan+14]	3321	50.0	3082	61.1	5.52*	84.6*
KG2E [He+15]	1352	59.6	2209	68.7	-	-
ConvKB [Ngu+18]	216	70.1	388	72.9	-	-
ComplEX [Tro+16]	713	65.3	456	72.7	2.59*	96.7*
HolE [NRP16]	98	73.7	97	76.4	-	-
RotatE [Sun+18]	47	76.9	61	77.5	-	-
CompGCN [Vas+20]	2.84	84.6	4.02	82.7	-	-
SimplE [KP18]	3.40	82.4	3.91	82.9	-	-
KG-BERT [YML19]	2.03	86.1	4.03	82.7	**1.47***	99.0*
exBERT	**1.80**	**87.4**	**2.11**	**84.2**	1.97	98.9

7.3.5 Experiment 3: Triple Classification

The objective of triple classification is to assign a score to each triple $T = (h, r, t)$ depending on whether or not it belongs to the underlying KG. Table 7.4 shows the classification results of exBERT against all other baselines on the benchmark datasets. exBERT outperforms all baseline KGE approaches confirming the effectiveness of our approach for the scholarly domain. We note that translation-based KGE such as TranE could not achieve high scores because of the cardinality of the knowledge graph relations (i.e., the existence of one-to-many, many-to-one, and many-to-many relations). However, its extensions, TransH, TransR, TransD, and TransG outperform TransE by introducing relation-specific parameters. The DistMult model performs relatively better than the Translation family. The CNN models, e.g., ConvKB, perform well, suggesting that a convolution model can capture global correlations among entities and relations in the scholarly domain. HolE performs proportionately well and comparably to ConvKB. SimplE achieves the best results compared to the other embedding-based approaches. Finally, exBERT outperforms KG-BERT leveraging the underlying semantic and contextualized knowledge of a transformer language model trained on scientific data.

Based on our observations, we identify the following reasons for the superior performance of exBERT for the triple classification task:

- Self-attention mechanisms employed by SciBERT can discover the essential tokens in a converted triple (especially in a scholarly context).
- The triple classification task is akin to the next sentence prediction, for which SciBERT is pre-trained with large text corpora. The fine-tuning weights are already positioned for the inference of correlation among triple components.
- Contextualized embeddings for the scholarly domain are explicit with our approach via the hidden token vectors. This has positively impacted the overall performance (successfully answering **RQ4.3**, and collectively answering **RQ4**).

7.3.6 Ablation Studies

Visualizing attention for the Triple Classification task To understand the impact of task-related context on triple classification, we leverage the attention visualization tool by Vig [Vig19] to visualize the attention in exBERT. Figure 7.3 illustrates the attention pattern at layer 11 of exBERT. To create the sequence, the triple components were joined with the `[SEP]` token. For the classification task, we prepend the special token `[CLS]`. We note that some special words such as *Wuhan* and the numbers (i.e., *2.68*) are indicated to be important by exBERT for performing the classification task. We also note that the separator tokens are highlighted with attention heads than other important words in the triple.

Impact of relation type on the Link Prediction task. For each dataset, we show the metrics (Mean reciprocal rank (MRR) because it is in range [0,1] and Hits@10) for exBERT and the top-performing baselines for various relation types (i.e., one-to-one, one-to-many, many-to-one, many-to-many). Figure 7.5 presents the performance of this task for every relation type and all datasets. Based on the fine-grained analysis, we suggest that:

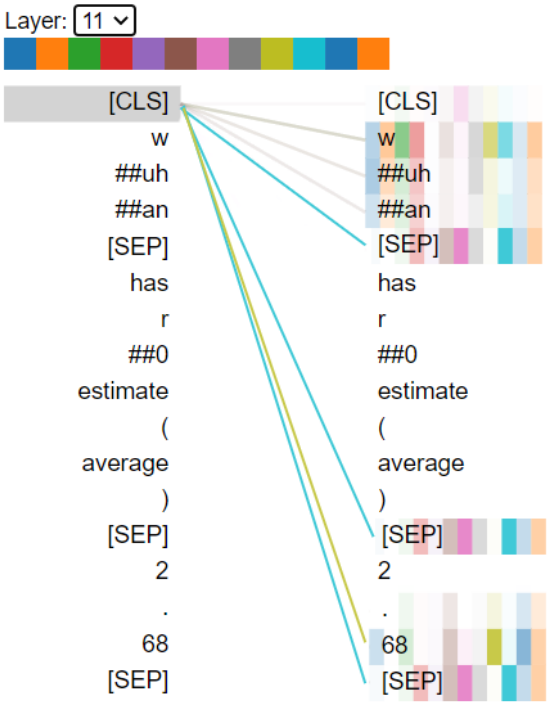


Figure 7.3: **Illustrations of attention pattern of exBERT**. A triple from the ORKG dataset is used as an example for the triple classification task. Different colors mean different attention heads. We highlight the attention weights between the special classification token `[CLS]` and other tokens in layer 11 of the language model.

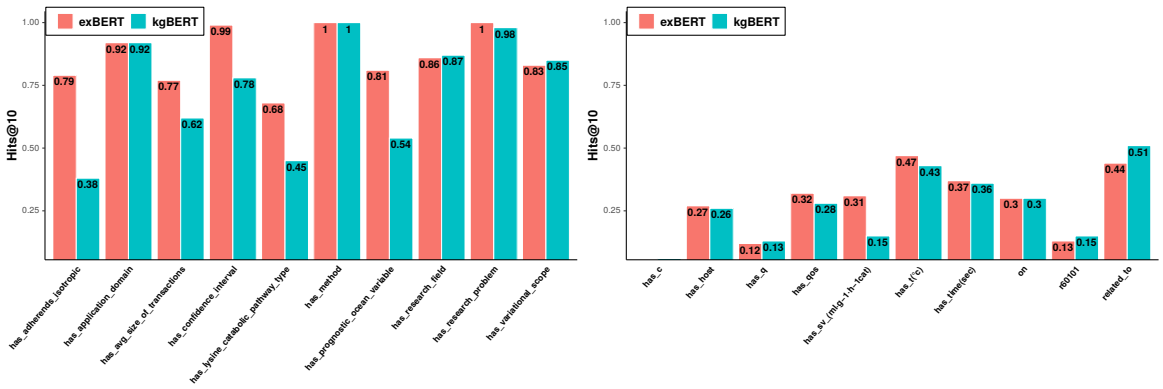


Figure 7.4: **Illustration of the Hits@10 metric** for the ten best (left plot) and worst (right plot) predicted relations in the ORKG21 dataset by exBERT and the second best baseline KG-BERT.

Table 7.4: **Triple classification accuracy (in percent) for different embedding methods**. The listed results were obtained by us and are not from the corresponding publications. Best results are indicated with bold font.

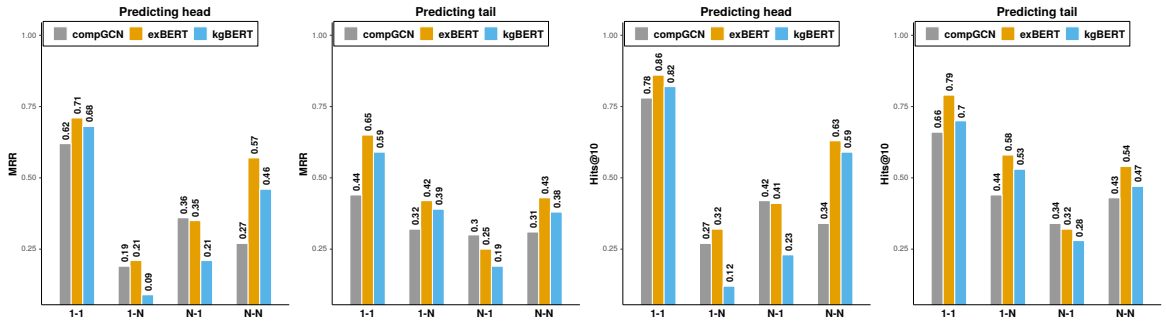
Method	ORKG21	PWC21	UMLS	Avg.
TransE [Bor+13]	77.6	77.3	78.1	77.7
TransH [Wan+14b]	78.8	77.9	79.2	78.7
TransR [Lin+15]	81.4	81.3	81.9	81.5
TransD [Ji+15]	84.3	83.6	84.9	84.2
TransG [XHZ16]	85.1	84.7	85.2	85.0
TEKE [WL16]	84.8	84.2	84.9	84.6
KG2E [He+15]	79.6	78.8	79.7	79.4
DistMult [Yan+14]	86.2	86.2	86.8	86.4
ERMLP [Don+14]	87.4	86.5	88.3	87.4
AATE [An+18]	87.9	87.7	88.4	88.0
ConvKB [Ngu+18]	87.3	86.8	83.1	85.7
DOLORES [WKW18]	87.6	87.3	85.4	86.8
HolE [NRP16]	87.3	87.3	88.2	87.6
NTN [Soc+13]	85.0	84.9	85.1	85.0
SimplE [KP18]	89.7	89.2	89.1	89.3
KG-BERT [YML19]	95.1	93.3	89.7	92.7
exBERT	**97.1**	**96.0**	**90.3**	**94.5**

- On the ORKG21 dataset, exBERT maintains superior performance across all relation types for both metrics. CompGCN performs slightly better than KG-BERT and exBERT on many-to-one relation types. One possible reason is that CompGCN jointly embeds nodes and relations in a graph that permits the model to handle dense relations [Vas+20].

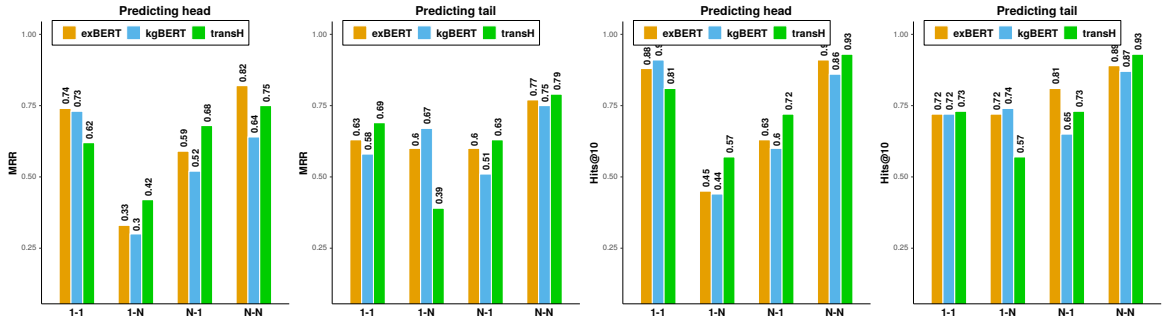
- On the UMLS dataset, exBERT suffers a performance drop in the head prediction that results in a lower performance observed in Table 7.3.

- On the PWC21 dataset, it is interesting to observe that all three models suffer significant performance drops for predicting head entities in the one-to-many relation category. However, models maintain steady performance across all other relation categories.

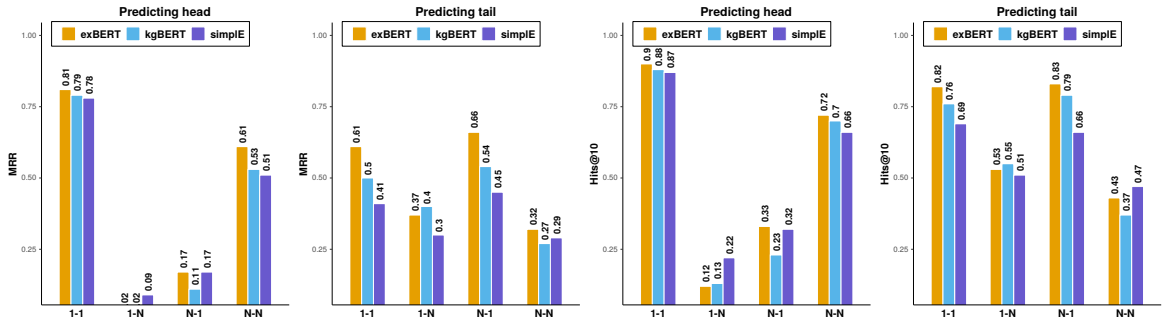
Relation Prediction results for best/worst relations. To further comprehend how relations affect the overall performance of the relation prediction task, we select the best/worst ten performing relations in the ORKG21 dataset. We plot the evaluation metrics graphs for exBERT compared to the second best baseline KG-BERT. Figure 7.4 illustrates the individual performance. KG-BERT performs comparably to exBERT for the generic relation types such as *has_method*. However, for peculiar scholarly relations such as *has_adrehends_isotrophic* and *has_prognostic_ocean_variable*, due to missing task-specific context, the performance of KG-BERT is limited. Observed results validate our hypothesis to supplement exBERT with scholarly context for the KG completion task.



(a) Link prediction performance on ORKG21.



(b) Link prediction performance on UMLS.



(c) Link prediction performance on PWC21.

Figure 7.5: **Link prediction performance for the top three performing baselines** per dataset against exBERT for each relation type. We show two metrics MRR and Hits@10 for each relation type (1-1, 1-N, N-1, N-N). We observe that depending on the dataset and sub-task (head/tail prediction), performance varies per relation type. The fine-grained analysis provides a detailed overview of the strength and weakness of exBERT against best performing models.

In Figure 7.4, most of the relations are quite ambiguous. Furthermore, in some cases, the scholarly context does not positively impact the performance of exBERT compared to KG-BERT. For instance, KG-BERT performs better than exBERT for the relation type *related_to*. KG-BERT is trained on a large unstructured corpus from generic domains, and *related_to* is a commonly occurring relation between two real-world entities.

7.4 Summary

The hypothesis investigated in this chapter was to study if task-specific context has an impact on scholarly KG completion task. For the same, we proposed exBERT and provided a set of experiments illustrating the positive impact of scholarly context encoded in exBERT for the KG completion task. We model the KG completion task as a sequence classification task, where we considered each KG triple as a set of sequences in a natural language. This allowed us to utilize SciBERT as an underlying model and adapt it for KG completion in the scholarly domain. We systematically studied the impact of our choices in the proposed approach. For instance, the ablation study demonstrates the effectiveness of scholarly context and provides insights on the strengths and weaknesses of exBERT. Albeit effective, exBERT is the first step of a larger research agenda in the direction of accurate scholarly knowledge graph completion.

CHAPTER 8

Question Answering on Scholarly Knowledge Graphs

Question Answering (QA) systems, such as Apple's Siri, Amazon's Alexa, or Google Now, answer questions by mining the answers from unstructured text corpora or open domain Knowledge Graphs (KG) [Kar+19]. The direct applicability of these approaches to specialized domains such as scholarly knowledge is questionable. On the one hand, no extensive knowledge graph for scholarly knowledge exists that can be employed in a question answering system. On the other hand, scholarly knowledge is represented mainly as unstructured raw text in articles (in proceedings or journals) [BM15b]. In unstructured artifacts, knowledge is not machine actionable, hardly processable, ambiguous [Bos+], and particularly also not FAIR [Wil+16a]. Still, amid unstructured information some semi-structured information exists, in particular in tabular representations (e.g., survey tables, literature overviews, and paper comparisons). The task of QA on tabular data has challenges [Lin02], shared with other types of question answering systems. We propose a method to perform QA specifically on scholarly knowledge graphs representing tabular data. Moreover, we create a benchmark of tabular data retrieved from a scholarly knowledge graph and a set of related questions. This benchmark is collected using the Open Research Knowledge Graph (ORKG) [Jar+19b].

The content of this chapter is mainly based on these publications [JSA20; Aue+22]. The results and contributions of this chapter aim to answer the following research question and tackles the subsequent research challenge.

> RQ5: How can we leverage state-of-the-art question answering techniques for scholarly knowledge?
>
> RC4: Retrieve Particular Information Based on User Needs from Natural Language Query.

In order to answer our research question, we propose JarvisQA. A system capable of performing question answering on tabular views of a scholarly knowledge graph. The contributions of this chapter can be summarized in the following:

- A question answering system JarvisQA, that supports multiple question and answer types on scholarly knowledge graphs.

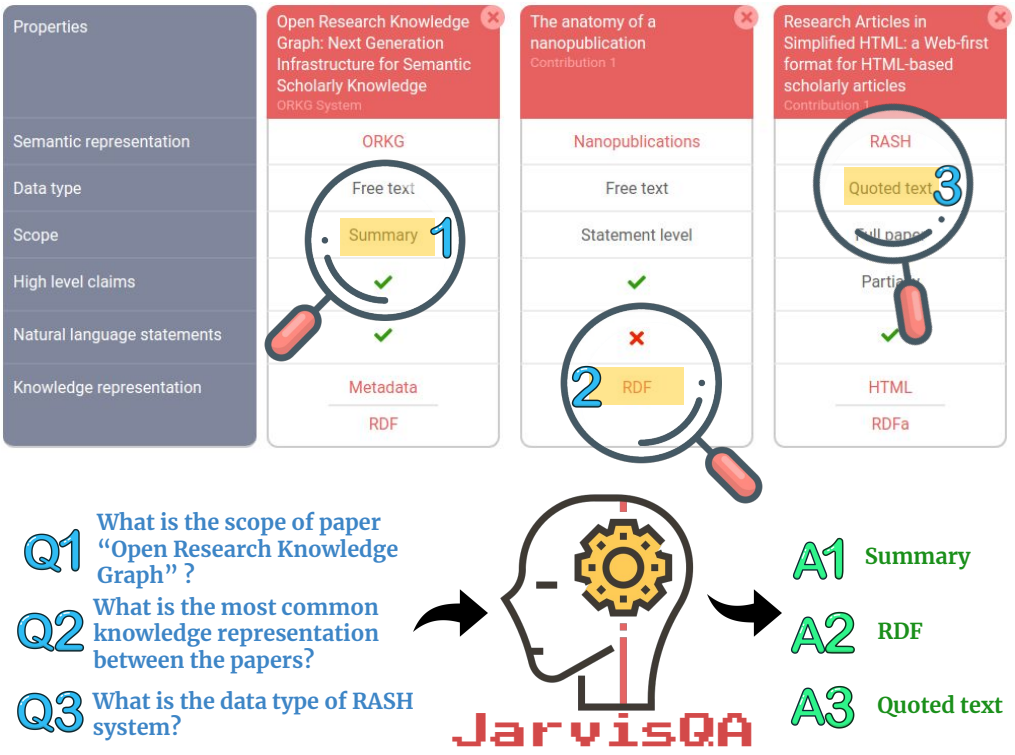


Figure 8.1: **Motivating Example.** JarvisQA takes as input a table of semi-structured information and tries to answer questions. Three types of questions are depicted here. (**Q1**) Answer is directly correlated with the question. (**Q2**) Aggregation of information from candidate results. (**Q3**) Answer relates to another cell in the table.

