

GOTTFRIED WILHELM LEIBNIZ UNIVERSITÄT HANNOVER  
FAKULTÄT FÜR ELEKTROTECHNIK UND INFORMATIK

# Information Retrieval Service Aspects of the Open Research Knowledge Graph

*A thesis submitted in fulfillment of the requirements for the degree of  
Master of Science in Computer Science*

BY

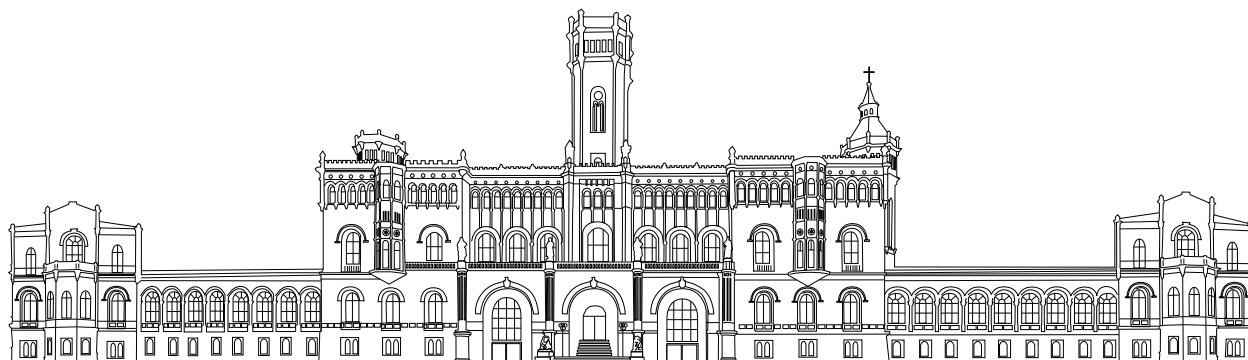
**Omar Arab Oghli (B.Sc.)**

Matriculation number: 10006018

E-mail: [omar.arab.oghli@stud.uni-hannover.de](mailto:omar.arab.oghli@stud.uni-hannover.de)

First examiner: Prof. Dr. Sören AUER  
Second examiner: Dr. Jennifer D'SOUZA  
Supervisor: Dr. Jennifer D'SOUZA

January, 2022





# Declaration of Authorship

I, Omar ARAB OGHLI, declare that this thesis titled, **Information Retrieval Service Aspects of the Open Research Knowledge Graph**, and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.

Omar ARAB OGHLI

Signature: \_\_\_\_\_ 

Date: 14.01.2022



## *Acknowledgements*

I would first like to thank my thesis supervisor and second examiner Dr. Jennifer D'SOUZA for her experienced advice and guidance in the right direction whenever her support was needed. Many thanks to Prof. Dr. Sören AUER my first examiner for giving the opportunity being a member in his research group and for providing everything needed. I also would like to thank Mohamad Yaser JARADEH for his previous work on the Contributions Similarity Service and continuous technical support. Thanks to Manuel *Prinz*, Suhas MURTHY and Kheir Eddine FARFAR for their technical support and to all ORKG team members and contributors.

Finally, I must express my gratefulness to my father, mother, brother, sisters, beloved friends, fellow students and colleagues for their continuous encouragement throughout my years of study and working on this thesis. Thank you!

Omar Arab Oghli.



## *Abstract*

Information Retrieval (**IR**) takes a fresh perspective in the context of the next-generation digital libraries such as the Open Research Knowledge Graph (**ORKG**). As scholarly digital libraries evolve from document-based to knowledge-graph-based representations of content, there is a need for their information technology services to suitably adapt as well. The **ORKG** enables a structured representation of scholarly contributions data as **RDF** triples - in turn, it fosters **FAIR** (Findable, Accessible, Interoperable, and Reusable) scholarly contributions. This thesis has practically examined three different **IR** service aspects in the **ORKG** with the aim to help users: (i) easily find and compare relevant scholarly contributions; and (ii) structure new contributions in a manner consistent to the existing **ORKG** knowledge base of structured contributions. In the first part, it will evaluate and enhance the performance of the default **ORKG Contributions Similarity Service**. An optimal representation of contributions as documents obtains better retrieval performance of the BM25 algorithm in Elasticsearch. To achieve this, evaluation datasets were created and the contributions search index reinitialized with the new documents. In its second part, this thesis will introduce a *Templates Recommendation Service*. Two approaches were tested. A supervised approach with a Natural Language Inference (**NLI**) objective that tries to infer a contribution template for a given paper if one exists or none. And an unsupervised approach based on search that tries to return the most relevant template for a queried paper. Our experiments favoring ease of practical installation resulted in the conclusion that the unsupervised approach was better suited to the task. In a third and final part, a *Grouped Predicates Recommendation Service* will be introduced. Inspired from prior work [6], the service implements K-Means clustering with an **IR** spin. Similar structured papers are grouped, their in-cluster predicate groups computed, and new papers are semantified based on the predicate groups of the most similar cluster. The resulting micro-averaged F-measure of 65.5% using TF-IDF vectors has shown a sufficient homogeneity in the clusters.