- Dataset of natural language questions and answers extracted from comparison tables of the ORKG.
- Evaluation of our system against common information retrieval tools on three datasets.
- Discussion about the evolution of the QA task on scholarly domain and the integration of our proposed dataset within a bigger benchmark.

The rest of the chapter, follows this structure: Section 8.1 starts by motivating the QA task and shows how intricate it is on scholarly knowledge. Section 8.2 present our approach and details about dataset collection. In Section 8.3 we describe the evaluation we performed, and how were the results.Section 8.4 discusses how our work integrates with the larger agenda of the SciQA benchmark. Finally, Section 8.5 summarizes the main contribution points.

8.1 Motivating Example

The research community has proposed many QA systems, but to the best of our knowledge none focus on scholarly knowledge. Leveraging the ORKG [Jar+19b] and its structured scholarly knowledge, we

propose a QA system specifically designed for this domain. In this case, the ORKG can be used as one source to tables describing scholarly information about specific papers. Figure 8.1 illustrates a tabular comparison view[1] of structured scholarly contribution descriptions. Additionally, three questions related to the content of the comparison table are shown. The answers are implicitly or explicitly provided in the cells of the table. JarvisQA can answer different types of questions. For Q1, the answer has a direct correlation with the question. For Q2, the system should first find the "knowledge representations" in the table and then find the most common value. For Q3, the answer is conditional upon finding another piece of information in the table first (i.e., JarvisQA has to find "RASH" in the table first), and then narrow its search to that column (or that paper) to find the correct answer.

In order to better analyze the research problem here, we split our research question **RQ5** into two sub-questions:

- RQ5.1: Can a QA system retrieve answers from tabular representations of scholarly knowledge?
- RQ5.2: What type of questions can be posed on tabular scholarly knowledge?

8.2 Approach

We propose a system, called JarvisQA , that answers Natural Language (NL) questions on tabular views of scholarly knowledge graphs, specifically tabular views comprising research contribution information from scientific articles.

8.2.1 Data and Questions Collection

In order to evaluate our QA system we create the ORKG-QA benchmark, collected using the ORKG. The ORKG provides structured comparisons [Oel+20] of research contributions obtained from papers. The ORKG-QA benchmark comprises a dataset that integrates 13 tables, covering information spanning more than 100 academic publications. The data is collected through the ORKG API[2], and the featured set of tables[3], which can be exported in CSV format.

Additionally, we created a set of questions that cover various types of information and facts that can be retrieved from those tables. The benchmark consists of 80 questions in English. The questions cover a variety of question types that can be asked in the context of tables in the scholarly literature. These types of questions include aggregation questions (e.g., min, average and most common), ask questions (i.e., true, false), answer listing questions, and questions that rely on combining information. In the ORKG-QA dataset[4], 39% are normal questions addressing individual cells in tables, 20% are aggregation questions, 11% are questions for which the answer relates to other parts of the table, and the rest are questions of different types (i.e., listings, ask queries, empty answers).

We also use the TabMCQ [JTH16] QA dataset, specifically questions on the *regents* tables. TabMCQ was derived from multiple choice questions of 4th grade science exams and contains 39 tables and 3 745

[1] https://www.orkg.org/orkg/comparison/R8618
[2] https://www.orkg.org/orkg/api/classes/Comparison/resources/?items=999
[3] https://www.orkg.org/orkg/featured-comparisons
[4] https://doi.org/10.25835/0038751

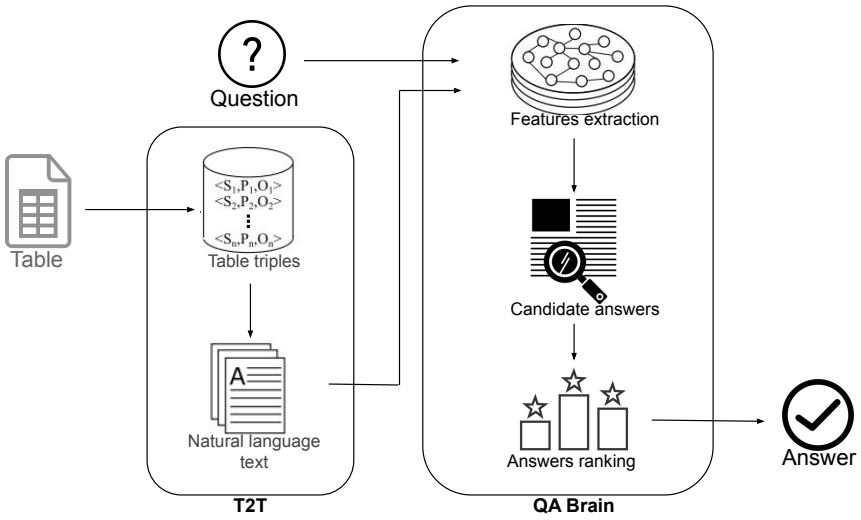


Figure 8.2: **System Architecture.** JarvisQA was designed with modularity in mind. The system has two main components. (a) **Table2Text (T2T)** component, which in turn has two functionalities: (1) to break the table into a set of triples $< s, p, o >$ and (2) to compile the triples into an NL sentence. Component (b) is the **engine of the QA system**, where an NL QA (BERT based) system is employed to answer the input question using the text, by extracting features, finding candidate answers, and ranking them.

related questions. While TabMCQ is not a scholarly dataset, but is to the best of our knowledge the closest one available. Since TabMCQ has only multiple-choice questions, we adapted the questions with only the correct choice. Although this is not directly related to scholarly information, we include it to indicate the performance increase in our approach.

8.2.2 JarvisQA System Architecture

JarvisQA is designed with modularity in mind. Hence, the core QA components are replaceable with newer or more fine-tuned versions. Figure 8.2 depicts the architecture in more detail. Since we used a natural language QA system, we need a pre-processing step that transforms the table information into the textual description (representing only the information contained in the table not the entire raw text of the article). With the output of the "Table2Text" step and the input question, the NL QA system can reason over the question with the provided context (textual table description) and attempts to answer the question. We now discuss the individual components of the architecture in more detail.

Table2Text (T2T) Converter. Although JarvisQA operates on tabular data, the core QA engine processes textual contexts. To that end, tables have to be converted into coherent text snippets that represent the entirety of the information presented in the table. T2T component splits tables into its entries and converts entries into triples. Table 8.1 illustrates a sample table containing some information about three publications, along with their triples and textual representations Furthermore, the T2T component enriches the textual description with aggregated information (i.e., max value of certain rows, most common value used within some columns). This enables the system to answer aggregation-type questions such as "Which is the maximum ...?" and "What is the most common ...?".

Table 8.1: **Sample of an input table.** The table is a part of the one shown in the motivating example.[5] Below, the representation in triples and as text is displayed.

Title	Semantic representation	Data type	Scope	High level claims
Paper 1 [Jar+19b]	ORKG	Free text	Summary	Yes
Paper 2 [GGV10]	Nanopublications	Free text	Statement level	Yes
Paper 3 [Per+17]	RASH	Quoted text	Full paper	Partially
Triples	<Paper1, hasSemanticRepresentation, ORKG> <Paper1, hasDataType, FreeText> <Paper1, hasScope, Summary> ...			
Text	Paper 1's semantic representation is "ORKG", its data type is "Free Text", and its scope is "Summary" ...			

QA Core Engine. This component is the primary building block of JarvisQA. It is where reasoning over questions happens. The component uses a pre-trained natural language QA model. The model is a deep transformer, fine tuned on the SQuADv2 dataset to perform the QA task. The component is replaceable with any other similar transformer model (of different sizes and architectures). Our base implementation uses a fine tuned version of a BERT model and we evaluate our model using different model sizes and architectures. The model parameters are set: *maximum sequence length* to 512, *document stride* to 128, *top k answers* to 10, *maximum answer length* to 15, and the *maximum question length* to 64. As illustrated in Figure 8.2, the QA engine extracts sets of features from the questions and the text (i.e., embeddings), then it finds a set of candidate answers and ranks them by confidence score. The benefits of such architecture are the flexibility in model choice, multilingualism, and reusability. Different transformer models can replace ours to support other languages, other datasets, and potentially other features. To accomplish these objectives, the system is built using the Transformers framework [Wol+20].

8.3 Experimental Study

We empirically study the behavior of JarvisQA in the context of scholarly tables against different baselines. The experimental setup consists of metrics and baselines. Table 8.2 lists the evaluation metrics for the performance measurements of the systems. Since a QA system can produce multiple answers and the correct answer can be any of the retrieved answers we use a metric that takes the position of the answer into account.
As baselines we use the following two methods for answer generation:

- *Random*: the answer is selected from all choices randomly. It is used to measure whether the model gains the result on this task.
- *Lucene*[6]: is a platform for indexing, retrieving unstructured information, and used as a search engine. We index the triple-generated sentences by Lucene. For each question, the top answer

[5] Fetched from https://www.orkg.org/orkg/c/Zg4b1N
[6] https://lucene.apache.org/

Table 8.2: **Evaluation metrics** used to experimentally benchmark JarvisQA against other baselines.

Metric	Definition
Global Precision	Ratio between correct answers retrieved in the top ranked position and the total number of questions.
Global Recall	Ratio between the number of questions answered correctly at any position (here till the 10th retrieved answer) and the total number of questions.
F1-Score	Harmonic mean of global precision and global recall.
Execution Time	Elapsed time between asking a question and returning the answer.
Inv. Time	$1 - \frac{average\ execution\ time\ for\ baseline}{maximum\ execution\ time\ for\ all\ systems}$.
In-Memory Size	The total memory size used by system.
Inv. Memory	$1 - \frac{memory\ size\ of\ baseline}{maximum\ memory\ size\ among\ all\ systems}$
Precision@K	Cumulative precision at position K.
Recall@K	Ratio of correctly answered questions in the top K position and total number of questions.
F1-Score@K	Harmonic mean of precision and recall at position K.

produced by Lucene is regarded as the final answer. We convert semi-structured tables into sentences, which can be indexed by Lucene. For each question, the top answer produced by Lucene is regarded as the final answer.

Implementation. The evaluation was performed on an Ubuntu 18.04 machine with 128GB RAM and a 12 core Xeon processor. The implementation is mostly based on HuggingFace Transformers[7], and is written in Python 3.7. The evaluation results for precision, recall, and F1-score are reproducible while other metrics such as time and memory depend on the evaluation system hardware. However, the ratio of the difference between the baselines should be similar or at least show a similar trend. The code to reproduce the evaluation results and the presented results are available online.[8]

8.3.1 JarvisQA performance on the ORKG-QA benchmark.

In order to evaluate the performance of JarvisQA , we run the system and other baselines on the ORKG-QA dataset at various k values (k denotes the position of the correct answer among all retrieved answers). For this experiment we evaluate $k \in \{1, 3, 5, 10\}$. Moreover, the experiment was conducted on a specific subset of questions (based on types) to show the performance of the system for certain categories of questions. The tested question categories are: *Normal*: normal questions about a specific cell in the table with a direct answer; *Aggregation*: questions about aggregation tasks on top of the table; *Related*: questions that require retrieving the answer from another cell in the table; *Similar*: questions that address the table using similar properties (e.g., synonyms). Table 8.3 shows the performance of the baselines and our system on the ORKG-QA benchmark. The results show that JarvisQA performs better by 2-3 folds against Lucene, and Random baselines respectively.

8.3.2 Different models of QA and their performance.

We evaluate different types of QA models simultaneously to show the difference in performance metrics, execution time, and resource usage. Table 8.4 illustrates the difference in performance on the ORKG-

[7] https://github.com/huggingface/transformers
[8] https://github.com/YaserJaradeh/JarvisQA

Table 8.3: **JarvisQA performance on the ORKG-QA benchmark dataset** of tabular data. The evaluation metrics are precision, recall, and F1-score at *k* position. JarvisQA is compared against two baselines on the overall dataset and specific question types. The symbol (-) indicates that the performance metric showed no difference than the reported value for higher *K* values. The results suggest that JarvisQA outperforms the baselines by 2-3 folds.

Question types	Baseline	Precision @K				Recall @K				F1-Score @K			
		#1	#3	#5	#10	#1	#3	#5	#10	#1	#3	#5	#10
All	Random	0.02	0.06	0.08	0.16	0.02	0.07	0.09	0.18	0.02	0.06	0.08	0.17
All	Lucene	0.09	0.19	0.20	0.25	0.09	0.18	0.19	0.24	0.09	0.18	0.19	0.24
Normal	JarvisQA	0.41	0.47	0.55	0.61	0.41	0.47	0.53	0.61	0.41	0.47	0.54	0.61
Aggregation	JarvisQA	0.45	-	-	-	0.45	-	-	-	0.45	-	-	-
Related	JarvisQA	0.50	0.50	1.00	1.00	0.50	0.50	1.00	1.00	0.50	0.500	1.00	1.00
Similar	JarvisQA	0.11	0.25	0.67	-	0.11	0.25	0.67	-	0.11	0.25	0.67	-
All	JarvisQA	0.34	0.38	0.46	**0.47**	0.35	0.38	0.46	**0.48**	0.34	0.38	0.45	**0.47**

QA benchmark dataset for different classes of questions and the overall dataset. Though different models perform better on certain types of questions. JarvisQA's QA engine employs the *BERT L/U/S2* as our base model due to its execution time and overall higher accuracy at higher positions.

8.3.3 Trade-offs between different performance metrics.

We illustrate trade-offs between different dimensions of performance metrics for the JarvisQA approach compared to the baselines. We choose global precision, global recall, F1-score, in-memory size, and execution time as five different dimensions. Figure 8.3 depicts the performance metrics trade-offs between our system and other baselines. JarvisQA achieves higher precision and recall while consuming considerably more time and memory than the other baselines.

8.3.4 Performance on TabMCQ.

We also show the performance of our system on the TabMCQ dataset against the ORKG-QA dataset. We see the same trend in both datasets, that JarvisQA outperforms the baselines by many folds. TabMCQ is not directly related to scholarly knowledge. However, it shows that JarvisQA can generalize to related data and can answer questions about it. Table 8.5 presents the results of this experiment.

8.4 Toward Scientific QA Benchmark

Our work here laid down the first steps in creating a usable dataset of questions and answers on scholarly knowledge. This attracted efforts of the community to create a bigger QA benchmark with a better coverage. SciQA[9] [Aue+22] is a question answering benchmark created on the ORKG[10]. This benchmarks extends the our presented dataset to include not only tabular views (i.e., comparison)

[9] The SciQA benchmark is an effort of bigger research team. Our contributions are to the base dataset of question, question creation, query engineering, and JarvisQA's evaluation.

[10] The complete benchmark can be found on https://zenodo.org/record/6517080#.YnJV1Z9BzmE

Table 8.4: **Performance comparison of different deep learning models** on the task of question answering with different model sizes and architectures using the ORKG-QA benchmark dataset. The results suggest that different models perform differently on various question types, and generally the bigger the model the better it performs. For each question type, the best results are highlighted.

	Questions type	Precision @K				Recall @K				F1-Score @K			
		#1	#3	#5	#10	#1	#3	#5	#10	#1	#3	#5	#10
BERT L/U/S1	Normal	0.35	0.49	0.53	**0.68**	0.34	0.47	0.51	**0.67**	0.34	0.48	0.52	**0.67**
	Aggregation	0.39	0.39	0.45	-	0.39	0.39	0.45	-	0.39	0.39	0.45	-
	Related	0.50	0.64	0.64	0.80	0.50	0.64	0.64	0.80	0.50	0.64	0.64	0.80
	Similar	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-
	All	0.31	0.38	0.44	**0.50**	0.31	0.38	0.43	**0.49**	0.3	0.38	0.43	**0.50**
BERT L/C/S1	Normal	0.31	0.44	0.45	-	0.31	0.43	0.45	-	0.31	0.43	0.45	-
	Aggregation	0.27	0.39	0.39	0.45	0.29	0.39	0.39	0.45	0.27	0.39	0.39	0.45
	Related	0.65	**1.00**	-	-	0.70	**1.00**	-	-	0.67	**1.00**	-	-
	Similar	0.11	0.11	0.25	0.43	0.11	0.11	0.25	0.43	0.11	0.11	0.25	0.43
	All	0.27	0.35	0.37	0.39	0.29	0.37	0.39	0.41	0.27	0.36	0.37	0.40
BERT B/C/S2	Normal	0.09	0.23	0.26	0.31	0.1	0.24	0.27	0.32	0.09	0.24	0.26	0.32
	Aggregation	0.07	0.19	0.19	0.33	0.07	0.19	0.19	0.33	0.07	0.19	0.19	0.33
	Related	0.12	0.29	0.29	0.38	0.12	0.28	0.28	0.38	0.12	0.28	0.28	0.38
	Similar	0.11	0.11	0.11	0.25	0.11	0.11	0.11	0.25	0.11	0.11	0.11	0.25
	All	0.087	0.16	0.17	0.24	0.1	0.18	0.2	0.26	0.09	0.17	0.18	0.24
BERT L/U/S2	Normal	0.41	0.47	0.55	0.61	0.41	0.47	0.54	0.61	0.41	0.47	0.54	0.61
	Aggregation	0.45	-	-	-	0.45	-	-	-	0.45	-	-	-
	Related	0.50	0.50	**1.00**	-	0.50	0.50	**1.00**	-	0.50	0.50	**1.00**	-
	Similar	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-
	All	0.35	0.38	0.46	0.48	0.35	0.38	0.46	0.48	0.34	0.38	0.46	0.48
Distil BERT B/U/S1	Normal	0.14	0.27	0.36	0.46	0.16	0.29	0.36	0.46	0.15	0.27	0.35	0.45
	Aggregation	0.22	0.39	-	-	0.25	0.41	-	-	0.24	0.39	-	-
	Related	0.31	0.50	0.64	-	0.31	0.50	0.64	-	0.31	0.50	0.64	-
	Similar	0.00	-	-	-	0.00	-	-	-	0.00	-	-	-
	All	0.16	0.23	0.28	0.33	0.17	0.26	0.29	0.35	0.16	0.24	0.28	0.33
ALBERT XL/S2	Normal	0.34	0.47	0.51	-	0.34	0.47	0.51	-	0.34	0.47	0.51	-
	Aggregation	0.45	0.45	**0.52**	-	0.45	0.45	**0.52**	-	0.45	0.45	**0.52**	-
	Related	**1.00**	-	-	-	**1.00**	-	-	-	**1.00**	-	-	-
	Similar	0.43	0.43	**0.67**	-	0.43	0.43	**0.67**	-	0.43	0.43	**0.67**	-
	All	0.36	0.42	0.46	-	0.37	0.43	0.47	-	0.36	0.42	0.46	-

B=Base; L=Large; XL=X-Large; C=Cased; U=Uncased; S1=Finetuned on SQuAD1; S2=Finetuned on SQuAD2

Table 8.5: **Performance comparison using the two datasets TabMCQ and ORKG-QA** against JarvisQA and the baselines. The results suggest that JarvisQA outperforms the baselines by substantially on both datasets. Best results are highlighted for both datasets.

System	Dataset	Precision @K				Recall @K				F1-Score @K			
		#1	#3	#5	#10	#1	#3	#5	#10	#1	#3	#5	#10
Random	TabMCQ	0.006	0.010	0.020	0.030	0.010	0.020	0.030	0.040	0.007	0.010	0.024	0.030
	ORKG	0.020	0.060	0.080	0.160	0.020	0.070	0.090	0.180	0.020	0.060	0.080	0.017
Lucene	TabMCQ	0.004	0.018	0.027	0.036	0.006	0.017	0.026	0.037	0.005	0.016	0.024	0.033
	ORKG	0.090	0.190	0.200	0.250	0.090	0.180	0.190	0.240	0.090	0.180	0.190	0.240
Jarvis	TabMCQ	0.060	0.090	0.100	**0.110**	0.070	0.090	0.110	**0.120**	0.060	0.080	0.100	**0.110**
	ORKG	0.340	0.380	0.460	**0.470**	0.350	0.380	0.460	**0.480**	0.340	0.380	0.450	**0.470**

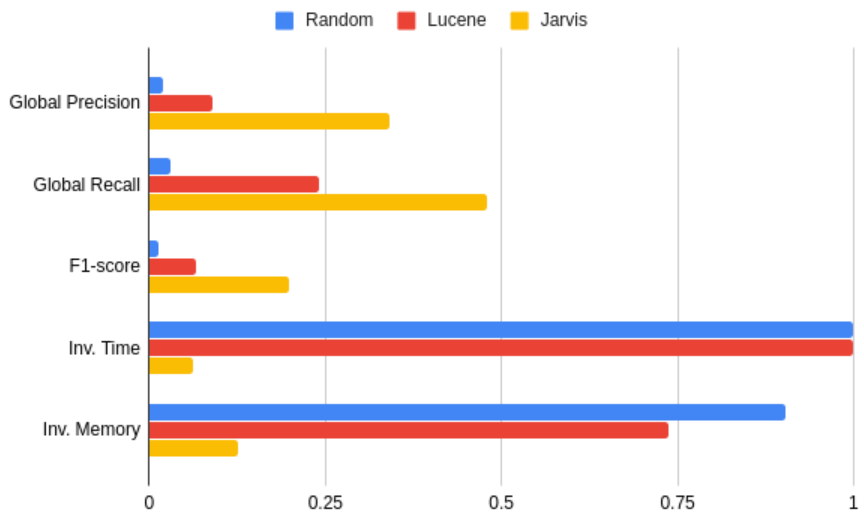


Figure 8.3: **Performance of the JarvisQA system**. JarvisQA and the baselines are compared in terms of Global Precision, Global Recall, Global F1-Score, Inv.Time, Inv.Memory; higher values are better. JarvisQA improves Precision, Recall, and F1-Score by up to three times at the cost of execution time and memory consumption.

centric questions, rather paper centric, and even graph center questions. It also provides paraphrasing of questions, structured queries in SPARQL over the graph, and details about question type and used keywords.

For the collection of the SciQA benchmark First of all, a list of research fields and the list of ORKG comparisons in this field are collected to limit the scope of the data being queried. Furthermore, several natural language questions according to different types, e.g., single comparison questions, True/False questions, aggregation questions (min, max, average), etc are defined. Lastly, at least one SPARQL query for each of the natural language questions in two variations (human-readable and machine-readable) is created.

The objective of SciQA is not only to create a dataset of questions and answers, rather also to provide a collection of knowledge graph scenarios, such as what questions can be asked on the KG or what usecases can utilize the underlying data. Thirteen researchers worked together on the creation of the SciQA benchmark to cover different perspectives. All the researchers peer-reviewed and cross-checked the questions and the structured queries multiple times in the benchmark to ensure syntactic and semantic soundness. Figure 8.4 shows the collection workflow that is followed when creating SciQA benchmark.

SciQA classifies the question it contains based on two dimensions. First, the **ORKG-content** which represents the structure of the subgraph in the ORKG relating to the answer of the question. This is mainly "paper-based" such as questions on the content of a individual research paper or a collection of papers. "Comparison-based" which are questions on the content of a comparison, i.e., on the properties of the comparison with relation to the contributions participating. Second, the **question-content** such as factoid and non-factoid questions, following Mikhailian, Dalmas, and Pinchuk [MDP09]. Factoid questions expect a direct and explicit mapping to the entities of the ORKG ontology. However, if the question requires more complicated inferencing of a sequence of facts, aggregation, filtering, or counting, it is considered non-factoid.

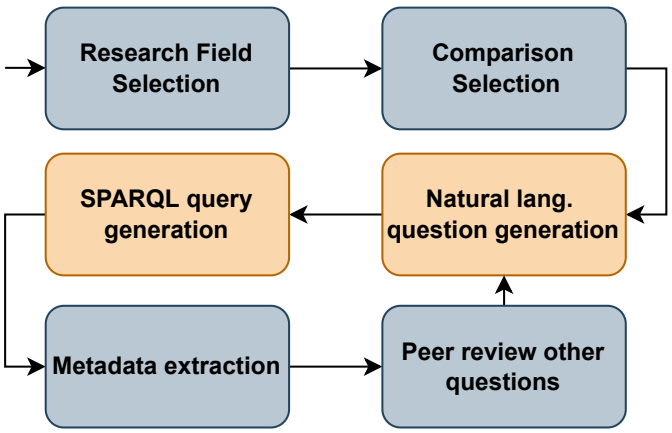


Figure 8.4: **SciQA benchmark collection workflow**. A research field and a list of comparisons in this field is selected. This is followed by the creation of questions and their SPARQL queries. Metadata is then collected on the questions and queries, e.g., type, and query shape. Finally, a peer review process is followed.

The resulting benchmark covered a big number of research fields in the ORKG (48). SciQA contains 100 SPARQL queries, with the same number of questions, and at least on paraphrasing per questions. The SPARQL covers a wide variety of shapes; 47 Tree, 39 Chain, 7 Star, and 5 Forest.

Excerpt 8.1 shows an example question of SciQA belongs to the research field *Ecology and Biodiversity of Animals and Ecosystems, Organismic Interactions* is "**Where did the study with maximal geographic scale take place in Genetic Variability (COI Variation) in Studies Large Sampled (>1000 Sequences)?**". This non-factoid question is based on the comparison *Genetic Variability (COI Variation) in Studies Large Sampled (>1000 Sequences)*[11] which compares the genetic variability in studies containing more than 1000 cytochrome c oxidase I (COI) barcoding sequences. The question aims to identify where the study with the maximum geographic scope took place, which in this case is a study conducted in the United States of America, Mexico, and Canada. The SPARQL query (below) has six triple patterns, uses six query components, and is shaped as a tree [Aue+22].

8.4.1 Feasibility Evaluation

We take a subset of the questions that can be answered by JarvisQA from SciQA and apply a feasibility analysis to see how will our system generalized on unseen types of questions and data. JarvisQA is fundamentally designed to answer questions about scholarly knowledge. The system is based on BERT but works only on tabular views of scholarly knowledge graphs, such as ORKG comparisons. SciQA does not rely only on tabular views (comparisons) but has a broader spectrum of question/answer types. For this reason, we can answer 52 questions (52%) with JarvisQA as they correspond to its input form.

We configure JarvisQA to run on the compatible questions of SciQA and we use seven distinct experimental setups that JarvisQA provides. Due to the limited coverage of questions that the system can answer we limit the results to two categories of questions. The evaluation is conducted in terms of *precision@k*, *recall@k*, and *f1@k* metrics.

[11] https://www.orkg.org/orkg/comparison/R149849

```
SELECT ?location, ?location_label
WHERE {
  {SELECT (MAX(?geo_scale_num) AS ?max_geo_scale)
   WHERE {
    orkgr:R149849 orkgp:compareContribution ?contrib.
    ?contrib orkgp:geographicScale ?geo_scale.
    BIND(xsd:integer(?geo_scale) AS ?geo_scale_num)
   }
  }
  orkgr:R149849 orkgp:compareContribution ?contrib.
  ?contrib orkgp:geographicScale ?geo_scale;
           orkgp:studyLocation ?location.
  BIND(xsd:integer(?geo_scale) AS ?geo_scale_num).
  ?location rdfs:label ?location_label.
}
GROUP BY(?location_label)
HAVING(?geo_scale_num = ?max_geo_scale)
```

Excerpt 8.1: **Example query from SciQA.** A SPARQL query on the ORKG data model to answer a single question from the SciQA benchmark.

Table 8.6: **Evaluation results of running JarvisQA against SciQA** benchmark questions. Top performing setup is indicated in bold, second best is underlined.

JarvisQA Setup	Normal						Overall					
	Precision		Recall		F1		Precision		Recall		F1	
	@1	@10	@1	@10	@1	@10	@1	@10	@1	@10	@1	@10
$BERT_{BUS}$	1905	.2712	1906	.2713	1905	.2712	1364	.1905	1364	.1905	1364	.1905
$BERT_{LCS}$	**.1935**	.2542	**.1937**	.2545	**.1936**	.2543	**.1379**	.1786	**.1379**	.1786	**.1379**	.1786
$BERT_{BCS2}$	.1343	.1875	.1344	.1876	.1343	.1875	.0978	.1348	.0978	.1348	.0978	.1348
$BERT_{LUS2}$	.1693	2883	.1692	2881	.1692	2882	.1222	2024	.1222	2024	.1222	2024
$DistBERT_{BUS}$	.1343	.2459	.1343	.2459	.1343	.2459	.0978	.1744	.0978	.1744	.0978	.1744
$ALBERT_{XLS2}$	.1719	**.3393**	.1719	**.3394**	.1719	**.3393**	.1250	**.2346**	.1250	**.2346**	.1250	**.2346**
$ALBERT_{XXLS2}$	.1692	.2459	.1692	.2459	.1692	.2459	.1222	.1744	.1222	.1744	.1222	.1744

B=Base; L=Large; XL=X-Large; C=Cased; U=Uncased; S1=Finetuned on SQuAD1; S2=Finetuned on SQuAD2

Table 8.6 shows the evaluation results of these experiments for two main categories of questions *normal* and *overall*. While the category *normal* refers to single answer questions, the category *overall* aggregates single answer questions and all other question types that JarvisQA can answer, such as listing and boolean questions. We note that the performance decreases across all the setups for the overall category because of the complex nature of the SciQA benchmark and the answers it expects, unlike what JarvisQA can answer.

The main objective of JarvisQA is to serve as a system that allows users to ask natural language questions on tablar views of scholarly knowledge. As such, the system addresses only a small part of the scholarly information corpus.

We performed several experimental evaluations to benchmark the performance of JarvisQA against other baselines using two different QA datasets. Different datasets showed different results based on the types of questions and the nature of the scholarly data encoded in the tables. Based on these extensive

experiments, we conclude that usual information retrieval techniques used in search engines are failing to find specific answers for questions posed by a user. JarvisQA outperforms the other baselines in terms of precision, recall, and F1-score measure at the cost of higher execution time and memory requirements. Moreover, our system cannot yet answer all types of questions (e.g., non-answerable questions and listing questions).

Since JarvisQA utilizes a BERT based QA component, different components can perform differently, depending on the use case and scenario. Our system struggles with answers spanning across multiple cells of the table, and also in answering true/false questions. Furthermore, the answers are limited to information in the table (extractive method), since tables are not supplemented with further background information to improve the answers. We also note that with the results from Table 8.6 in mind, we see that JarvisQA is able to address only a small portion of what a scholarly question answering system should be able to. The integration with the SciQA benchmark is critical because it will help evolve our system plus other community approaches to push the boundaries of question answering on scholarly knowledge graphs.

8.5 Summary

Retrieving answers from scientific literature is a complicated task. Manually answering questions on scholarly data is cumbersome, time consuming, and requires expertise. Thus, an automatic method of answering questions posed on scientific content is needed. JarvisQA is a question answering system addressing scholarly data that is encoded in tables or sub-graphs representing tabular content. It can answer several types of questions on table content (such as listing, aggregation, and term extraction). Furthermore, our ORKG-QA dataset is a starting point to collaborate on adding more data to better train, evaluate, and test QA systems designed for tabular views of scholarly knowledge, which is fulfilled with the integration of our preliminary dataset into a much more thought and detailed scholarly QA benchmark (SciQA). To conclude, JarvisQA addresses several open questions in current information retrieval in the scholarly communication domain, and contributes towards improved information retrieval on scholarly knowledge. It can help researchers, librarians, and ordinary users to inquire for answers with higher accuracy than traditional information retrieval methods.

CHAPTER 9

Conclusions and Future Directions

THROUGHOUT THIS THESIS we studied the methods of constructing a scholarly knowledge graph leveraging natural language processing techniques. To this goal we presented and co-implemented the Open Research Knowledge Graph, an infrastructure capable of representing structured descriptions of scholarly communications using a knowledge graph. With the infrastructure as the technical foundation for our work and research, we investigate various methods of automated machine support of NLP tools for the goal of populating a knowledge graph. Thus, we propose a technique for dynamic blind information extraction pipeline generation, and a complementing technique for targeted information extraction. We support these techniques with another for automated population of the scholarly knowledge graph from academic text. Lastly, we propose an intelligent information retrieval method through question answering on top of tabular representations of scholarly knowledge graphs.

9.1 Revisiting the Research Questions

RQ1: What are the requirements and possible implementation options for a semantic scholarly knowledge graph representing scientific contributions?

Chapter 4 presents a vision and an implementation of a next generation semantically enabled scholarly infrastructure i.e., the Open Research Knowledge Graph (ORKG), which is poised to store, represent, model, manipulate, and query structured description of scholarly communication. ORKG lays down the foundation to create a scholarly knowledge graph that is centered around scientific papers (or even single contribution of a paper). We co-implement and co-design the ORKG ecosystem with other members of the ORKG R&D team. We present and RDF-like data model that is used to model descriptions maintain provenance and other metadata. We also describe the architecture of the ORKG ecosystem, and formalize the most important features of the system. Furthermore, we highlight various approach of knowledge acquisitions implemented within the ORKG, what is the usecase of each one of them, and who are the targeted stakeholders to use it. We conducted preliminary user study to see if researchers are willing to contribute structured description into the knowledge graph, as well as an empirical study to see how automated techniques can be integrated. The results show that researchers

are willing to use the approach, as long as certain accommodations are provided to ease the process and save time an effort. Our work on the ORKG is part of a bigger agenda achieved with a larger ORKG team, and it serves as the basis for all other research questions and contributions presented in this thesis.

RQ2: How does the dynamic selection of pipelines based on the input text affect the end-to-end information extraction task?

Chapter 5 addresses part of the information extraction challenge over scholarly knowledge. In this chapter we present PLUMBER, and approach and a framework to dynamically create information extraction pipelines from community created IE tools. We analyze the individual efforts of the research community in various areas of NLP subtasks, and we find that we can combine existing work for the bigger objective of information extraction. We propose an architecture that can combine various IE sub-tasks - e.g., named entity recognition, relation linking, and coreference resolution - under dynamic pipelines that are composed based on characteristic of input text. Moreover, we evaluate PLUMBER on three knowledge graphs and three datasets. With comprehensive error analysis and ablation studies to gain more accurate insights into error propagation, and bad performance. Lastly, we demo how this approach fits into the workflow of the ORKG ecosystem and how it can help users throughout their process of scholarly data curation. Our work on PLUMBER showcases that open IE or blind IE is not suitable enough for tight integration within scholarly data management systems or scholarly knowledge graph, and it highlights the need for a more direct and targeted approach.

RQ3: How can we guide information extraction processes on scholarly documents?

To answer this research question, we propose a novel approach of targeted information extraction on scholarly articles. In Chapter 6 we present a structured approach (MORTY) that leverages structured summaries to perform information extraction. First, we motivate the problem with a real example of how difficult it is to extract certain pieces of information from scholarly text. Then, we describe the architecture of your approach and formalize it. We also present a sizable dataset specially for the task of information extraction from scholarly articles alongside structured summaries . Furthermore, we evaluate the MORTY using multiple underlying models, and compare it against two common approaches of information extraction to study the efficacy of the approach. We discuss how to integrate our presented approach into a usecase of automated completion of literature reviews. Finally, we discuss the limitations of MORTY and what are the open points that should be investigated in the future. The work on this chapter is complementary to Chapter 5 on the same research challenge and can be seen in contrast of each other.

RQ4: What is the impact of task-specific context on scholarly KG completion?

In Chapter 7 we highlight yet another automated approach of knowledge graph completion. In this chapter we present EXBERT, a method of employing textual triple classification for scholarly knowledge graph completion. KG completion is important, because of the incomplete methods of KG creation and unreliable source. We describe two newly created datasets from scholarly-natured data sources that can be used for the training and evaluation of KG completion systems. Moreover, we perform an extensive evaluation of EXBERT against various other knowledge graph embedding techniques over three different scholarly datasets. We also perform an in-depth analysis of which relations are affecting

the KG completion task and what other elements have impact on the overall performance. Last but not least, we discuss the insights we gained from this process and make assumptions and hypothesis that should be further investigated by the research community.

RQ5: How can we leverage state-of-the-art question answering techniques for scholarly knowledge?

Chapter 8 tackles this research question by describing the foundations of a question answering system that is able to operate on scholarly sub-graphs. We present JarvisQA, a system that answers natural language questions on tabular views of the ORKG knowledge graph (a.k.a. literature comparisons). We also put forward a dataset of natural language questions and their answers collected from the ORKG as a first scholarly benchmark, which is picked by the SciQA benchmark and expanded upon. Furthermore, we evaluate the performance of JarvisQA on three datasets, and analyze metrics trade-off based on the underlying language model. Finally, we discuss the results and highlights the strengths and the limitations of the proposed approach, and how to improve it. Our work in this chapter builds on everything above as it is at the top of the pyramid. Meaning, in order for any information retrieval methods to work, structured annotations have to be present in a knowledge graph, which in turn have to be extracted from legacy non machine-actionable scholarly articles.

9.2 Open Issues and Future Directions

This section contains the open issues and future directions of the research conducted, and presented throughout this thesis. Since the contributions of the thesis, are layered in three layers, we also consider the open the challenges in the same manner (remember Figure 1.1). At the middle layer (i.e., the semantic scholarly infrastructure), the open challenge is to find the best representation of the scholarly communication, as well as the balance of knowledge acquisition from pure crowdsourcing to fully automated. Since this layer is the middleware between machine actionable data consumption and the extraction of this data, the changes it undertakes carries a lot of repercussions on other layers in the workflow. Soundly, for the bottom extraction layer, which contains all the automated systems that enable the autonomous population of the knowledge graph. Variety of open challenges remain here, of how to integrate such tools within the workflow. How much should the graph rely on the data without user intervention. And what other approaches should be used in conjunction with everything presented here in this thesis. Finally, on the top layer, the main challenge is having large and better annotated data that can be used to train and improve data consumption applications.