*Keywords: Digital Libraries, Knowledge Graph, ORKG, Structured Data, Information Retrieval*





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Related Work</b>	<b>4</b>
2.1	Scholarly Communication as Knowledge Graphs . . . . .	4
2.2	Information Retrieval over Knowledge Graphs . . . . .	4
2.3	Information Retrieval Evaluation Metrics . . . . .	6
<b>3</b>	<b>The Open Research Knowledge Graph</b>	<b>7</b>
<b>4</b>	<b>Scholarly Contributions Similarity</b>	<b>16</b>
4.1	Technical Description . . . . .	17
4.2	ORKG Similarity Service Evaluation Corpus . . . . .	20
4.3	Analysis of the Default ORKG Contributions Similarity Service . . . . .	23
4.3.1	The DocumentCreator Module . . . . .	24
4.3.2	The Elasticsearch Module . . . . .	25
4.3.3	The Default ORKG Contributions Similarity Service Response Analysis . . . . .	26
4.4	Thesis Implementation of Simcomp Enhanced . . . . .	27
4.5	Evaluations and Discussions . . . . .	30
4.5.1	The “with response” and “without response” Results . . . . .	31
4.5.2	Heterogeneous Set Results with Mean Reciprocal Rank . . . . .	32
4.5.3	Homogeneous Set Results with Mean Average Precision . . . . .	34
4.5.4	Qualitative Analysis . . . . .	37
4.6	Conclusion . . . . .	40
<b>5</b>	<b>Templates Recommendation</b>	<b>43</b>
5.1	Service Feasibility Analysis . . . . .	46
5.1.1	Task Dataset . . . . .	47
5.1.2	Dataset Analysis . . . . .	49

5.2	Service Implementation . . . . .	52
5.2.1	Unsupervised Approach with Elasticsearch . . . . .	53
5.2.2	Supervised Approach with SciBERT . . . . .	53
5.3	Service Evaluation . . . . .	56
5.3.1	Experimental Setup . . . . .	56
5.3.2	Results and Discussion . . . . .	59
5.4	Conclusion . . . . .	60
<b>6</b>	<b>ORKG Grouped Predicates Recommendation</b>	<b>63</b>
6.1	Introduction . . . . .	63
6.2	Service Feasibility Analysis . . . . .	65
6.2.1	Dataset of Single Predicates . . . . .	66
6.2.2	Dataset of Clustered Predicates . . . . .	67
6.3	Service Implementation . . . . .	69
6.3.1	K-means Clustering of Contribution Vectors . . . . .	72
6.4	Service Evaluation . . . . .	73
6.4.1	Experimental Setup . . . . .	73
6.4.2	Results and Discussion . . . . .	75
6.5	Conclusion . . . . .	76
	<b>Bibliography</b>	<b>79</b>

# Acronyms

**(S, P, O)** (Subject, Predicate or Property, Object). 19, 28, 63, 77

**BM25** Best Matching 25. 17, 19, 24, 26, 38, 40, 42, 53, 59

**DL** Digital Libraries. 1, 2, 7

**DOI** Digital Object Identifier. 48, 53, 54

**FAIR** Findable Accessible Interoperable Reusable [75]. III, VIII, 16

**KG** Knowledge Graph. 4, 5, 7, 63

**MAP** Mean Average Precision. 6, 32, 34–36, 38, 42, 57

**MRR** Mean Reciprocal Rank. 6, 32–35, 38, 42, 57

**NDCG** Normalized Discounted Cumulative Gain. 6

**NLI** Natural Language Inference. III, 52, 54, 61, 62

**NLP** Natural Language Processing. 4, 25, 27, 29

**NLTK** Natural Language Toolkit. 29, 30

**ORKG** Open Research Knowledge Graph. III–V, 2–5, 7–17, 19–28, 30, 37, 40–48, 50–53, 58, 60–66, 68–72, 74–78

**PWC** Papers With Code. 4

**RDF** Resource Description Framework [17]. III, VIII, 21, 47, 65

**RR** Reciprocal Rank. 32

**SKG** Scholarly Knowledge Graph. 4, 63

**SPAR** Semantic Publishing and Referencing. 4

**TF-IDF** Term Frequency - Inverse Document Frequency. 17

**TIB** Technische Informationsbibliothek (The German National Library of Science and Technology). VIII, 19

# Glossary

**Back-end** The software layer behind the user interface where the program operates the user input and interact with a database or other back-end services. 10, 62