We consider these open challenges as out-of-scope of this work and should be investigated furthermore. Moreover, we deem the following future works relating to our contributions crucial to guide future research:

Mass import of structured descriptions. The importing of semi-structured scholarly knowledge from other sources, such existing content is crucial to build a sizable database to use for future development, testing and demonstration of a semantic scholarly infrastructure. With the mass import, user authentication and versioning to track the evolution of curated scholarly knowledge is of the utmost importance.

Intuitive methods of scholarly knowledge exploration and curation. Integrating scholarly knowledge (graph) visualization approaches in the ORKG in order to support novel and advanced explorations (e.g., relation finding, and pattern spotting). Furthermore, the addition of a discussion feature that enables debating existing scholarly knowledge, e.g., for knowledge correctness. As well as, employing automated approaches by not autonomously adding knowledge, rather to suggest to users and let users control what is added or not.

Multilinguality support. Our research in this thesis focuses only on English. We argue that the work should be expanded to include multi-lingual parsers and extractors. This would help as well to find inter-lingual connection and transfer knowledge from one language to the other.

Incorporating background knowledge. Multiple tools and components developed by the community relies on a set of hand-crafted rules or learned a set of constrained parameters and do not include any insight into background knowledge [Mul+20; Bas+21]. We argue that including background knowledge improves the performance of individual components. For instance, PLUMBER's text triple extractor suffer from quality across KGs, we need to incorporate the KG's underlying schema to guide the triple extraction process.

Morphology-less processing. Various pre-processing approaches used by NLP tools and method suffer from issues such as case-sensitivity in entity linkers, or implicit entities also challenge the entity linking performance. Yang et al. [Yan+19] introduced entity descriptions as additional context to support implicit entity linking, and we deem such approaches to be beneficial for entity linking tools as well as other NLP methods. Similarly for handling implicit relations and covered relations in relation linking which are primary sources of error.

Improve structured summaries. Experimenting with various structured summary formats and studying their effect is important to understand the effects of some design decisions for targeted information extraction, as well as enabling larger input sizes language models. We also suggest to incorporate active learning with user feedback collected from a user interface within a scholarly infrastructure. Moreover, to perform a user evaluation to study the efficacy of the IE task on scholarly data for users as their point of view might be unaligned with what an empirical evaluation would show.

Enhance semantic representations of human readable labels. In the case of scholarly knowledge graph completion, scholarly context has positively impacted the performance; however, there are several relation types for which EXBERT showed limited performance. One potential reason is the scarcity of training occurrences for certain relations such as *has_qos*. We believe that for unseen entities and relations, a zero-shot setting would be more suitable, and a set of additional experiments are needed to verify these observations. Similarly, relying on labels and textual descriptions of an entity, data quality is a crucial aspect—as is typical for knowledge-intensive tasks [WKB18]. We argue that quality of contextual data impacts the performance of all natural language processing tasks.

Including context for better information retrieval. In the case of JARVISQA, improving answer selection techniques, and supporting more types of questions requires incorporating more context and being able to analyze this context and make assumptions based on it. Moreover, for IR tasks to be able

to not only finding a term (or an answer) from the text, rather to find the text first and then the target term is a bigger undertaking and requires the ability to narrow down the scope to a particular context and extract values.

Better training datasets. As it is with all machine learning and natural language processing methods, a dataset is very important and the more the dataset is expressive and contains more samples, the better the resulting models will be. In our contributions of this thesis, we have collected and annotated some to verify our observations and hypotheses. We argue that for future directions to succeed, more annotations, better coverage, and larger amounts of data are required to train models on various tasks like question answering, information retrieval, or knowledge graph completion.

As it is, each contribution chapter in this thesis still has a lot of room to improve singular components or the overall strategy. For instance, in PLUMBER, relation linking components compromised the performance a lot and it seems that we would need to create more performance and tailored linkers for scholarly knowledge. In the case of EXBERT, we relied on the labels of resource and not anything else, and we didn't study the effect of different languages, and representations on the performance of knowledge graph completion. This is but a few of the points mentioned in each of the individual publications of the core contributions, which leaves many open questions for the research community.

9.3 Closing Words

The growing amount of published research and researchers on the Web demands that we move scholarly communication to the next century. Structuring and semantifying knowledge from scholarly publication is only one step in this larger agenda of shedding a new light on the way we do research. Natural language processing tools as well as machine learning, are important keystones to take advantage of to improve, collect, extract, and verify scholarly communication. In this thesis, we have shown a proposal of a semantic scholarly infrastructure with a semantic description model to hold and represent scholarly descriptions of articles and contributions in a machine readable/actionable manner. Moreover, we build a suite of approaches that compliments the structured descriptions by extracting and auto-completing scholarly annotations from textual articles into an interlinked graph structure. Which is then ingested by what we propose to be a question answering system that can handle natural language queries on the scholarly knowledge graph and retrieves suitable and precise answers for them. The work of this thesis, is one step in the direction of cementing the foundations of applying natural language processing techniques on scholarly communication and knowledge graphs. Finally, our efforts are continued in the openly available and accessible ORKG[1] system taking part in a practical usecase that aims to pushing the entirety of research forward.

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

Bibliography

[AB07] B. Adida and M. Birbeck, *RDFa Core 1.1* (2007) (cit. on p. 17).

[AA19] K. Adnan and R. Akbar, *An analytical study of information extraction from unstructured and multidimensional big data*, Journal of Big Data **6** (2019) 91, ISSN: 2196-1115, DOI: 10.1186/s40537-019-0254-8 (cit. on p. 28).

[AI21] F. AI, *The latest in Machine Learning | Papers With Code*, https://paperswithcode.com/, (Accessed on 05/14/2021), 2021 (cit. on p. 101).

[ABV18] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual String Embeddings for Sequence Labeling," *COLING 2018, 27th International Conference on Computational Linguistics*, Association for Computational Linguistics, 2018 1638 (cit. on p. 37).

[Ali+21] M. Ali et al., *PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings*, Journal of Machine Learning Research **22** (2021) 1 (cit. on p. 102).

[All17] R. B. Allen, *Rich Semantic Models and Knowledgebases for Highly-Structured Scientific Communication.*, CoRR **abs/1708.08423** (2017) (cit. on p. 32).

[All12] R. B. Allen, *Supporting Structured Browsing for Full-Text Scientific Research Reports*, CoRR **abs/1209.0036** (2012) (cit. on p. 32).

[AN14] H. F. Alrøe and E. Noe, *Second-Order Science of Interdisciplinary Research: A Polyocular Framework for Wicked Problems*, Constructivist Foundations (2014) 65 (cit. on p. 43).

[Amm+18] W. Ammar et al., "Construction of the Literature Graph in Semantic Scholar," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, Association for Computational Linguistics, 2018 84 (cit. on pp. 32, 97).

[An+18] B. An, B. Chen, X. Han, and L. Sun, "Accurate Text-Enhanced Knowledge Graph Representation Learning," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, 2018 745 (cit. on pp. 95, 102, 107).

[Ana+95] R. Anand, K. Mehrotra, C. K. Mohan, and S. Ranka, *Efficient classification for multiclass problems using modular neural networks*, IEEE Transactions on Neural Networks **6** (1995) 117 (cit. on p. 67).

[AJM15] G. Angeli, M. J. Johnson Premkumar, and C. D. Manning, "Leveraging Linguistic Structure For Open Domain Information Extraction," ACL, 2015 344 (cit. on pp. 39, 71, 72).

[Ara22] O. Arab Oghli, *Information Retrieval Service Aspects of the Open Research Knowledge Graph*, en (2022), DOI: 10.15488/11834, URL: https://www.repo.uni-hannover.de/handle/123456789/11929 (cit. on p. 48).

[Arn+21] D. Arndt et al., *RDF-star and SPARQL-star*, https://www.w3.org/2021/12/rdf-star.html, (Accessed on 05/11/2022), 2021 (cit. on p. 21).

[AW17] A. Aryani and J. Wang, "Research Graph: Building a Distributed Graph of Scholarly Works using Research Data Switchboard," *Open Repositories CONFERENCE*, 2017, published (cit. on p. 32).

[Aue+07] S. Auer et al., "DBpedia: A nucleus for a Web of open data," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4825 LNCS, 2007 722, ISBN: 3540762973, DOI: 10.1007/978-3-540-76298-0_52 (cit. on pp. 4, 16, 17, 34, 39, 49, 61, 69, 95).

[Aue+20] S. Auer et al., *Improving Access to Scientific Literature with Knowledge Graphs*, BIBLIOTHEK – Forschung und Praxis (2020), DOI: http://dx.doi.org/10.18452/22049 (cit. on p. 44).

[Aue+22] S. Auer et al., *SciQA – A Question Answering Benchmark for Scholarly Knowledge*, Under review at ISWC 2022, 2022 (cit. on pp. 33, 111, 117, 120).

[Aue+18] S. Auer et al., "Towards a Knowledge Graph for Science," *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, 2018 (cit. on p. 44).

[BM19] V. Bairagi and M. V. Munot, *Research methodology: A practical and scientific approach*, CRC Press, 2019 (cit. on p. 1).

[Bak16] M. Baker, *1,500 scientists lift the lid on reproducibility*, Nature News **533** (2016) 452 (cit. on p. 97).

[BAH19a] I. Balazevic, C. Allen, and T. Hospedales, *TuckER: Tensor Factorization for Knowledge Graph Completion*, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (2019) (cit. on p. 102).

[BAH19b] I. Balažević, C. Allen, and T. Hospedales, *Multi-relational Poincaré Graph Embeddings*, 2019, arXiv: 1905.09791 [cs.LG] (cit. on p. 102).

[Bal18] K. Balog, "Entity Linking," *Entity-Oriented Search*, Springer, 2018 147 (cit. on p. 39).

[Bas+21] A. Bastos et al., "RECON: Relation Extraction Using Knowledge Graph Context in a Graph Neural Network," *Proceedings of the Web Conference 2021*, WWW '21, Ljubljana, Slovenia: Association for Computing Machinery, 2021 1673, ISBN: 9781450383127, DOI: 10.1145/3442381.3449917 (cit. on pp. 39, 61, 126).

[BK11] F. Bauer and M. Kaltenböck, *Linked open data: The essentials*, Edition mono/monochrom, Vienna **710** (2011) (cit. on p. 21).

[BGL15] R. Beaumont, B. Grau, and A.-L. Ligozat, "SemGraphQA@QALD-5: LIMSI participation at QALD-5@CLEF," 2015 (cit. on p. 36).

[Bec+10] S. Bechhofer, D. De Roure, M. Gamble, C. Goble, and I. Buchan, *Research objects: Towards exchange and reuse of digital knowledge*, IRCDL (2010) (cit. on p. 32).

[BCL19] I. Beltagy, A. Cohan, and K. Lo, *SciBERT: Pretrained Contextualized Embeddings for Scientific Text*, CoRR **abs/1903.10676** (2019) (cit. on p. 48).

[BLC19] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A Pretrained Language Model for Scientific Text," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics, 2019 3615 (cit. on p. 97).

[BPC20] I. Beltagy, M. E. Peters, and A. Cohan, *Longformer: The long-document transformer*, arXiv preprint arXiv:2004.05150 (2020) (cit. on pp. 35, 41, 90).

[Ber] T. Berners-Lee, *Linked Data*, https://www.w3.org/DesignIssues/LinkedData.html, (Accessed on 10/06/2020) (cit. on p. 68).

[BHL01a] T. BERNERS-LEE, J. HENDLER, and O. LASSILA, *THE SEMANTIC WEB*, Scientific American **284** (2001) 34, ISSN: 00368733, 19467087, URL: http://www.jstor.org/stable/26059207 (visited on 05/09/2022) (cit. on p. 15).

[BHL01b] T. Berners-Lee, J. Hendler, and O. Lassila, *The semantic web*, Scientific american **284** (2001) 34 (cit. on p. 61).

[BMC+04] P. V. Biron, A. Malhotra, W. W. W. Consortium, et al., *XML schema part 2: Datatypes*, 2004 (cit. on p. 16).

[Blo+07] S. Bloehdorn et al., "Ontology-based question answering for digital libraries," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4675 LNCS, Springer, Berlin, Heidelberg, 2007 14, ISBN: 9783540748502, DOI: 10.1007/978-3-540-74851-9_2 (cit. on p. 35).

[Bod04] O. Bodenreider,
The Unified Medical Language System (UMLS): integrating biomedical terminology,
Nucleic Acids Research **32** (2004) D267, ISSN: 0305-1048 (cit. on pp. 49, 101).

[Boj+17] 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,
ISSN: 2307-387X (cit. on pp. 37, 47).

[Bor+13] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko,
"Translating Embeddings for Modeling Multi-relational Data,"
Advances in Neural Information Processing Systems,
ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger,
vol. 26, Curran Associates, Inc., 2013 (cit. on pp. 37, 95, 96, 101–104, 107).

[BM15a] L. Bornmann and R. Mutz, *Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references*,
Journal of the Association for Information Science and Technology **66** (2015)
(cit. on p. 43).

[BM15b] L. Bornmann and R. Mutz, *Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references*,
Journal of the Association for Information Science and Technology **66** (2015) 2215,
DOI: 10.1002/asi.23329 (cit. on p. 111).

[Bos+] J. Bosman et al.,
The Scholarly Commons - principles and practices to guide research communication (),
DOI: 10.31219/OSF.IO/6C2XT (cit. on pp. 43, 111).

[Bos+19] A. Bosselut et al.,
Comet: Commonsense transformers for automatic knowledge graph construction,
arXiv preprint arXiv:1906.05317 (2019) (cit. on p. 37).

[Bot+16] A. Both et al., "Qanary - A Methodology for Vocabulary-Driven Open Question Answering Systems," vol. 9678, 2016 625 (cit. on p. 68).

[BFY08] B. Boyan, R. Femke, and Q. Yi,
SKIing with DOLCE: toward an e-Science Knowledge Infrastructure,
Frontiers in Artificial Intelligence and Applications **183** (2008) 208, ISSN: 0922-6389
(cit. on p. 32).

[BK16] J. S. Brennen and D. Kreiss, *Digitalization*,
The international encyclopedia of communication theory and philosophy (2016) 1
(cit. on p. 1).

[BG14] D. Brickley and R. Guha,
RDF Schema 1.1 - W3C Recommendation 25 February 2014,
World Wide Web Consortium (W3C), 2014,
URL: http://www.w3.org/TR/rdf-schema/ (cit. on p. 46).

[Bro+20] T. B. Brown et al., *Language Models are Few-Shot Learners*, 2020,
arXiv: 2005.14165 [cs.CL] (cit. on p. 95).

[Bur+17] A. Burton et al., *The Scholix Framework for Interoperability in Data-Literature Information Exchange*, D-Lib Magazine **Volume 23** (2017), published (cit. on p. 32).

[CZC18] H. Cai, V. W. Zheng, and K. C.-C. Chang, *A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications*, IEEE Transactions on Knowledge and Data Engineering **30** (2018) 1616 (cit. on pp. 35, 95).

[Cap+17] S. Capadisli et al., "Decentralised Authoring, Annotations and Notifications for a Read-Write Web with dokieli," *International Conference on Web Engineering*, 2017 469 (cit. on pp. 32, 33).

[Car+05] J. J. Carroll, C. Bizer, P. Hayes, and P. Stickler, *Named graphs*, Journal of Web Semantics **3** (2005) 247 (cit. on p. 22).

[Cet+18] M. Cetto, C. Niklaus, A. Freitas, and S. Handschuh, "Graphene: Semantically-Linked Propositions in Open Information Extraction," *Proceedings of the 27th COLING*, 2018 2300 (cit. on pp. 39, 71, 72, 76).

[CPB] A. T. Chaganty, A. Paranjape, and J. Bolton et al., *Stanford at TAC KBP 2017: Building a Trilingual Relational Knowledge Graph.* () (cit. on p. 71).

[CHA20] Y. CHAI., *Evaluation metrics of Name Entity Recognition systems*, 2020, URL: https://ychai.uk/notes/2018/11/21/NLP/NER/Evaluation-metrics-of-Name-Entity-Recognition-systems/ (cit. on p. 72).

[Cha+06] C.-H. Chang, M. Kayed, M. R. Girgis, and K. F. Shaalan, *A survey of web information extraction systems*, IEEE transactions on knowledge and data engineering **18** (2006) 1411 (cit. on p. 83).

[Cha+21] L. Chao, J. He, T. Wang, and W. Chu, *PairRE: Knowledge Graph Embeddings via Paired Relation Vectors*, 2021, arXiv: 2011.03798 [cs.CL] (cit. on p. 104).

[Che+18] M. Cheatham et al., *The GeoLink knowledge graph*, Big Earth Data **2** (2018) 131 (cit. on p. 32).

[CLF09] A. Chebotko, S. Lu, and F. Fotouhi, *Semantics preserving SPARQL-to-SQL translation*, Data & Knowledge Engineering **68** (2009) 973 (cit. on p. 20).

[CY93] C. Chen and G. You, *Class sensitive neural networks*, Neural Parallel Sci. Comput. **1** (1993) 93 (cit. on p. 67).

[Che+20] Z. Chen et al., *Knowledge graph completion: A review*, Ieee Access **8** (2020) 192435 (cit. on p. 6).

[Che+17] J. Cheng, S. Reddy, V. Saraswat, and M. Lapata, "Learning Structured Natural Language Representations for Semantic Parsing," *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2017 44, DOI: 10.18653/v1/P17-1005 (cit. on p. 34).

[Cho+14] K. Cho et al., "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2014 1724 (cit. on p. 35).

[Cho20] K. R. Chowdhary, "Natural Language Processing," *Fundamentals of Artificial Intelligence*, New Delhi: Springer India, 2020 603, ISBN: 978-81-322-3972-7, DOI: 10.1007/978-81-322-3972-7_19 (cit. on p. 23).

[Chr+18] V. Christen et al., "A Learning-Based Approach to Combine Medical Annotation Results - (Short Paper).," *DILS*, ed. by S. Auer and M.-E. Vidal, vol. 11371, Lecture Notes in Computer Science, Springer, 2018 135, ISBN: 978-3-030-06016-9 (cit. on pp. 163, 164).

[CD21] F. C. Chua and N. P. Duffy, "DeepCPCFG: Deep Learning and Context Free Grammars for End-to-End Information Extraction," *Document Analysis and Recognition – ICDAR 2021*, ed. by J. Lladós, D. Lopresti, and S. Uchida, Cham: Springer International Publishing, 2021 838, ISBN: 978-3-030-86331-9 (cit. on pp. 40, 41).