**Cypher** A graph query language.. 24

**Front-end** User Interface or Client calling services behind a server. 8–10, 18, 19, 26, 44, 66

**Hyperparameters** Parameters that control the learning process or the output of a function for some purposes. 18

**IR** Information Retrieval. The process of retrieving information documents from a system or documents collection, that are relevant to the search query, i.e. satisfies the information need. III, 2, 4, 17, 40, 52, 60, 63, 65, 76, 78

**OLS** Ontology Lookup Service for TIB. The TIB service URL <https://service.tib.eu/ts4tib/index> and the OLS URL <https://www.ebi.ac.uk/ols/index..> 47

**Ontology** The science of being. It describes how entities are grouped and interact with each other. 16

**Python** Object oriented, interpreted dynamic typed programming language. 11–13, 19

**Simcomp** A project introduced by Oelen et al. [51] for selecting similar research contributions and comparing them under the FAIR principles. IV, 16, 17, 19, 20, 26, 27, 29–32, 37, 41, 42, 69

**SPARQL** RDF query language. 21

**YAML** Human-readable data-serialization language. 28, 30, 34, 36, 42



# Chapter 1

## Introduction

Digital libraries (DL) play an important role in scholarly knowledge dissemination and exploration, which is a critical part of scholarly communication lifecycle. The availability of massive scholarly records in DL platforms have already brought practical accessibility benefits to a variety of research communities for scholarship development. However, given the present massive publication trends [36], researchers, digital librarians, and scientific data curators are faced with new challenges. In particular, information organization via keyword-based search engines over document-based scholarly records are proving to be no longer an efficient model for conducting research or managing the research record. In this scenario, the task of systematic literature reviews [41], which involve summarizing vast amounts of investigations on a specific topic, even in one's own narrow discipline, is becoming practically impossible. The core point of the problem is, that researchers have to glance over large volumes of scholarly articles in order to acquire answers w.r.t a specific scientific inquiry [40]. A problem that is only more acute in a multidisciplinary context [24]. Scholarly knowledge in its different forms like new ideas and approaches, standards and best practices, the description of new phenomena and data, are still mostly *unstructured* w.r.t. the machine interpretability terms, even that they have been existing for long time abundantly as digital records on the web. Thus to the computer, scholarly content is semantically just an index of keywords, which clearly leaves buried layers of their rich content. Considering the current semantic technologies, for instance, the evolution of semantic modeling languages in terms of expressivity and an expanding network of ontologies, DLs have an increased possibility to be more semantically *structured*. Thus, scholarly applications could access the inquired data more accurately. As a consequence, researchers can easily find what they are looking for using such semantically *structured* data. To optimize scholarly knowledge orga-



nization and representation in DL, some initiatives [34, 74] advocate for building an interlinked and semantically rich knowledge graph structure using a combination of human curation and machine learning.

This thesis explores technology in the context of such a knowledge-graph-based next-generation digital library (DL). Specifically, the DL is called the Open Research Knowledge Graph (ORKG) wherein the aim is to represent structured *scholarly contributions*. In the ORKG, a scholarly contribution is interpreted as *the result of the investigation that contributes towards the advancement of scientific human knowledge by adding something new*. The ORKG is an infrastructure for the acquisition, curation, publication and processing of semantic scholarly knowledge. It leverages structured scholarly knowledge acquisition using crowdsourcing and text mining techniques as well as supports knowledge curation, publication and processing.

Thus in the broader context of the ORKG as a DL service, this thesis focuses on its Information Retrieval service aspects, while leveraging the structured data characteristic of the ORKG. Whereas unstructured data retrieval is concerned with retrieval of raw text out of a documents collection, structured data retrieval aims in contrast to retrieve text fragments which put more semantic emphasis on the information inquiry. Even in a structured data context the task of keyword-based searching to find similar contributions is an exceedingly difficult task for the researcher. Adding a new perfectly structured contribution to the ORKG can also be considered as a non-trivial task due to the large number of existing entities with various semantic meanings in the ORKG and the multiplicity of researchers scientific backgrounds.

Particularly, this thesis aims to evaluate and enhance an existing information retrieval service in the ORKG called the *Contributions Similarity Service* and also to enrich the ORKG with two new recommendation services to help researchers in the process of structuring their contributions in the graph. For the latter purpose, a recommendation service for each of *Templates*<sup>1</sup> and *Predicates* will be introduced. Having such IR services in the ORKG enhances the user experience in two respects: 1) similarly structured contributions will be found more easily and better relevance; and 2) given unstructured text, existing template patterns or predicate clusters that were used in semantifying similar texts will be available as suggestions thus expediting the structuring of a paper. Based on the mentioned aims the research questions of this thesis are formally defined as follows:

- i. What are the features of the default *Contributions Similarity Service* implementation, what are its quantitative evaluations, and how can it be enhanced/optimized?