[Cla+20] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators," *ICLR*, 2020 (cit. on p. 95).

[CM16] K. Clark and C. D. Manning, "Deep Reinforcement Learning for Mention-Ranking Coreference Models," *Proceedings of the 2016 EMNLP*, 2016 2256 (cit. on pp. 39, 40, 71, 72, 76).

[CCG14] T. Clark, P. N. Ciccarese, and C. A. Goble, *Micropublications: a semantic model for claims, evidence, arguments and annotations in biomedical communications*, Journal of Biomedical Semantics **5** (2014) (cit. on pp. 32, 33).

[Cle+19] C. B. Clement, M. Bierbaum, K. P. O'Keeffe, and A. A. Alemi, *On the Use of ArXiv as a Dataset*, 2019, arXiv: 1905.00075 [cs.IR] (cit. on p. 41).

[CK22] E. Cortes and O. Karras, *Question Answering over linked data benchmark comparison*, en, 2022, DOI: 10.48366/R161787, URL: https://www.orkg.org/orkg/comparison/R161787 (cit. on p. 33).

[Cui+14] W. Cui, S. Liu, Z. Wu, and H. Wei, *How Hierarchical Topics Evolve in Large Text Corpora*, IEEE TVCG **20** (2014) 2281 (cit. on p. 61).

[Cui+11] W. Cui et al., *TextFlow: Towards Better Understanding of Evolving Topics in Text*, IEEE TVCG **17** (2011) 2412 (cit. on p. 61).

[Cui+17] W. Cui et al.,
KBQA: Learning Question Answering over QA Corpora and Knowledge Bases,
Proc. VLDB Endow. **10** (2017) 565, ISSN: 2150-8097 (cit. on p. 95).

[DSo+20] J. D'Souza et al.,
"The STEM-ECR Dataset: Grounding Scientific Entity References in STEM Scholarly Content to Authoritative Encyclopedic and Lexicographic Sources," English,
Proceedings of the 12th Language Resources and Evaluation Conference,
Marseille, France: European Language Resources Association, 2020 2192,
ISBN: 979-10-95546-34-4,
URL: https://aclanthology.org/2020.lrec-1.268 (cit. on p. 51).

[Dai+13] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes,
"Improving Efficiency and Accuracy in Multilingual Entity Extraction,"
Proceedings of the 9th I-Semantics, 2013 (cit. on pp. 39, 40, 64, 71).

[Das+21] P. Dasigi et al., "A Dataset of Information-Seeking Questions and Answers Anchored in Research Papers,"
Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,
Online: Association for Computational Linguistics, 2021 4599,
DOI: 10.18653/v1/2021.naacl-main.365,
URL: https://aclanthology.org/2021.naacl-main.365
(cit. on pp. 40, 41).

[De +06] A. De Waard, L. Breure, J. G. Kircz, and H. Van Oostendorp,
"Modeling rhetoric in scientific publications," *International Conference on Multidisciplinary Information Sciences and Technologies, InSciT2006*, 2006
(cit. on p. 32).

[DG13] L. Del Corro and R. Gemulla, "ClausIE: Clause-Based Open Information Extraction,"
Proceedings of the 22nd International Conference on World Wide Web, WWW '13,
ACM, 2013 355 (cit. on pp. 40, 71).

[Del+19] P. L. Delamater, E. J. Street, T. F. Leslie, Y. T. Yang, and K. H. Jacobsen,
Complexity of the basic reproduction number (Ro),
Emerging infectious diseases **25** (2019) 1 (cit. on p. 96).

[Del19] A. Delpeuch, *OpenTapioca: Lightweight Entity Linking for Wikidata*, 2019,
arXiv: 1904.09131 [cs.CL] (cit. on pp. 39, 61, 71).

[Der+15] L. Derczynski et al., *Analysis of named entity recognition and linking for tweets*,
Information Processing & Management **51** (2015) 32 (cit. on pp. 75, 78).

[Des+20] D. Dessì et al.,
"AI-KG: An Automatically Generated Knowledge Graph of Artificial Intelligence,"
The Semantic Web – ISWC 2020, ed. by J. Z. Pan et al.,
Springer International Publishing, 2020 127, ISBN: 978-3-030-62466-8 (cit. on p. 97).

[Det+18] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 1, 2018 (cit. on pp. 95, 101).

[Dev+19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2019 4171, DOI: 10.18653/v1/N19-1423 (cit. on pp. 34, 37, 39, 41, 90, 95, 97, 99).

[Die+18] 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, ISSN: 02193116, DOI: 10.1007/s10115-017-1100-y (cit. on p. 34).

[Die+17] D. Diefenbach, T. P. Tanon, K. D. Singh, and P. Maret, "Question Answering Benchmarks for Wikidata," *SEMWEB*, 2017 (cit. on p. 33).

[Die+19] D. Diefenbach et al., "QAnswer: A question answering prototype bridging the gap between a considerable part of the LOD cloud and end-users," *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, Association for Computing Machinery, Inc, 2019 3507, ISBN: 9781450366748, DOI: 10.1145/3308558.3314124 (cit. on p. 34).

[Don+19] T. Dong, Z. Wang, J. Li, C. Bauckhage, and A. B. Cremers, "Triple classification using regions and fine-grained entity typing," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019 77 (cit. on p. 61).

[Don+14] X. Dong et al., "Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion," *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, Association for Computing Machinery, 2014 601, ISBN: 9781450329569 (cit. on pp. 102, 107).

[DSM15] P. Donohoe, J. Sherman, and A. Mistry, "The Long Road to JATS," *Journal Article Tag Suite Conference (JATS-Con) Proceedings 2015 [Internet]*, Bethesda (MD): National Center for Biotechnology Information (US), 2015 (cit. on p. 32).

[DKV13] M. Dua, S. Kumar, and Z. S. Virk, "Hindi language graphical user interface to database management system," *Proceedings - 2013 12th International Conference on Machine Learning and Applications, ICMLA 2013*, vol. 2, IEEE Computer Society, 2013 555, DOI: 10.1109/ICMLA.2013.176 (cit. on p. 34).

[Dub+19] M. Dubey, D. Banerjee, A. Abdelkawi, and J. Lehmann, "LC-QuAD 2.0: A Large Dataset for Complex Question Answering over Wikidata and DBpedia," *Proceedings of the 18th ISWC*, 2019 (cit. on p. 33).

[Dub+18a] M. Dubey, D. Banerjee, D. Chaudhuri, and J. Lehmann, "EARL: Joint Entity and Relation Linking for Question Answering over Knowledge Graphs," *Lecture Notes in Computer Science*, Springer International Publishing, 2018 108 (cit. on pp. 39, 40, 61, 71, 78).

[Dub+18b] M. Dubey, D. Banerjee, D. Chaudhuri, and J. Lehmann, "EARL: Joint Entity and Relation Linking for Question Answering over Knowledge Graphs," *The Semantic Web – ISWC 2018*, Cham: Springer International Publishing, 2018 108, ISBN: 978-3-030-00671-6 (cit. on p. 58).

[DS05] M. Dürst and M. Suignard, *Internationalized resource identifiers (IRIs)*, tech. rep., RFC 3987, January, 2005 (cit. on p. 16).

[Edd04] S. R. Eddy, *What is a hidden Markov model?* Nature biotechnology **22** (2004) 1315 (cit. on p. 25).

[Edw+13] P. N. Edwards et al., *Knowledge Infrastructures: Intellectual Frameworks and Research Challenges*, tech. rep., University of Michigan School of Information, 2013 (cit. on p. 45).

[ElS+18] H. ElSahar et al., "T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples," *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. 2018 (cit. on p. 69).

[Etz+08] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld, *Open information extraction from the web*, Communications of the ACM **51** (2008) 68 (cit. on pp. 29, 83).

[FGG+07] M. Fabian, K. Gjergji, W. Gerhard, et al., "Yago: A core of semantic knowledge unifying wordnet and wikipedia," *WWW*, 2007 697 (cit. on p. 61).

[FSE11] A. Fader, S. Soderland, and O. Etzioni, "Identifying Relations for Open Information Extraction," *Proceedings of the 2011 EMNLP*, 2011 1535 (cit. on pp. 64, 71).

[Fär19] M. Färber, "The microsoft academic knowledge graph: a linked data source with 8 billion triples of scholarly data," *International Semantic Web Conference*, Springer, 2019 113 (cit. on p. 97).

[Fat+17] S. Fathalla, S. Vahdati, S. Auer, and C. Lange, "Towards a Knowledge Graph Representing Research Findings by Semantifying Survey Articles," *Research and Advanced Technology for Digital Libraries*, 2017 315 (cit. on p. 32).

[Fen+11] D. Fensel, F. M. Facca, E. Simperl, and I. Toma, "Semantic Web," *Semantic Web Services*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011 87, ISBN: 978-3-642-19193-0, DOI: 10.1007/978-3-642-19193-0_6 (cit. on p. 16).

[FS10a] P. Ferragina and U. Scaiella, "TAGME: on-the-fly annotation of short text fragments (by wikipedia entities)," 2010 1625 (cit. on pp. 39, 61, 71, 72).

[FS10b] P. Ferragina and U. Scaiella, "TAGME: on-the-fly annotation of short text fragments (by wikipedia entities).," *CIKM*, ed. by J. Huang et al., ACM, 2010 1625, ISBN: 978-1-4503-0099-5 (cit. on p. 58).

[FT95] S. Fredrickson and L. Tarassenko, *Text-independent speaker recognition using neural network techniques* (1995) (cit. on p. 67).

[FBH] A. Freitas, B. Bermeitinger, and S. Handschuh, *Lambda-3/PyCobalt: Coreference Resolution in Python*, https://github.com/Lambda-3/PyCobalt (cit. on p. 76).

[FC14] A. Freitas and E. Curry, "Natural Language Queries over Heterogeneous Linked Data Graphs: A Distributional-Compositional Semantics Approach," *Proceedings of the 19th International Conference on Intelligent User Interfaces*, IUI '14, Haifa, Israel: Association for Computing Machinery, 2014 279, ISBN: 9781450321846, DOI: 10.1145/2557500.2557534 (cit. on p. 36).

[Fri+18] A. Friedrich, L. de la Garza, O. Kohlbacher, and S. Nahnsen, "Interactive Visualization for Large-Scale Multi-factorial Research Designs," *Data Integration in the Life Sciences - 13th International Conference, DILS 2018, Hannover, Germany, November 20-21, 2018, Proceedings*, 2018 75 (cit. on pp. 163, 164).

[Gar+17] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "Creating Training Corpora for NLG Micro-Planners," 2017 179 (cit. on p. 69).

[GGC17] K. Gashteovski, R. Gemulla, and L. del Corro, "MinIE: Minimizing Facts in Open Information Extraction," *Proceedings of the 2017 EMNLP*, 2017 2630 (cit. on pp. 39, 71).

[GBS19] G. A. Gesese, R. Biswas, and H. Sack, "A Comprehensive Survey of Knowledge Graph Embeddings with Literals: Techniques and Applications.," *DL4KG@ ESWC*, 2019 31 (cit. on p. 95).

[GGH09] K. Goel, R. V. Guha, and O. Hansson, *Introducing Rich Snippets | Google Search Central Blog | Google Developers*, https://developers.google.com/search/blog/2009/05/introducing-rich-snippets, (Accessed on 05/10/2022), 2009 (cit. on p. 18).

[Gon+19] Y. Gong et al., "Deep cascade multi-task learning for slot filling in online shopping assistant," *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 01, 2019 6465 (cit. on p. 24).

[Gre00] E. Greengrass, *Information retrieval: A survey* (2000), URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.33.1855 (cit. on p. 3).

[Gri15] R. Grishman, *Information extraction*, IEEE Intelligent Systems **30** (2015) 8 (cit. on p. 38).

[GGV10] P. Groth, A. Gibson, and J. Velterop, *The anatomy of a nanopublication*, Information Services and Use **30** (2010) 51, ISSN: 01675265, DOI: 10.3233/ISU-2010-0613 (cit. on pp. 32, 33, 115).

[Gro19] M. P. W. Group, *Stakeholder consultation on the future of scholarly publishing and scholarly communication* (2019), DOI: 10.5281/zenodo.3246729 (cit. on p. 44).

[Gro+07] T. Groza, S. Handschuh, K. Möller, and S. Decker, "SALT - Semantically Annotated LaTeX for Scientific Publications," *Extended Semantic Web Conference*, 2007 518 (cit. on p. 32).

[GBM16] R. V. Guha, D. Brickley, and S. Macbeth, *Schema. org: evolution of structured data on the web*, Communications of the ACM **59** (2016) 44 (cit. on p. 18).

[GG12] P. Gupta and V. Gupta, *A survey of text question answering techniques*, International Journal of Computer Applications **53** (2012) (cit. on p. 23).

[Ham+14] T. Hamon, N. Grabar, F. Mougin, and F. Thiessard, "Description of the POMELO System for the Task 2 of QALD-2014," *CLEF*, 2014 (cit. on p. 36).

[HDD15] K. L. Hanson, T. DiLauro, and M. Donoghue, "The RMap Project: Capturing and Preserving Associations Amongst Multi-Part Distributed Publications," *Proceedings of the 15th ACM/IEEE-CE JCDL*, ACM, 2015 281 (cit. on p. 32).

[Har+21] M. Haris, K. E. Farfar, M. Stocker, and S. Auer, "Federating Scholarly Infrastructures with GraphQL," *Towards Open and Trustworthy Digital Societies*, ed. by H.-R. Ke, C. S. Lee, and K. Sugiyama, Cham: Springer International Publishing, 2021 308, ISBN: 978-3-030-91669-5 (cit. on p. 46).

[Har01] A. Hars, *Designing Scientific Knowledge Infrastructures: The Contribution of Epistemology.*, Information Systems Frontiers **3** (2001) 63 (cit. on pp. 32, 43, 59).

[HP14] P. J. Hayes and P. F. Patel-Schneider, *RDF 1.1 Semantics - RDFS Entailment*, https://www.w3.org/TR/rdf11-mt/#rdfs-entailment, (Accessed on 05/09/2022), 2014 (cit. on p. 17).

[He+15] S. He, K. Liu, G. Ji, and J. Zhao, "Learning to Represent Knowledge Graphs with Gaussian Embedding," *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, Association for Computing Machinery, 2015 623, ISBN: 9781450337946 (cit. on pp. 102, 104, 107).

[He+14] S. He, Y. Zhang, K. Liu, and J. Zhao, "CASIA@V2: A MLN-based Question Answering System over Linked Data," *CLEF*, 2014 (cit. on p. 36).

[Hei+21] G. Heidari, A. Ramadan, M. Stocker, and S. Auer, "Leveraging a Federation of Knowledge Graphs to Improve Faceted Search in Digital Libraries," *Linking Theory and Practice of Digital Libraries*, ed. by G. Berget, M. M. Hall, D. Brenn, and S. Kumpulainen, Cham: Springer International Publishing, 2021 141, ISBN: 978-3-030-86324-1, DOI: http://dx.doi.org/10.1007/978-3-030-86324-1_18 (cit. on p. 46).

[Hen+18] V. Henry et al., "Converting Alzheimer's Disease Map into a Heavyweight Ontology: A Formal Network to Integrate Data.," *DILS*, ed. by S. Auer and M.-E. Vidal, vol. 11371, Lecture Notes in Computer Science, Springer, 2018 207, ISBN: 978-3-030-06016-9 (cit. on pp. 163, 164).

[Her06] W. R. Hersh, "Information Retrieval and Digital Libraries," *Medical Informatics*, Kluwer Academic Publishers, 2006 237, DOI: 10.1007/0-387-25739-X_9 (cit. on p. 35).

[HTT+09] A. J. Hey, S. Tansley, K. M. Tolle, et al., *The fourth paradigm: data-intensive scientific discovery*, vol. 1, Microsoft research Redmond, WA, 2009 (cit. on p. 95).

[Hof+11] J. Hoffart et al., "Robust Disambiguation of Named Entities in Text," *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, Edinburgh, United Kingdom: Association for Computational Linguistics, 2011 782, ISBN: 978-1-937284-11-4 (cit. on pp. 61, 71).

[Hou+19] Y. Hou, C. Jochim, M. Gleize, F. Bonin, and D. Ganguly, "Identification of Tasks, Datasets, Evaluation Metrics, and Numeric Scores for Scientific Leaderboards Construction," Association for Computational Linguistics (ACL), 2019 5203, DOI: 10.18653/v1/p19-1513 (cit. on p. 61).

[HAV14] B. Hyland, G. Atemezing, and B. Villazón-Terrazas, *Best Practices for Publishing Linked Data*, https://www.w3.org/TR/ld-bp/, (Accessed on 05/10/2022), 2014 (cit. on p. 18).

[Ibr+19] Y. Ibrahim, M. Riedewald, G. Weikum, and D. Zeinalipour-Yazti, "Bridging Quantities in Tables and Text," *2019 IEEE 35th ICDE*, 2019 1010 (cit. on p. 61).

[JPS22] K. M. Jablonka, L. Patiny, and B. Smit, *Making the collective knowledge of chemistry open and machine actionable*, Nature Chemistry **14** (2022) 365 (cit. on p. 5).

[JA19] K. Jacksi and S. Abass, *Development History Of The World Wide Web*, International Journal of Scientific & Technology Research **8** (2019) 75 (cit. on p. 15).

[Jan10] H. Jansen, *The Logic of Qualitative Survey Research and its Position in the Field of Social Research Methods*, en,
Forum Qualitative Sozialforschung / Forum: Qualitative Social Research **11** (2010),
ISSN: 1438-5627, (visited on 01/09/2019) (cit. on p. 57).

[Jar+19a] M. Jaradeh, A. Oelen, M. Prinz, M. Stocker, and S. Auer,
Open Research Knowledge Graph: A System Walkthrough, vol. 11799 LNCS, 2019,
ISBN: 9783030307592, DOI: 10.1007/978-3-030-30760-8_31
(cit. on p. 44).

[Jar22a] M. Y. Jaradeh,
Comparison of Open Information Extraction components - Comparison - ORKG,
https://www.orkg.org/orkg/comparison/R189409/,
(Accessed on 06/09/2022), 2022 (cit. on p. 40).

[Jar22b] M. Y. Jaradeh,
Information Extraction systems and their approaches - Comparison - ORKG,
https://www.orkg.org/orkg/comparison/R189475/,
(Accessed on 06/09/2022), 2022 (cit. on p. 40).

[JO19] M. Y. Jaradeh and A. Oelen, *Question answering systems - Comparison - ORKG*,
https://www.orkg.org/orkg/comparison/R6757,
(Accessed on 06/08/2022), 2019 (cit. on p. 36).