---

<sup>1</sup>An ORKG template is a subgraph defining a reusable domain-specific set of predicates and their data types as building blocks of a structured contribution.

- 
- ii. How can an applicable template from the **ORKG** knowledge base be recommended to structure a new incoming paper?
  - iii. How can an applicable cluster of **ORKG** predicates be recommended as a unit based on a query represented by a research paper to structure the paper?

The remainder of this work is structured as follows. Chapter 2 reviews the related work. Chapter 3 presents the features of the **ORKG** as a software platform. In chapter 4 we evaluate and suggest enhancements to the default *Contributions Similarity Service*. Chapter 5 describes one supervised and one unsupervised approach to build a *Templates Recommendation Service*. Finally, chapter 6 introduces a clustering method to group research papers, to which predicates can be recommended. Each of the main thesis parts are concluded within their respective chapters.

# Chapter 2

## Related Work

### 2.1 Scholarly Communication as Knowledge Graphs

Scholarly metadata **KGs** are common. Research Objects [12] and Nanopublications [29] use machine-readable abstract structures to link the products of a research investigation, including articles, data, and other research artifacts. Recently, **SKGs** are being based on article content. E.g., initiatives as the Semantic Publishing and Referencing (**SPAR**) Ontologies [55] and the Journal Article Tag Suite [23] which consider document structure and fine-grained discourse elements. There are other comprehensive conceptual models for scholarly knowledge that capture problems, methods, theories, statements, concepts, and their relations [31, 21, 15, 50]. Allen [3, 4] investigated issues for implementing entire research reports as structured knowledge bases. Fathalla et al. [27] semantically represent key aspects of surveys as research problems, approaches, implementations, and evaluations. **NLP**-based Semantic Scholar [5] and empirical research focused <https://paperswithcode.com/> (**PWC**) are related systems. The key distinction of the **ORKG** from these works is knowledge capture focus (full content vs. only contributions in **ORKG**).

### 2.2 Information Retrieval over Knowledge Graphs

Knowledge Graphs, with their ubiquitousness and structured representation of information, are an instrument in Information Retrieval that enable semantic search. Reinanda et al. [62] summarize **IR** over **KGs** in different tasks and approaches. In particular, *Document Retrieval* and *Entity Retrieval* tasks. We subsequently provide a brief description of each task and the respective common approaches.

**Query Document Retrieval.** The aim of this task is to rank documents w.r.t. each other from a documents collection given a query based on its relevance to the query. *Expansion-based* approaches [20, 76] leverage different query expansion strategies to enrich the query with information from the **KG** entities and their links. *Latent factor* approaches [76, 45] map queries and documents to a latent space based on the entities and their relationships in the **KG** and then rank the documents with a similarity measure like KL-Divergence [38]. *Language modeling* approaches [61, 25] introduce document retrieval by leveraging language models based on entity or term frequencies in the **KG**. Ensan and Bagheri [25] computed the query likelihood model based on an undirected graphical model built around the entities. *Deep learning* approaches [77, 46] utilize **KG** embeddings, where query and documents are represented as embeddings of entities and the semantic relatedness is computed in the embedding space.

**Query Entity Retrieval.** This task aims at ranking entities of a **KG** based on their relevance to a given query. As in document retrieval tasks, some works [10, 11, 56] leverage *language modeling* approaches to retrieve entities from **KGs** with different modeling strategies. Van Gysel et al. introduce in their works [72, 71, 30] neural approaches for entity retrieval, where they learn distributed word representation from textual evidence in an unsupervised way and improve their work by learning term and entity representations in a different latent space. *Multi-fielded representations* approaches [39, 58, 70] define a **KG** entity as an object with fields and their relationships, where it is represented as a set of fields with bag-of-words values. Then they apply multi-fielded retrieval algorithms such as TF-IDF [65] and BM25F [54] to build a language model.

This work utilizes the knowledge gained from different related works. In particular, we leverage the multi-fielded entities representation in chapters 4 and 5 with the BM25 [64] multi-fielded retrieval algorithm to build a language model for both queries and documents. In chapter 5 we also introduce a supervised neural approach, where the multi-fielded entities and text queries are represented in SciBERT [13] latent space. Whereas in chapter 6 we utilize an unsupervised algorithm to group similar texts represented in SciBERT latent space that are then mapped to structured entities in the **ORKG**.

## 2.3 Information Retrieval Evaluation Metrics

As stated in the standard textbook on Information Retrieval [49] and in a survey on evaluation in Information Retrieval [66], there is a set of common evaluation metrics in the research field that can be classified depending on the retrieval results set. For ranked retrieval results, the most common metrics are Mean Average Precision (**MAP**), Mean Reciprocal Rank (**MRR**) and Normalized Discounted Cumulative Gain (**NDCG**), which can handle an ordered set of results. On the other hand, Precision, Recall and their harmonic mean F-measure are considered as the common metrics for dealing with unranked retrieval results. In **chapter 3**, **MAP** and **MRR** are applied in the evaluation part to measure retrieval ranks. And in **chapters 5** and **6**, the precision, recall, and f-measure, are applied where only retrieval results are considered.

## Chapter 3

# The Open Research Knowledge Graph

Since the Open Research Knowledge Graph (ORKG) digital library forms the basis of this thesis, it is explained in detail in this chapter for its existing features.

The ORKG leverages the knowledge graph (KG) model for representing *scholarly contributions*. In the ORKG, a scholarly contribution is interpreted as *the result of the investigation that contributes towards the advancement of scientific human knowledge by adding something new*. The ORKG is an infrastructure for the acquisition, curation, publication and processing of semantic scholarly knowledge. It leverages structured scholarly knowledge acquisition using crowdsourcing and text mining techniques as well as supports knowledge curation, publication and processing. The platform is hosted online at <https://www.orkg.org/>. The software underlying the platform is open source at <https://gitlab.com/TIBHannover/orkg> and users can provide feedback on issues and features, guide future development with requirements, and, last but not the least, contribute to the implementation itself. Eight main characteristic features of the knowledge-graph-based ORKG DL that differentiate it from the traditional document-based DLs are the following.

**1. Structured research contributions.** The ORKG aims to capture structured and semantified scholarly articles for their research contributions.<sup>1</sup> The structured information includes: basic metadata about the paper (e.g., title and authors); structured content about the research contribution describing the addressed problem, the utilized materials and methods, and the obtained result. In the graph, the metadata and contributions underlay different encapsulating nodes. And a paper can be con-

---

<sup>1</sup><https://www.orkg.org/orkg/add-paper>

nected to more than one contribution node. An example of a structured article in the **ORKG** for its contributions is depicted in **Figure 3.1**.

The screenshot shows the front-end view of a scholarly article in the ORKG platform. The article title is "Transmission interval estimates suggest pre-symptomatic spread of COVID-19". It lists several authors: March 2020, Virology, Lauren Tindale, Michelle Coombe, Jessica Stockdale, Emma Garlock, Wing Yin Venus Lau, Manu Saraswat, Yen-Hsiang Brian Lee, Louxin Zhang, Dongxuan Chen, Jacco Wallinga, and Caroline Colijn. The DOI is 10.1101/2020.03.03.20029983.

The article is structured into two contributions: "Contribution 1" (selected) and "Contribution 2". Contribution 1 details the research problems, contribution data, and provenance information.

**Research problems**

- COVID-19 reproductive number

**Contribution data**

95% Confidence interval	1.45-2.48
Location	Singapore
Mean incubation period	7.1 (6.13, 8.25) days
Mean serial interval	4.56 (2.69, 6.42) days
R0 estimates (average)	1.97
Study date	2020-01-19/2020-02-26
Study location	Singapore

**Provenance**

- Belongs to observatory: COVID-19
- Added on: 08 Apr 2020
- Added by: Jennifer D'Souza
- Contributors: Kheir Eddine, Mariya Yanchevska, Sören Auer, Import robot covid

Figure 3.1: The **Front-end** flattened graph view of a scholarly article having its two contributions (see red Tabs titled ‘Contribution 1’ and ‘Contribution 2’) structured in the Open Research Knowledge Graph platform.

**2. Templates.** **ORKG** supports the possibility of creating templates that standardize the structure of content types of a contribution; and of reusing the templates when describing new research contributions. **Figure 3.2** shows the specification of the attributes of a process that estimates the basic reproduction number of a population using a specific method.

**3. Comparisons.** The **ORKG** also supports downstream applications such as the creation of surveys over structured scholarly contributions. In other words, given