[Jar+21a] M. Y. Jaradeh, K. Singh, M. Stocker, and S. Auer,
"Plumber: A Modular Framework to Create Information Extraction Pipelines,"
Companion Proceedings of the Web Conference 2021,
New York, NY, USA: Association for Computing Machinery, 2021 678,
ISBN: 9781450383134, DOI: 10.1145/3442442.3458603 (cit. on p. 62).

[Jar+21b] M. Y. Jaradeh, K. Singh, M. Stocker, and S. Auer,
"Triple Classification for Scholarly Knowledge Graph Completion,"
Proceedings of the 11th on Knowledge Capture Conference, K-CAP '21,
Virtual Event, USA: Association for Computing Machinery, 2021 225,
ISBN: 9781450384575, DOI: 10.1145/3460210.3493582 (cit. on pp. 40, 41, 95).

[Jar+21c] M. Y. Jaradeh, K. Singh, M. Stocker, A. Both, and S. Auer,
"Better Call the Plumber: Orchestrating Dynamic Information Extraction Pipelines,"
Web Engineering, ed. by M. Brambilla, R. Chbeir, F. Frasincar, and I. Manolescu,
Cham: Springer International Publishing, 2021 240, ISBN: 978-3-030-74296-6,
DOI: 10.1007/978-3-030-74296-6_19 (cit. on p. 62).

[JSA22] M. Y. Jaradeh, M. Stocker, and S. Auer, *MORTY: Structured Summarization for Targeted Information Extraction out of Scholarly Articles*,
Under review at CIKM 2022, 2022 (cit. on p. 83).

[JSA20] M. Y. Jaradeh, M. Stocker, and S. Auer,
"Question Answering on Scholarly Knowledge Graphs,"
Digital Libraries for Open Knowledge,

ed. by M. Hall, T. Merčun, T. Risse, and F. Duchateau,
Cham: Springer International Publishing, 2020 19, ISBN: 978-3-030-54956-5,
DOI: 10.1007/978-3-030-54956-5_2 (cit. on pp. 95, 111).

[Jar+19b] M. Y. Jaradeh et al., *Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge*, Marina Del K-CAP **19** (2019) 243, DOI: 10.1145/3360901.3364435 (cit. on pp. 33, 44, 69, 83, 97, 101, 111, 112, 115).

[JTH16] S. K. Jauhar, P. Turney, and E. Hovy,
TabMCQ: A Dataset of General Knowledge Tables and Multiple-choice Questions (2016), URL: http://arxiv.org/abs/1602.03960 (cit. on p. 113).

[Jes+20] J. Jeschke et al., *Hi-Knowledge, version 2.0*, https://hi-knowledge.org/, (Accessed on 05/23/2022), 2020 (cit. on p. 83).

[Jet11] JetBrains, *The Kotlin Programming Language*, 2011,
URL: https://kotlinlang.org/ (visited on 01/22/2019) (cit. on p. 54).

[Ji+20] D. Ji, P. Tao, H. Fei, and Y. Ren, *An end-to-end joint model for evidence information extraction from court record document*,
Information Processing & Management **57** (2020) 102305, ISSN: 0306-4573,
DOI: 10.1016/j.ipm.2020.102305 (cit. on pp. 40, 41).

[Ji+15] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao,
"Knowledge Graph Embedding via Dynamic Mapping Matrix,"
Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, 2015 687 (cit. on pp. 101, 104, 107).

[Ji+16] G. Ji, K. Liu, S. He, and J. Zhao,
"Knowledge Graph Completion with Adaptive Sparse Transfer Matrix,"
Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, AAAI Press, 2016 985 (cit. on p. 95).

[Jia12] J. Jiang, "Information extraction from text," *Mining text data*, Springer, 2012 11 (cit. on p. 83).

[JW81] A. K. Joshi and S. Weinstein,
"Control of Inference: Role of Some Aspects of Discourse Structure-Centering.,"
IJCAI, 1981 385 (cit. on p. 23).

[Jou+16] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov,
Bag of Tricks for Efficient Text Classification, arXiv preprint arXiv:1607.01759 (2016) (cit. on p. 37).

[KLW19] U. Kamath, J. Liu, and J. Whitaker, *Deep learning for NLP and speech recognition*, vol. 84, Springer, 2019 (cit. on p. 23).

[Kar+20] M. Kardas et al.,
"AxCell: Automatic Extraction of Results from Machine Learning Papers," *EMNLP*, 2020 (cit. on p. 40).

[Kar+19] B. Karki et al., *Question Answering via Web Extracted Tables and Pipelined Models* (2019), URL: http://arxiv.org/abs/1903.07113 (cit. on p. 111).

[KP18] S. M. Kazemi and D. Poole,
"SimplE Embedding for Link Prediction in Knowledge Graphs,"
Advances in Neural Information Processing Systems, ed. by S. Bengio et al., vol. 31, Curran Associates, Inc., 2018 (cit. on pp. 102, 104, 107).

[KI17] N. Kertkeidkachorn and R. Ichise, "T2KG: An End-to-End System for Creating Knowledge Graph from Unstructured Text," *AAAI Workshops*, vol. WS-17, 2017 (cit. on pp. 39, 61, 70, 72–74, 78).

[Ket17] N. Ketkar, "Stochastic Gradient Descent," *Deep Learning with Python*, Apress, 2017 113, DOI: 10.1007/978-1-4842-2766-4_8 (cit. on p. 102).

[Kha15] A. Khalili, *A Semantics-based User Interface Model for Content Annotation, Authoring and Exploration*, 2015, DOI: 10.13140/RG.2.1.1951.5600 (cit. on p. 17).

[Kha+16] W. Khan, A. Daud, J. A. Nasir, and T. Amjad,
A survey on the state-of-the-art machine learning models in the context of NLP,
Kuwait journal of Science **43** (2016) (cit. on p. 25).

[Kim+17] J.-D. Kim et al., *OKBQA Framework for collaboration on developing natural language question answering systems* (2017) (cit. on p. 61).

[KB17] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, 2017, arXiv: 1412.6980 [cs.LG] (cit. on p. 102).

[KW17] T. N. Kipf and M. Welling,
"Semi-Supervised Classification with Graph Convolutional Networks,"
Proceedings of the 5th International Conference on Learning Representations (ICLR), ICLR '17, 2017 (cit. on p. 35).

[KT00] M. Kobayashi and K. Takeda, *Information Retrieval on the Web*,
ACM Comput. Surv. **32** (2000) 144, ISSN: 0360-0300,
DOI: 10.1145/358923.358934 (cit. on p. 3).

[KK13] J. Krishnamurthy and T. Kollar, *Jointly Learning to Parse and Perceive: Connecting Natural Language to the Physical World*,
Transactions of the Association for Computational Linguistics **1** (2013) 193,
DOI: 10.1162/tacl_a_00220 (cit. on p. 34).

[KC02] C. Kwak and A. Clayton-Matthews, *Multinomial logistic regression*,
Nursing research **51** (2002) 404 (cit. on p. 28).

[Kwi+] T. Kwiatkowski, E. Choi, Y. Artzi, and L. Zettlemoyer,
Scaling Semantic Parsers with On-the-fly Ontology Matching, tech. rep. 1545,
URL: www.wiktionary.com (cit. on p. 34).

[Laa+11] M. Laakso et al., *The development of open access journal publishing from 1993 to 2009*,
PloS one **6** (2011) e20961 (cit. on p. 2).

[Lan+19] Z. Lan et al.,
ALBERT: A Lite BERT for Self-supervised Learning of Language Representations (2019), URL: http://arxiv.org/abs/1909.11942 (cit. on p. 34).

[Lan13] C. Lange, *Ontologies and languages for representing mathematical knowledge on the Semantic Web*, Semantic Web **4** (2013) 119 (cit. on p. 32).

[Les86] M. Lesk, "Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone,"
Proceedings of the 5th Annual International Conference on Systems Documentation, SIGDOC '86,
Toronto, Ontario, Canada: Association for Computing Machinery, 1986 24,
ISBN: 0897912241, DOI: 10.1145/318723.318728 (cit. on p. 23).

[Leu07] K. M. Leung, *Naive bayesian classifier*, Polytechnic University Department of Computer Science/Finance and Risk Engineering **2007** (2007) 123 (cit. on p. 28).

[Lew+19] M. Lewis et al., *Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension*,
arXiv preprint arXiv:1910.13461 (2019) (cit. on pp. 41, 90).

[Lia+20a] S. Liang, K. Stockinger, T. M. de Farias, M. Anisimova, and M. Gil,
Querying Knowledge Graphs in Natural Language (2020) (cit. on pp. 61, 78).

[Lia+20b] J. Liao et al., *Improving readability for automatic speech recognition transcription*,
arXiv preprint arXiv:2004.04438 (2020) (cit. on p. 25).

[Lin02] J. Lin,
"The Web as a Resource for Question Answering: Perspectives and Challenges,"
LREC, Las Palmas, 2002,
URL: https://www.aclweb.org/anthology/L02-1085/
(cit. on p. 111).

[Lin+15] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu,
"Learning Entity and Relation Embeddings for Knowledge Graph Completion,"
Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence,
AAAI'15, AAAI Press, 2015 2181, ISBN: 0262511290 (cit. on pp. 37, 95, 101, 104, 107).

[Liu+18] Y. Liu, T. Zhang, Z. Liang, H. Ji, and D. McGuinness, *Seq2RDF: An End-to-end Application for Deriving Triples from Natural Language Text* (2018) (cit. on p. 39).

[Liu+07] Y. Liu, K. Bai, P. Mitra, and C. L. Giles,
"TableSeer: Automatic Table Metadata Extraction and Searching in Digital Libraries,"
Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries,
JCDL '07, Vancouver, BC, Canada: Association for Computing Machinery, 2007 91,
ISBN: 9781595936448, DOI: 10.1145/1255175.1255193 (cit. on pp. 40, 41).

[Liu+19a] Y. Liu et al., *RoBERTa: A Robustly Optimized BERT Pretraining Approach*, 2019,
arXiv: 1907.11692 [cs.CL] (cit. on pp. 39, 68).

[Liu+19b] Y. Liu et al., *Roberta: A robustly optimized bert pretraining approach*,
arXiv preprint arXiv:1907.11692 (2019) (cit. on pp. 41, 90).

[Lopo9] P. Lopez, "GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications,"
International conference on theory and practice of digital libraries, Springer, 2009 473 (cit. on p. 89).

[Lop+13] V. Lopez, C. Unger, P. Cimiano, and E. Motta,
Evaluating question answering over linked data, Journal of Web Semantics **21** (2013) 3, ISSN: 15708268, DOI: 10.1016/j.websem.2013.05.006 (cit. on p. 34).

[LI97] B.-L. Lu and M. Ito, "Task decomposition based on class relations: A modular neural network architecture for pattern classification," 1997 330, ISBN: 978-3-540-69074-0 (cit. on p. 67).

[M21] O. M, *Evaluation Framework for Information Extraction - CSE Developer Blog*, https://devblogs.microsoft.com/cse/2021/01/13/evaluation-framework-for-information-extraction/, (Accessed on 05/11/2022), 2021 (cit. on p. 27).

[Ma+21] T. Ma et al., *T-bertsum: Topic-aware text summarization based on bert*,
IEEE Transactions on Computational Social Systems (2021) (cit. on p. 85).

[MH16] X. Ma and E. H. Hovy,
End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF,
CoRR **abs/1603.01354** (2016) (cit. on p. 48).

[MGN18] B. Malone, A. García-Durán, and M. Niepert,
"Knowledge Graph Completion to Predict Polypharmacy Side Effects.," *DILS*, ed. by S. Auer and M.-E. Vidal, vol. 11371, Lecture Notes in Computer Science, Springer, 2018 144, ISBN: 978-3-030-06016-9 (cit. on pp. 163, 164).

[Mal+] S. Malyshev, M. Krötzsch, L. González, J. Gonsior, and A. Bielefeldt,
Getting the Most Out of Wikidata () (cit. on p. 69).

[MMM+04] F. Manola, E. Miller, B. McBride, et al., *RDF primer*,
W3C recommendation **10** (2004) 6 (cit. on pp. 16, 101).

[Mar+14] E. Marx et al., "Towards an open question answering architecture,"
ACM International Conference Proceeding Series, vol. 2014-September, September, Association for Computing Machinery, 2014 57,
DOI: 10.1145/2660517.2660519 (cit. on p. 34).

[Mau+12] Mausam, M. Schmitz, S. Soderland, R. Bart, and O. Etzioni,
"Open Language Learning for Information Extraction,"
Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, ACL, 2012 523 (cit. on p. 71).

[McC+19] J. P. McCrae et al., *The linked open data cloud*, Lod-cloud. net (2019) (cit. on p. 21).

[McP09] S. S. McPherson, *Tim Berners-Lee: Inventor of the World Wide Web*, 1st,
USA: Twenty First Century Books, 2009, ISBN: 0822572737 (cit. on p. 15).

[Mei17] V. G. Meister, "Towards a Knowledge Graph for a Research Group with Focus on Qualitative Analysis of Scholarly Papers," *Proceedings of the First Workshop on Enabling Open Semantic Science co-located with 16th International Semantic Web Conference (ISWC 2017)*, 2017 71 (cit. on p. 32).

[Mes+19] F. Mesquita, M. Cannaviccio, J. Schmidek, P. Mirza, and D. Barbosa, "KnowledgeNet: A Benchmark Dataset for Knowledge Base Population," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, ACL, 2019 749 (cit. on pp. 70, 72–74, 78).

[Mih+20] N. Mihindukulasooriya et al., *Leveraging Semantic Parsing for Relation Linking over Knowledge Bases*, ISWC (2020) (cit. on p. 78).

[MDP09] A. Mikhailian, T. Dalmas, and R. Pinchuk, "Learning Foci for Question Answering over Topic Maps," *ACL-IJCNLP*, 2009 (cit. on p. 119).

[Mik+13] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, Curran Associates Inc., 2013 3111 (cit. on p. 37).

[Mon05] B. Mons, *Which gene did you mean?* BMC Bioinformatics **6** (2005) 142, ISSN: 1471-2105, DOI: 10.1186/1471-2105-6-142 (cit. on p. 45).

[Moo98] G. E. Moore, *Cramming more components onto integrated circuits*, Proceedings of the IEEE **86** (1998) 82 (cit. on p. 24).

[MRN14] A. Moro, A. Raganato, and R. Navigli, *Entity Linking meets Word Sense Disambiguation: a Unified Approach.*, TACL **2** (2014) 231 (cit. on p. 58).

[Mul+20] I. O. Mulang' et al., "Evaluating the impact of knowledge graph context on entity disambiguation models," *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020 2157 (cit. on pp. 97, 126).

[Nak+05] T. Nakayama, N. Hirai, S. Yamazaki, and M. Naito, *Adoption of structured abstracts by general medical journals and format for a structured abstract*, en, J Med Libr Assoc **93** (2005) 237 (cit. on p. 85).

[NJM18] Z. Nasar, S. W. Jaffry, and M. K. Malik, *Information extraction from scientific articles: a survey*, Scientometrics **117** (2018) 1931 (cit. on p. 83).

[Nay+19] M. Nayyeri, S. Vahdati, J. Lehmann, and H. S. Yazdi, *Soft marginal transe for scholarly knowledge graph completion*, arXiv preprint arXiv:1904.12211 (2019) (cit. on p. 37).

[Ngu+18] D. Q. Nguyen, T. D. Nguyen, D. Q. Nguyen, and D. Phung, "A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Association for Computational Linguistics, 2018 327 (cit. on pp. 101, 104, 107).

[NBS14] 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*, 2014 759 (cit. on p. 22).

[NRP16] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, 1, 2016 (cit. on pp. 102, 104, 107).

[NTK11] M. Nickel, V. Tresp, and H.-P. Kriegel, "A Three-Way Model for Collective Learning on Multi-Relational Data," *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, Omnipress, 2011 809, ISBN: 9781450306195 (cit. on p. 37).

[Nik+18] C. Niklaus, M. Cetto, A. Freitas, and S. Handschuh, "A Survey on Open Information Extraction," *Proceedings of the 27th COLING*, 2018 3866 (cit. on p. 39).

[Oel22] A. Oelen, *Semantic representations of scholarly communication*, en, 2022, DOI: 10.48366/R8364, URL: https://www.orkg.org/orkg/comparison/R8364 (cit. on p. 33).

[Oel+21] A. Oelen, M. Y. Jaradeh, M. Stocker, and S. Auer, "Chapter 10. Organizing Scholarly Knowledge leveraging Crowdsourcing, Expert Curation and Automated Techniques," *Linking Knowledge*, Ergon – ein Verlag in der Nomos Verlagsgesellschaft, 2021 181, DOI: 10.5771/9783956506611-181 (cit. on p. 44).

[Oel+20] A. Oelen, M. Y. Jaradeh, M. Stocker, and S. Auer, *Generate FAIR Literature Surveys with Scholarly Knowledge Graphs*, JCDL '20: The 20th ACM/IEEE Joint Conference on Digital Libraries (2020), DOI: 10.1145/3383583.3398520 (cit. on pp. 44, 50, 113).

[PHH18] R. W. Palmatier, M. B. Houston, and J. Hulland, *Review articles: purpose, process, and structure*, Journal of the Academy of Marketing Science **46** (2018) 1, ISSN: 1552-7824, DOI: 10.1007/s11747-017-0563-4 (cit. on p. 83).

[Pan+22] B. Pang et al., *Long Document Summarization with Top-down and Bottom-up Inference*, arXiv preprint arXiv:2203.07586 (2022) (cit. on p. 91).

[PSL14] S. Park, H. Shim, and G. G. Lee, "ISOFT at QALD-4: Semantic Similarity-based Question Answering System over Linked Data," *CLEF*, 2014 (cit. on p. 36).

[PSM14] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2014 1532 (cit. on p. 37).

[Per14] S. Peroni, "The Semantic Publishing and Referencing Ontologies," *Semantic Web Technologies and Legal Scholarly Publishing*, vol. 15, Law, Governance and Technology, Springer, Cham, 2014 121, ISBN: 978-3-319-04777-5 (cit. on p. 32).

[PS20] S. Peroni and D. Shotton, *OpenCitations, an infrastructure organization for open scholarship*, Quantitative Science Studies **1** (2020) 1, DOI: 10.1162/qss_a_00023 (cit. on p. 35).

[Per+17] S. Peroni et al., *Research Articles in Simplified HTML: A Web-first format for HTML-based scholarly articles*, PeerJ Computer Science **2017** (2017), ISSN: 23765992, DOI: 10.7717/peerj-cs.132 (cit. on pp. 32, 33, 115).

[PAS14] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online Learning of Social Representations," *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, Association for Computing Machinery, 2014 701, ISBN: 9781450329569 (cit. on p. 35).

[Pet+18] M. Peters et al., "Deep Contextualized Word Representations," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, 2018 2227 (cit. on p. 37).