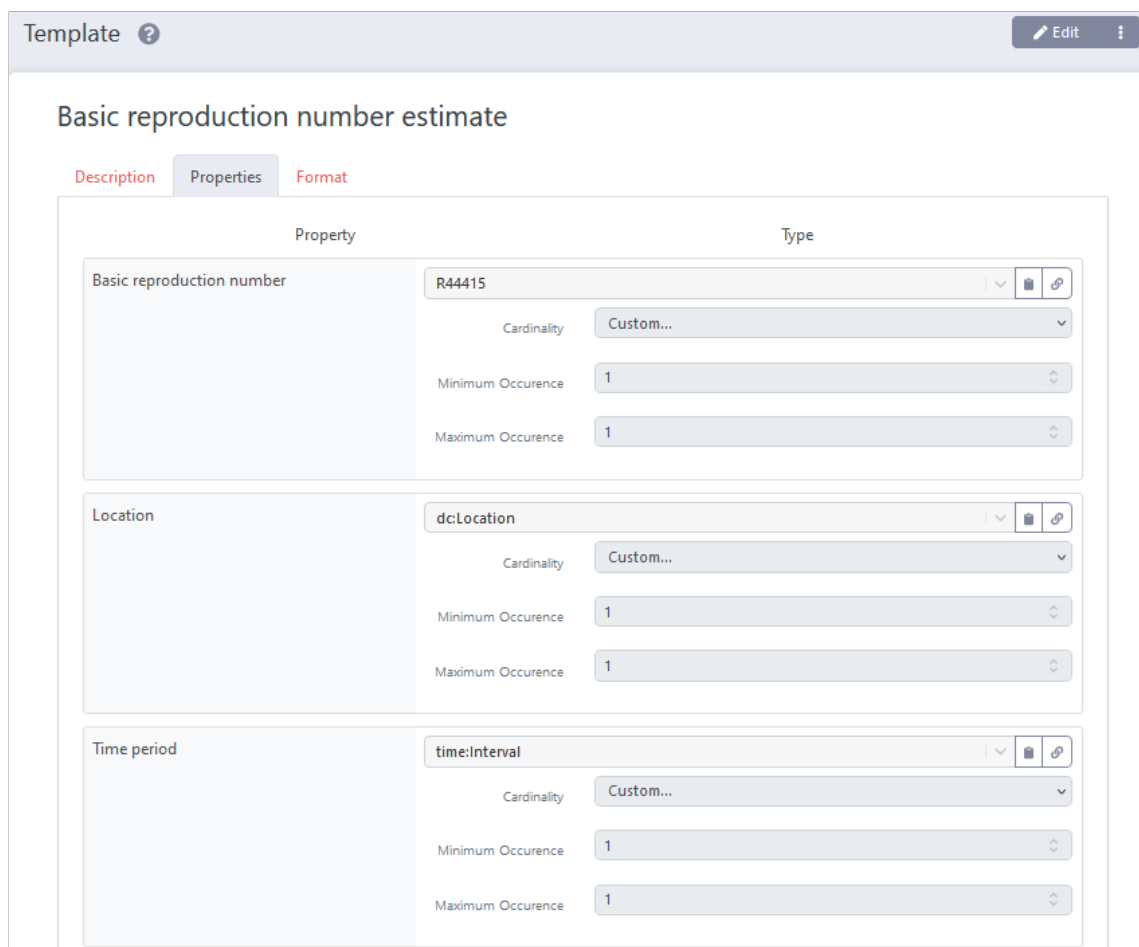


Figure 3.2: A template in the Open Research Knowledge Graph to model the standardized properties of the ‘Basic reproduction number estimate’ contribution information from scholarly articles.

articles structured for their research contributions, it is possible to compare several contributions across papers that address a specific research problem. [Figure 3.3](#) shows an example of an automatically computed survey.

**4. Graph visualization.** The example screenshots so far show how research contribution data is presented in the statement browser in the [ORKG Front-end](#). However, since the [ORKG](#) is a knowledge graph, paper and research contribution descriptions can also be visualized as a graph. Indeed this is the data structure for



Properties	Transmission interval estimates suggest pre-symptomatic spread of COVID-19 Contribution 1 - 2020	Transmission interval estimates suggest pre-symptomatic spread of COVID-19 Contribution 2 - 2020	Estimation of the epidemic properties of the 2019 novel coronavirus: A mathematical modeling study Contribution 1 - 2020
R0 estimates (average)*	1.97	1.87	4.38
95% confidence interval	1.45-2.48	1.65-2.09	3.63-5.13
Study date	2020-01-19/2020-02-26	2020-01-21/2020-02-27	2020-01-10/2020-01-23
Location	Singapore	Tianjin, China	Wuhan City, China

Figure 3.3: An automatically compiled survey of structured contributions for the Covid-19 reproductive number ([https://www.orkg.org/orkg/problem/R12219/COVID-19\\_reproductive\\_number](https://www.orkg.org/orkg/problem/R12219/COVID-19_reproductive_number)) research problem in the Open Research Knowledge Graph platform.

storing the structured contribution descriptions in the **Back-end**. In the **Front-end**, additionally, the **ORKG** makes available an alternative graph view providing a complementary way to interact with **ORKG** content. It is a dynamic user interface for visual exploration of graph data and includes a range of features to make exploration of highly structured graph data intuitive. The graph is automatically optimally arranged on the screen. Nodes can easily be expanded, collapsed or removed. The nodes can be represented in a graph or tree layout. Users can search for information in the graph. An example of a graph view of a structured paper contributions can be seen in **Figure 3.4**.

**5. Observatories.** The **ORKG** implements the model of Observatories to organize and highlight structured scholarly knowledge contributors at the institutional level. With the help of this feature, the pooling of disciplinary expertise is enabled. Observatories comprise essentially groups of experts (e.g., senior researchers) affiliated with different institutions that curate and organize **ORKG** content for a specific discipline, and within that typically a specific research area or even a research problem. Since such knowledge curation and organization is time consuming, **ORKG** acknowledges the contribution of experts as well as the institutions they are members of. **Figure 3.5** shows how this acknowledgment is implemented as provenance information for the

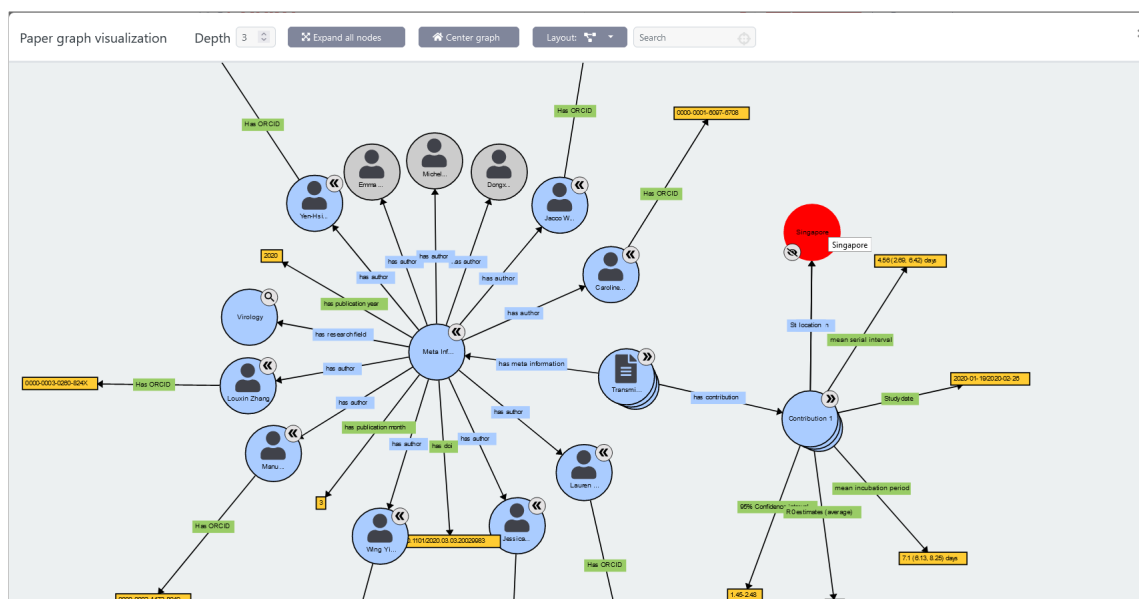


Figure 3.4: The graph view of one research paper in the Open Research Knowledge Graph having its metadata (e.g. title, authors, date) and structured data of one contribution. Expanding, collapsing or removing a node can be triggered by clicking on the respective icon at the respective node.

description of research contributions. Observatories and their experts can contribute in numerous ways to **ORKG**. In addition to adding and describing papers or curating existing papers, observatories play a crucial role in knowledge organization for a particular research area. In **ORKG**, observatories can for instance specify templates for the information types that are most relevant to their area of research. In doing so, observatories help ensure the creation of high quality and comparable structured scholarly knowledge for their area.

**6. Data Science.** The structured and semantic description of scholarly knowledge enables easier reuse of fine-grained structured scholarly knowledge in downstream applications. The comparisons feature discussed earlier is only one example of a downstream advanced information technology application on the **ORKG**. The structured data, however, can be leveraged in various other data analytics settings. To support this diversity, the **ORKG** implements a web-based interface (REST API), which can be used in, for instance, **Python** programming to access and process **ORKG** content. For **Python**, we also provide a specialized **ORKG** library, which further simplifies

Energy Disaggregation Based Deep Learning Techniques: A Pre-Processing Stage To Enhance The Household Load Forecasting

January 2018 Social and Behavioral Sciences Ebrahim AF Mohammed OA

Published in:

Share this paper:

**Contribution 1**

**Research problems**  Add to comparison  
Indoor environmental perception and behaviour

**Contribution data**

optimization based on the defined objective function	✘
Data-driven models: agglomerative hierarchical clustering	✘
Data-driven models: apriori	✘
Data-driven models: arma, arima, etc..	✘

**Provenance** **Timeline**

OCCUPANTS' PERCEPTION AND BEHAVIOUR  
**UNIKLINIK RWTH AACHEN**

DATE ADDED  
03 Jun 2020

ADDED BY  
Marcel Schweiker

CONTRIBUTORS  
Marcel Schweiker

Figure 3.5: A research contribution in the ORKG with a provenance of the observatory “Occupants’ Perception and Behaviour” [https://www.orkg.org/orkg/observatory/Occupants\\_perception\\_and\\_behaviour](https://www.orkg.org/orkg/observatory/Occupants_perception_and_behaviour) and the organization “Uniklinik RWTH Aachen” [https://www.orkg.org/orkg/organizations/Uniklinik\\_RWTH\\_Aachen](https://www.orkg.org/orkg/organizations/Uniklinik_RWTH_Aachen) shown on the right side of the contribution view.