[PY13] J. Piskorski and R. Yangarber, "Information extraction: Past, present and future," *Multi-source, multilingual information extraction and summarization*, Springer, 2013 23 (cit. on p. 83).

[PDW18] M. Ponza, L. Del Corro, and G. Weikum, "Facts That Matter," *Proceedings of the 2018 EMNLP*, ACL, 2018 1043 (cit. on pp. 39, 40).

[PS08] E. Prud'hommeaux and A. Seaborne, *SPARQL Query Language for RDF*, https://www.w3.org/TR/rdf-sparql-query/, (Accessed on 05/10/2022), 2008 (cit. on p. 19).

[Rad+18] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, *Improving language understanding by generative pre-training* (2018) (cit. on p. 37).

[Rad+19] A. Radford et al., *Language models are unsupervised multitask learners*, OpenAI blog **1** (2019) 9 (cit. on pp. 41, 90).

[Raf+19] C. Raffel et al., *Exploring the limits of transfer learning with a unified text-to-text transformer*, arXiv preprint arXiv:1910.10683 (2019) (cit. on pp. 41, 90).

[RLJ18] D. Raggett, A. Le Hors, and I. Jacobs, *HTML 4.01 Specification*, https://www.w3.org/TR/html40/, (Accessed on 05/09/2022), 2018 (cit. on p. 15).

[Rag+10] K. Raghunathan et al., "A Multi-Pass Sieve for Coreference Resolution," *EMNLP*, 2010 (cit. on pp. 39, 71).

[RJL18] P. Rajpurkar, R. Jia, and P. Liang, *Know what you don't know: Unanswerable questions for SQuAD*, arXiv preprint arXiv:1806.03822 (2018) (cit. on p. 91).

[Raj+16] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuad: 100,000+ questions for machine comprehension of text," *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, Association for Computational Linguistics (ACL), 2016 2383, ISBN: 9781945626258, DOI: 10.18653/v1/d16-1264 (cit. on p. 34).

[Ram+03] J. Ramos et al., "Using tf-idf to determine word relevance in document queries," *Proceedings of the first instructional conference on machine learning*, vol. 242, Piscataway, NJ, 2003 133 (cit. on pp. 3, 34, 48).

[RMG15] S. Ray Choudhury, P. Mitra, and C. L. Giles, "Automatic extraction of figures from scholarly documents," *Proceedings of the 2015 ACM Symposium on Document Engineering*, 2015 47 (cit. on p. 85).

[Res07] D. B. Resnik, *The price of truth: How money affects the norms of science*, Oxford University Press, 2007 (cit. on p. 1).

[RP16] P. Ristoski and H. Paulheim, "Rdf2vec: Rdf graph embeddings for data mining," *International Semantic Web Conference*, Springer, 2016 498 (cit. on p. 37).

[RR15] A. Roshdi and A. Roohparvar, *Information retrieval techniques and applications*, International Journal of Computer Networks and Communications Security **3** (2015) 373 (cit. on p. 3).

[Sad+17] A. Sadeghi, C. Lange, M.-E. Vidal, and S. Auer, "Integration of Scholarly Communication Metadata Using Knowledge Graphs," *Research and Advanced Technology for Digital Libraries*, 2017 328 (cit. on p. 32).

[SK21] T. Safavi and D. Koutra, *Relational world knowledge representation in contextual language models: A review*, arXiv preprint arXiv:2104.05837 (2021) (cit. on p. 97).

[Sak+20] A. Sakor, K. Singh, A. Patel, and M.-E. Vidal, "Falcon 2.0: An Entity and Relation Linking Tool over Wikidata," *CIKM*, 2020 (cit. on pp. 39, 40, 61, 71, 72).

[Sak+19] A. Sakor et al., "Old is Gold: Linguistic Driven Approach for Entity and Relation Linking of Short Text," ACL, 2019 2336 (cit. on pp. 39, 58, 61, 64, 69, 71, 72, 78).

[SD03] E. F. Sang and F. De Meulder, *Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition*, arXiv preprint cs/0306050 (2003) (cit. on p. 91).

[SWR19] V. Sanh, T. Wolf, and S. Ruder, *A Hierarchical Multi-Task Approach for Learning Embeddings from Semantic Tasks*, Proceedings of the AAAI **33** (2019) 6949 (cit. on pp. 39, 40, 71).

[SB21] S. Santy and P. Bhattacharya, *A Discussion on Building Practical NLP Leaderboards: The Case of Machine Translation*, arXiv preprint arXiv:2106.06292 (2021) (cit. on p. 23).

[Sar08] S. Sarawagi, *Information extraction*, Now Publishers Inc, 2008 (cit. on p. 83).

[Sch97] B. R. Schatz, *Information retrieval in digital libraries: Bringing search to the net*, Science **275** (1997) 327, ISSN: 00368075, DOI: 10.1126/science.275.5298.327 (cit. on p. 35).

[Sch10] H. Schmid, *7 Decision Trees*, The Handbook of Computational Linguistics and Natural Language Processing (2010) 180 (cit. on p. 25).

[She+15] S. Shekarpour, E. Marx, A. C. Ngonga Ngomo, and S. Auer, *SINA: Semantic interpretation of user queries for question answering on interlinked data*, Journal of Web Semantics **30** (2015) 39, ISSN: 15708268, DOI: 10.1016/j.websem.2014.06.002 (cit. on p. 34).

[SW17] B. Shi and T. Weninger, "ProjE: Embedding Projection for Knowledge Graph Completion," *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, AAAI Press, 2017 1236 (cit. on pp. 37, 101, 104).

[Sim22] Simplilearn, *What is Web 1.0, Web 2.0, and Web 3.0? Definition, Difference & Similarities | Simplilearn*, https://www.simplilearn.com/what-is-web-1-0-web-2-0-and-web-3-0-with-their-difference-article, (Accessed on 05/09/2022), 2022 (cit. on p. 15).

[SS20] H. Singh and S. Shekhar, "Stl-cqa: Structure-based transformers with localization and encoding for chart question answering," *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020 3275 (cit. on p. 85).

[Sin+17] K. Singh et al., "Capturing Knowledge in Semantically-typed Relational Patterns to Enhance Relation Linking," *Proceedings of the Knowledge Capture Conference, K-CAP 2017, Austin, TX, USA, December 4-6, 2017*, 2017 31:1 (cit. on pp. 39, 40, 61, 71, 78).

[Sin+19] K. Singh et al., "Qaldgen: Towards microbenchmarking of question answering systems over knowledge graphs," *ISWC*, 2019 277 (cit. on pp. 75, 76).

[Sin+18] K. Singh et al., "Why Reinvent the Wheel: Let's Build Question Answering Systems Together," *WWW '18: Proceedings of the 2018 World Wide Web Conference*, Association for Computing Machinery (ACM), 2018 1247, DOI: 10.1145/3178876.3186023 (cit. on pp. 34, 45, 61, 63, 68, 70, 72–75, 78).

[Sin+16] M. Singh et al.,
OCR++: A Robust Framework For Information Extraction from Scholarly Articles,
CoRR **abs/1609.06423** (2016), arXiv: 1609.06423,
URL: http://arxiv.org/abs/1609.06423 (cit. on p. 83).

[Sin+15] 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,
New York, New York, USA: Association for Computing Machinery, Inc, 2015 243,
ISBN: 9781450334730, DOI: 10.1145/2740908.2742839 (cit. on p. 35).

[Soc+13] R. Socher, D. Chen, C. D. Manning, and A. Y. Ng,
"Reasoning with Neural Tensor Networks for Knowledge Base Completion,"
Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1, NIPS'13, Curran Associates Inc., 2013 926
(cit. on pp. 37, 101, 104, 107).

[Sol+18] Y. A. Solangi et al., "Review on natural language processing (NLP) and its toolkits for opinion mining and sentiment analysis," *2018 IEEE 5th International Conference on Engineering Technologies and Applied Sciences (ICETAS)*, IEEE, 2018 1 (cit. on p. 25).

[SP04] L. B. Sollaci and M. G. Pereira, *The introduction, methods, results, and discussion (IMRAD) structure: a fifty-year survey*, eng,
Journal of the Medical Library Association : JMLA **92** (2004), 15243643[pmid] 364,
ISSN: 1536-5050, URL: https://pubmed.ncbi.nlm.nih.gov/15243643
(cit. on p. 85).

[SL09] H. V. de Sompel and C. Lagoze,
"All aboard: toward a machine-friendly scholarly communication system,"
The Fourth Paradigm, 2009 (cit. on p. 43).

[Spa+20] G. Spadaro et al., *The Cooperation Databank: Machine-Readable Science Accelerates Research Synthesis*, 2020, DOI: 10.31234/osf.io/rveh3,
URL: psyarxiv.com/rveh3 (cit. on p. 83).

[Sto+18] M. Stocker, M. Prinz, F. Rostami, and T. Kempf, "Towards Research Infrastructures that Curate Scientific Information: A Use Case in Life Sciences,"
Data Integration in the Life Sciences - 13th International Conference, DILS 2018, Hannover, Germany, November 20-21, 2018, Proceedings, 2018 61
(cit. on pp. 59, 163, 164).

[SJY08] L. Sun, S. Ji, and J. Ye, "Hypergraph Spectral Learning for Multi-Label Classification,"
Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, Association for Computing Machinery, 2008 668, ISBN: 9781605581934 (cit. on p. 35).

[Sun+18] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang,
"RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space,"
International Conference on Learning Representations, 2018
(cit. on pp. 95, 101, 102, 104).

[Tah+21] N. Tahir et al., *FNG-IE: an improved graph-based method for keyword extraction from scholarly big-data*, PeerJ Computer Science **7** (2021) (cit. on pp. 40, 41).

[TK07] O. Tas and F. Kiyani, *A survey automatic text summarization*, PressAcademia Procedia **5** (2007) 205 (cit. on p. 84).

[Ten+19] J. P. Tennant et al., *Ten Hot Topics around Scholarly Publishing*, Publications **7** (2019), ISSN: 2304-6775, DOI: 10.3390/publications7020034 (cit. on p. 44).

[TR21] M. D. L. Tosi and J. C. dos Reis, *Scikgraph: a knowledge graph approach to structure a scientific field*, Journal of Informetrics **15** (2021) 101109 (cit. on p. 97).

[Tri+17] P. Trivedi, G. Maheshwari, M. Dubey, and J. Lehmann, "Lc-quad: A corpus for complex question answering over knowledge graphs," *ISWC*, 2017 210 (cit. on p. 78).

[Tro+16] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," *International Conference on Machine Learning*, PMLR, 2016 2071 (cit. on pp. 102, 104).

[TB21] A. Turchin and L. F. F. Builes, *Using Natural Language Processing to Measure and Improve Quality of Diabetes Care: A Systematic Review*, Journal of Diabetes Science and Technology **15** (2021), PMID: 33736486 553, DOI: 10.1177/19322968211000831 (cit. on p. 24).

[TUR50] A. M. TURING, *COMPUTING MACHINERY AND INTELLIGENCE*, Mind **LIX** (1950) 433, ISSN: 0026-4423, DOI: 10.1093/mind/LIX.236.433 (cit. on p. 23).

[Usb+18] R. Usbeck, R. H. Gusmita, A.-C. N. Ngomo, and M. Saleem, "9th Challenge on Question Answering over Linked Data (QALD-9) (invited paper)," *Semdeep/NLIWoD@ISWC*, 2018 (cit. on p. 33).

[URa15] R. Usbeck, M. Röder, and N. N. et al., "GERBIL: general entity annotator benchmarking framework," *Proceedings of the 24th WWW*, 2015 1133 (cit. on pp. 68, 78).

[Use22] D. User, *An Overview of State of the Art (SOTA) Deep Neural Networks (DNNs)*, https://deci.ai/blog/sota-dnns-overview/, (Accessed on 06/09/2022), 2022 (cit. on p. 38).

[VS17] S. Vakulenko and V. Savenkov, *TableQA: Question Answering on Tabular Data* (2017), URL: http://arxiv.org/abs/1705.06504 (cit. on p. 34).

[Vas+20] S. Vashishth, S. Sanyal, V. Nitin, and P. Talukdar, "Composition-based Multi-Relational Graph Convolutional Networks," *International Conference on Learning Representations*, 2020 (cit. on pp. 104, 107).

[Vas+17] A. Vaswani et al., "Attention is All You Need," *Proceedings of the 31st International Conference on Neural Information Processing Systems*, vol. abs/1706.03762, NIPS'17, Curran Associates Inc., 2017 6000, ISBN: 9781510860964, arXiv: 1706.03762, URL: http://arxiv.org/abs/1706.03762 (cit. on p. 41).

[Vau+13] K. Vaughan et al., *Development of the research lifecycle model for library services*, Journal of the Medical Library Association : JMLA **101** (2013) 310 (cit. on p. 44).

[Vig19] J. Vig, "A Multiscale Visualization of Attention in the Transformer Model," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Association for Computational Linguistics, 2019 37 (cit. on p. 105).

[VR18] L. Virginio and J. C. dos Reis, "Automated Coding of Medical Diagnostics from Free-Text: The Role of Parameters Optimization and Imbalanced Classes.," *DILS*, ed. by S. Auer and M.-E. Vidal, vol. 11371, Lecture Notes in Computer Science, Springer, 2018 122, ISBN: 978-3-030-06016-9 (cit. on pp. 163, 164).

[VK14] D. Vrandečić and M. Krötzsch, *Wikidata: A Free Collaborative Knowledgebase*, Commun. ACM **57** (2014) 78, ISSN: 0001-0782 (cit. on pp. 32, 39, 69, 95).

[Wan+] H. Wang et al., *A Neural Question Answering Model Based on Semi-Structured Tables*, tech. rep. 1941 (cit. on p. 34).

[WKW18] H. Wang, V. Kulkarni, and W. Y. Wang, *DOLORES: Deep Contextualized Knowledge Graph Embeddings*, 2018, arXiv: 1811.00147 [cs.CL] (cit. on pp. 37, 102, 107).

[Wan+17] Q. Wang, Z. Mao, B. Wang, and L. Guo, *Knowledge Graph Embedding: A Survey of Approaches and Applications*, IEEE Transactions on Knowledge and Data Engineering **29** (2017) 2724 (cit. on pp. 37, 95).

[Wan+14a] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge Graph and Text Jointly Embedding," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2014 1591 (cit. on p. 37).

[Wan+14b] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge Graph Embedding by Translating on Hyperplanes," *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, AAAI Press, 2014 1112 (cit. on pp. 101, 104, 107).

[WL16] Z. Wang and J. Li, "Text-Enhanced Representation Learning for Knowledge Graph," *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, AAAI Press, 2016 1293, ISBN: 9781577357704 (cit. on pp. 95, 102, 107).

[WKB18] A. Weichselbraun, P. Kuntschik, and A. M. Braşoveanu, "Mining and leveraging background knowledge for improving named entity linking," *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, 2018 1 (cit. on p. 126).

[Wei+20] G. Weikum, L. Dong, S. Razniewski, and F. Suchanek, *Machine Knowledge: Creation and Curation of Comprehensive Knowledge Bases*, arXiv preprint arXiv:2009.11564 (2020) (cit. on p. 61).

[Whi+17] K. E. White, C. Robbins, B. Khan, and C. Freyman,
Science and engineering publication output trends: 2014 shows rise of developing country output while developed countries dominate highly cited publications,
National Center for Science and Engineering Statistics InfoBrief (2017) 1 (cit. on p. 97).

[Wil+16a] M. D. Wilkinson et al.,
The FAIR Guiding Principles for scientific data management and stewardship,
Scientific Data **3** (2016) 1, DOI: 10.1038/sdata.2016.18 (cit. on pp. 97, 111).

[Wil+16b] K. Williams, J. Wu, Z. Wu, and C. L. Giles,
"Information extraction for scholarly digital libraries,"
Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries,
2016 287 (cit. on p. 83).

[Wol+20] T. Wolf et al., "Transformers: State-of-the-Art Natural Language Processing,"
Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations,
Online: Association for Computational Linguistics, 2020 38, URL:
https://www.aclweb.org/anthology/2020.emnlp-demos.6
(cit. on pp. 90, 115).

[Xia08] F. Xia, "The evolution of a statistical NLP course,"
Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics,
2008 45 (cit. on p. 24).

[Xia+17a] F. Xia, W. Wang, T. M. Bekele, and H. Liu, *Big Scholarly Data: A Survey*,
IEEE Transactions on Big Data **3** (2017) 18,
DOI: 10.1109/TBDATA.2016.2641460 (cit. on p. 85).

[Xia+19] Y. Xia et al.,
"Tied transformers: Neural machine translation with shared encoder and decoder,"
Proceedings of the AAAI conference on artificial intelligence, vol. 33, 01, 2019 5466
(cit. on p. 85).

[Xia+17b] H. Xiao, M. Huang, L. Meng, and X. Zhu, "SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions,"
Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI'17,
AAAI Press, 2017 3104 (cit. on p. 37).

[XHZ16] H. Xiao, M. Huang, and X. Zhu,
"TransG : A Generative Model for Knowledge Graph Embedding,"
Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, 2016
2316 (cit. on pp. 101, 107).

[Xie+16] R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun,
"Representation Learning of Knowledge Graphs with Entity Descriptions,"
Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16,
AAAI Press, 2016 2659 (cit. on p. 37).

[Xu+14] K. Xu, S. Zhang, Y. Feng, and D. Zhao,
"Answering Natural Language Questions via Phrasal Semantic Parsing,"
Natural Language Processing and Chinese Computing,
ed. by C. Zong, J.-Y. Nie, D. Zhao, and Y. Feng,
Berlin, Heidelberg: Springer Berlin Heidelberg, 2014 333, ISBN: 978-3-662-45924-9,
DOI: 10.1007/978-3-662-45924-9_30 (cit. on p. 36).

[Yah+12] M. Yahya et al., *Natural Language Questions for the Web of Data*, tech. rep., 2012 12 (cit. on pp. 34, 36).

[Yal21] O. G. Yalçın, *Symbolic vs. Subsymbolic AI Paradigms for AI Explainability | by Orhan G. Yalçın | Towards Data Science*, https://towardsdatascience.com/symbolic-vs-subsymbolic-ai-paradigms-for-ai-explainability-6e3982c6948a, (Accessed on 05/11/2022), 2021 (cit. on p. 23).

[Yan+20] Y. Yan et al.,
Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training,
arXiv preprint arXiv:2001.04063 (2020) (cit. on p. 90).

[Yan+14] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng,
Embedding entities and relations for learning and inference in knowledge bases,
arXiv preprint arXiv:1412.6575 (2014) (cit. on pp. 37, 101, 104, 107).

[Yan+19] X. Yang et al., "Learning Dynamic Context Augmentation for Global Entity Linking,"
EMNLP-IJCNLP, 2019 271 (cit. on pp. 80, 126).

[Yan+10] Y. Yang, F. Nie, S. Xiang, Y. Zhuang, and W. Wang, "Local and global regressive mapping for manifold learning with out-of-sample extrapolation,"
Proceedings of the AAAI Conference on Artificial Intelligence, vol. 24, 1, 2010 (cit. on p. 35).

[YML19] L. Yao, C. Mao, and Y. Luo, *KG-BERT: BERT for Knowledge Graph Completion*, 2019, arXiv: 1909.03193 [cs.CL]
(cit. on pp. 38, 39, 61, 95–98, 100, 102–104, 107).

[YV14] X. Yao and B. Van Durme,
"Information extraction over structured data: Question answering with freebase,"
Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2014 956 (cit. on p. 89).

[Yih+16] W.-t. Yih, M. Richardson, C. Meek, M.-W. Chang, and J. Suh,
"The Value of Semantic Parse Labeling for Knowledge Base Question Answering,"
Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers),
Berlin, Germany: Association for Computational Linguistics, 2016 201,
DOI: 10.18653/v1/P16-2033,
URL: https://aclanthology.org/P16-2033 (cit. on p. 33).

[Yin+15] J. Yin et al., *Neural Generative Question Answering*, IJCAI International Joint Conference on Artificial Intelligence **2016-January** (2015) 2972,
URL: http://arxiv.org/abs/1512.01337 (cit. on p. 35).

[Yu+] W. Yu, Z. Li, Q. Zeng, and M. Jiang, "Tablepedia: Automating PDF Table Reading in an Experimental Evidence Exploration and Analytic System," WWW '19 3615 (cit. on p. 61).

[Zah+20] M. Zaheer et al., *Big bird: Transformers for longer sequences*, Advances in Neural Information Processing Systems **33** (2020) 17283 (cit. on pp. 35, 41, 90).

[Zha+16] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016 353 (cit. on p. 95).

[Zha+20a] H. Zhang, H. Zhao, C. Liu, and D. Yu, "Task-to-Task Transfer Learning with Parameter-Efficient Adapter," *Natural Language Processing and Chinese Computing*, ed. by X. Zhu, M. Zhang, Y. Hong, and R. He, Springer International Publishing, 2020 391 (cit. on p. 98).

[Zha+20b] J. Zhang, Y. Zhao, M. Saleh, and P. Liu, "PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization," *Proceedings of the 37th International Conference on Machine Learning*, ed. by H. D. III and A. Singh, vol. 119, Proceedings of Machine Learning Research, PMLR, PMLR, 2020 11328 (cit. on pp. 37, 41, 90, 91).

[Zha+20c] P. Zhang et al., "TRIE: End-to-End Text Reading and Information Extraction for Document Understanding," *Proceedings of the 28th ACM International Conference on Multimedia*, New York, NY, USA: Association for Computing Machinery, 2020 1413, ISBN: 9781450379885, DOI: 10.1145/3394171.3413900 (cit. on pp. 40, 41).

[Zha+19a] Z. Zhang et al., "ERNIE: Enhanced Language Representation with Informative Entities," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, 2019 1441 (cit. on p. 38).

[Zha+19b] Z. Zhang et al., *SG-Net: Syntax-Guided Machine Reading Comprehension* (2019), URL: http://arxiv.org/abs/1908.05147 (cit. on p. 34).

[Zin+21] C. Zinke-Wehlmann et al., "Linked Data and Metadata," *Big Data in Bioeconomy: Results from the European DataBio Project*, ed. by C. Södergård et al., Cham: Springer International Publishing, 2021 79, ISBN: 978-3-030-71069-9, DOI: 10.1007/978-3-030-71069-9_7 (cit. on p. 21).

[Zin12] W. Zinsser, *On Writing Well, 30th Anniversary Edition: An Informal Guide to Writing Nonfiction*, HarperCollins, 2012, ISBN: 9780062250506 (cit. on p. 34).

[Zou+14] L. Zou et al.,
"Natural language question answering over RDF - A graph data driven approach,"
Proceedings of the ACM SIGMOD International Conference on Management of Data, Association for Computing Machinery, 2014 313, ISBN: 9781450323765,
DOI: 10.1145/2588555.2610525 (cit. on p. 34).

[Zuc+15] G. Zuccon, B. Koopman, P. Bruza, and L. Azzopardi,
"Integrating and evaluating neural word embeddings in information retrieval,"
Proceedings of the 20th Australasian document computing symposium, 2015 1
(cit. on p. 25).

[ZBW20] W. O. Zuhayeni Madjida, B. Budi, and S. P. Widodo,
Linked Open Data Implementation for Integrated Dissemination, 2020,
DOI: 10.13140/RG.2.2.27344.94724 (cit. on p. 22).

APPENDIX A

List of Publications

The following publications have been produced during the work on this thesis. All of these articles have been published in conferences or journals.

- *Conference papers*:
  1. **Mohamad Yaser Jaradeh**, Allard Oelen, Kheir Eddine Farfar, Manuel Prinz, Jennifer D'Souza, Gábor Kismihók, Markus Stocker, and Sören Auer, *Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge*, In Proceedings of the 10th International Conference on Knowledge Capture (K-CAP '19), November 19–21, Marina Del Rey, CA, USA. ACM, New York, NY, USA, 4 pages, 2019. https://doi.org/10.1145/3360901.3364435
  2. Kuldeep Singh, **Mohamad Yaser Jaradeh**, Saeedeh Shekarpour, Akash Kulkarni, Arun Sethupat Radhakrishna, Ioanna Lytra, Maria-Esther Vidal, Jens Lehmann: *Towards Optimisation of Collaborative Question Answering over Knowledge Graphs*, CoRR abs/1908.05098, 2019.
  3. Allard Oelen, **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer, *Generate FAIR Literature Surveys with Scholarly Knowledge Graphs*, In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020. JCDL '20: The ACM/IEEE Joint Conference on Digital Libraries in 2020. https://doi.org/10.1145/3383583.3398520
  4. **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer, *Question Answering on Scholarly Knowledge Graphs*, In: Hall M., Merčun T., Risse T., Duchateau F. (eds) Digital Libraries for Open Knowledge, TPDL 2020, Lecture Notes in Computer Science, vol 12246, Springer, Cham, 2020. https://doi.org/10.1007/978-3-030-54956-5_2
  5. Jennifer D'Souza, Anett Hoppe, Arthur Brack, **Mohamad Yaser Jaradeh**, Sören Auer, and Ralph Ewerth, *The STEM-ECR Dataset: Grounding Scientific Entity References in STEM Scholarly Content to Authoritative Encyclopedic and Lexicographic Sources*, In Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020), pages 2192–2203, Marseille, France, European Language Resources Association, 2020. https://aclanthology.org/2020.lrec-1.268/

6. Jan Portisch, Omaima Fallatah, Sebastian Neumaier, **Mohamad Yaser Jaradeh**, and Axel Polleres, *Challenges of Linking Organizational Information in Open Government Data to Knowledge Graphs*, In: Keet, C.M., Dumontier, M. (eds) Knowledge Engineering and Knowledge Management, EKAW 2020, Lecture Notes in Computer Science, vol 12387. Springer, Cham, 2020. https://doi.org/10.1007/978-3-030-61244-3_19

7. **Mohamad Yaser Jaradeh**, Kuldeep Singh, Markus Stocker, Andreas Both, and Sören Auer, *Better Call the Plumber: Orchestrating Dynamic Information Extraction Pipelines*, In: Brambilla M., Chbeir R., Frasincar F., Manolescu I. (eds) Web Engineering, ICWE 2021, Lecture Notes in Computer Science, vol 12706, Springer, Cham, 2021. https://doi.org/10.1007/978-3-030-74296-6_19

8. **Mohamad Yaser Jaradeh**, Kuldeep Singh, Markus Stocker, and Sören Auer, *Triple Classification for Scholarly Knowledge Graph Completion*, In Proceedings of the 11th on Knowledge Capture Conference (K-CAP '21). Association for Computing Machinery, New York, NY, USA, 225–232, 2021. https://doi.org/10.1145/3460210.3493582

9. **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer, *MORTY: Structured Summarization for Targeted Information Extraction out of Scholarly Articles*, In Proceedings of the 24th International Conference on Asia-Pacific Digital Libraries, ICADL 2022, 2022. https://doi.org/10.1007/978-3-031-21756-2_23

10. Sören Auer, Dante Barone, Cassiano Bartz, Eduardo Cortes, **Mohamad Yaser Jaradeh**, Oliver Karras, Manolis Koubarakis, Dmitry Mouromtsev, Dmitry Pliukhin, Daniil Radyush, Ivan Shilin, Markus Stocker and Eleni Tsalapati, *SciQA – A Question Answering Benchmark for Scholarly Knowledge*, Under review in Nature's Scientific Reports 2022.

- *Workshop and Demo papers*:

11. **Mohamad Yaser Jaradeh**, Allard Oelen, Manuel Prinz, Markus Stocker, Sören Auer, *Open Research Knowledge Graph: A System Walkthrough*, In: Doucet, A., Isaac, A., Golub, K., Aalberg, T., Jatowt, A. (eds) Digital Libraries for Open Knowledge, TPDL 2019, Lecture Notes in Computer Science, vol 11799, Springer, Cham, 2019. https://doi.org/10.1007/978-3-030-30760-8_31

12. Allard Oelen, **Mohamad Yaser Jaradeh**, Kheir Eddine Farfar, Markus Stocker, and Sören Auer, *Comparing Research Contributions in a Scholarly Knowledge Graph*. In Proceedings of the Third International Workshop on Capturing Scientific Knowledge co-located with the 10th International Conference on Knowledge Capture (K-CAP 2019), Marina Del Rey, CA, USA, November 19, 2019. http://ceur-ws.org/Vol-2526/

13. **Mohamad Yaser Jaradeh**, Kuldeep Singh, Markus Stocker, and Sören Auer, *Plumber: A Modular Framework to Create Information Extraction Pipelines*, In Companion Proceedings of the Web Conference 2021 (WWW '21), Association for Computing Machinery, New York, NY, USA, 678–679, 2021. https://doi.org/10.1145/3442442.3458603

- *Journal articles and Book chapters*:

14. Auer, Sören, Allard Oelen, Muhammad Haris, Markus Stocker, Jennifer D'Souza, Kheir Eddine Farfar, Lars Vogt, Manuel Prinz, Vitalis Wiens, and **Mohamad Yaser Jaradeh**, *Improving Access to Scientific Literature with Knowledge Graphs*, Bibliothek Forschung und Praxis, vol. 44, no. 32020, pp. 516-529, 2020. https://doi.org/10.18452/22049

15. Allard Oelen, **Mohamad Yaser Jaradeh**, Markus Stocker, and Sören Auer. *Chapter 10. Organizing Scholarly Knowledge leveraging Crowdsourcing, Expert Curation and Automated Techniques*, In: Linking Knowledge: Linked Open Data for Knowledge Organization and Visualization, 181–98, 2021. https://doi.org/10.5771/9783956506611-181

APPENDIX B

DILS2018 User Study

B.1 Structured Contributions Summary

Here we show two tables summarizing the research contributions collected about the user initial reactions to the ORKG early implementation in the experiment conducted at DILS2018 conference.

Table B.1: **Results of representing DILS2018 publications semantically as knowledge graph (Problem & Approach).** The table states the exact information collected from the participants.

Publication	Problem	Approach
Henry et al. [Hen+18]	formalism	deep data integration
	Alzheimer's disease	ADO connection
Christen et al. [Chr+18]	entity linking	—
Stocker et al. [Sto+18]	machine-readability	leverage web technologies
	scholarly communication not evolving with technology	
	reproducibility crisis	capturing information before publishing
	information embedded in graphs	
Virginio et al. [VR18]	imbalanced classes	SVM
		class weighting
Friedrich et al. [Fri+18]	reproducibility crisis	Factorial experimental designs
	sample size	
Malone et al. [MGN18]	multi-relational link prediction	negative sampling
		mixture of experts
	polypharmacy side effect prediction	embedding
		machine learning

Table B.2: **Implementation & Evaluation results of representing DILS2018 publications semantically.** Left value is property, right value is the object value. The table states the exact information collected from the participants.

Publication	Implementation		Evaluation	
Henry et al. [Hen+18]	environment:	protégé	—	
		protégé Cellifie plugin		
	based on:	RO		
		mEPN		
		SBO		
		OWL		
	is a:	ontology		
Christen et al. [Chr+18]	Uses:	Annotation tools	performs better:	set-based tool combination
		Ontology		
		unstructured medical documents	uses:	F1-Score
		machine learning		
Stocker et al. [Sto+18]	uses ontology:	BFO	—	
		GO		
		STATO		
		IAO		
	described by:	RDF		
	programming language:	Python		
	environment:	JupyterLab		
	interacts with:	REST API		
Virginio et al. [VR18]	programming language:	Python	metric:	fmeasure:
			technique:	parameter searching
Friedrich et al. [Fri+18]	uses framework:	isatools	—	
		javafx		
		vaadin		
	uses:	openBIS		
		data management system		
	programming language:	Java		
		Javascript		
	uses library:	isatools		
		dagre.js		
		D3.js		
	method:	graph aggregation		
Malone et al. [MGN18]	—		datasets:	only drugs with known gene targets
				drug-drug only
				drug and drug-gene
			qualitative:	interpretable example
			metrics:	AP@50
				AuPR
				AUC

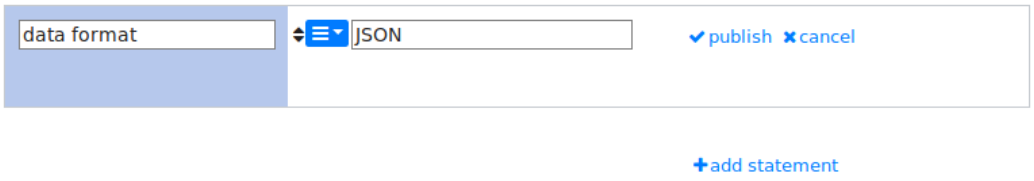
B.2 User Study Questionnaire

Here is a list of questions that were asked to everyone of the 12 participants of the user evaluation study.

1. What is your name? (Optional)
2. Title of publication at DILS? (Required)
   - Input as free text
3. How intuitive is the user interface navigation? (Required)
   - Linear scale ranging from one (not intuitive) to five (very intuitive)
4. Was the terminology (resources, literals, predicates, statements) well undetstood? (Required)
   - Linear scale ranging from one (hard to understand) to five (easy to understand)

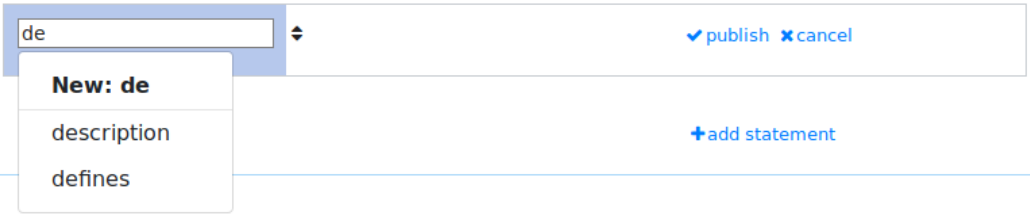
New predicate

New resource

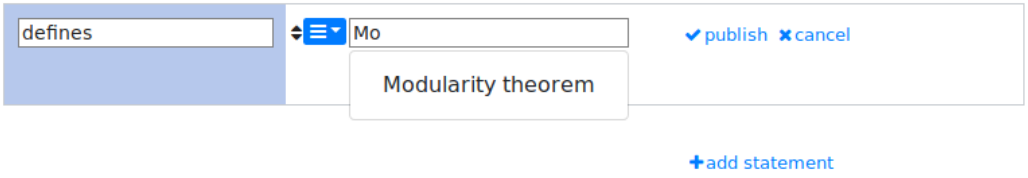


5. Do you prefer to add predicates yourself or have them predefined? (Required)
   - ○ Add myself
   - ○ Choose from predefined list
   - ○ Other (with free text input)
6. How useful is the autocomplete feature (suggestion) to add a predicate or a resource? (Required)
   - Linear scale ranging from one (not helpful at all) to five (very helpful)

Predicate selection

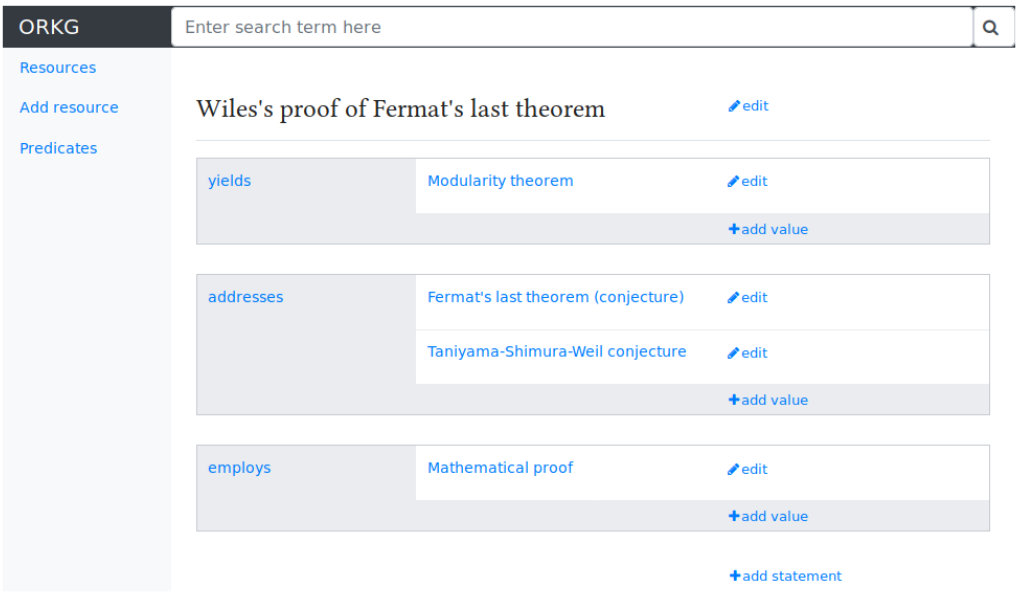


Resource selection



7. How much supervision or guidance did you need to use the application? (Required)
   - Linear scale ranging from one (not at all) to five (all the time)
8. How likely is that you will suggest others to try out the application? (Required)
   - Linear scale ranging from one (not likely) to nine (very likely)

UI design



9. How much do you like the design of the user interface? (Required)
   - Linear scale ranging from one (not at all) to nine (very much)

10. How can we improve user experience? (Optional)
    - Long answer free text)
11. Any other comments? (Optional)
    - Long answer free text)

List of Figures

List of Tables

[1] Fetched from https://www.orkg.org/orkg/c/Zg4b1N