accessing ORKG content (<https://pypi.org/project/orkg/0.11.3/>).<sup>1</sup> With it one can load ORKG structured information (individual contribution descriptions and comparisons) into a data analysis environment such as Jupyter Notebooks and language-native data structures such as pandas DataFrame to further process data and create domain-specific applications for data visualization, interpolation, modeling, simulation, etc. An example is depicted in Figure 3.6. More information on performing data science with the ORKG data is available online [https://www.orkg.org/orkg/help-center/article/15/ORKG\\_Data\\_Science\\_with\\_Jupyter](https://www.orkg.org/orkg/help-center/article/15/ORKG_Data_Science_with_Jupyter).

<sup>1</sup>The documentation of the ORKG Python package is accessible online at <https://orkg.readthedocs.io/en/latest/index.html>.

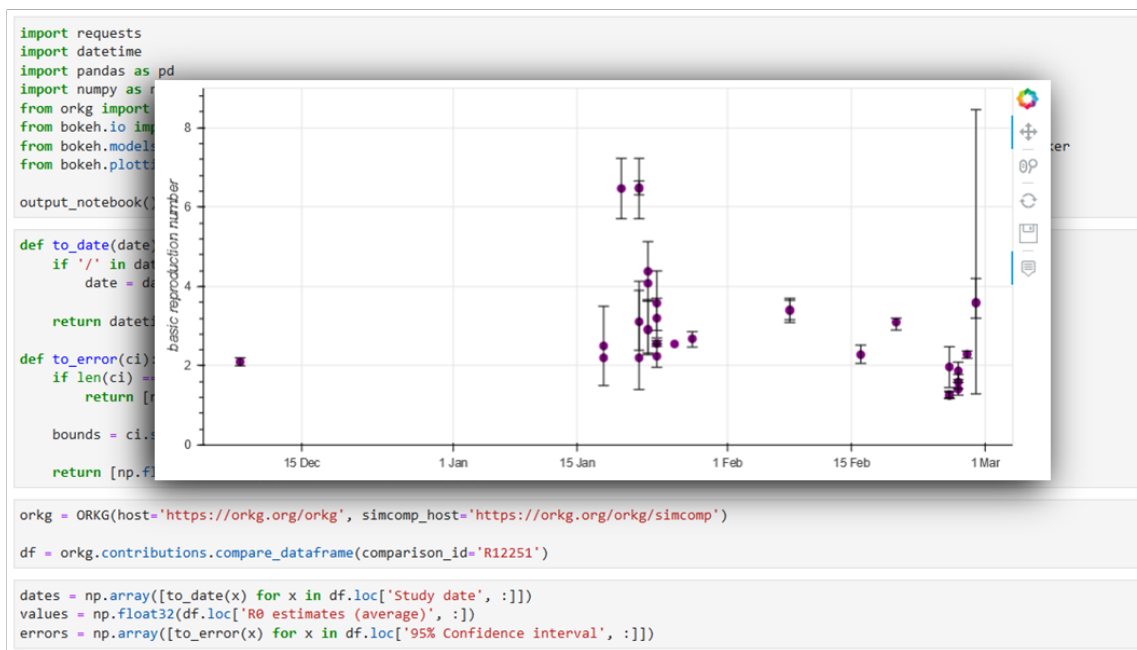


Figure 3.6: An instance of implementing visualization data analytics on an imported ORKG comparison data in a Jupyter notebook using the ORKG Python API.

**7. Benchmarks.** This feature enables tracking the progress of scientific research tasks defined as datasets. E.g. Question Answering defined over the SQUAD dataset [59] (<https://www.orkg.org/orkg/benchmark/R119758/problem/R2061>). Benchmarks are a special case of an ORKG Contribution. A Benchmark for an eligible paper can be defined by leveraging the Leaderboard template (<https://www.orkg.org/orkg/template/R107801>) to structure contributions from the paper, and the relevant dataset, model, and evaluation measures will automatically be plotted in a new or existing performance evolution trends chart. The charts are grouped by *Research problem*, *Dataset*, and *Metric*. Figure 3.7 illustrates a benchmark page.

**8. SmartReviews.** This is a novel conceptualization of a research article in the context of leveraging the ORKG structure contributions and surveys. This new feature called SmartReviews enables multiple users to create review or a survey article basically grouping multiple surveys etc. in the context of one scientific findings narrative. SmartReviews are dynamic, community maintained articles that anyone can contribute to (similar to Wikipedia). SmartReview articles consist of several ORKG components, including: comparisons, visualizations, and individual paper

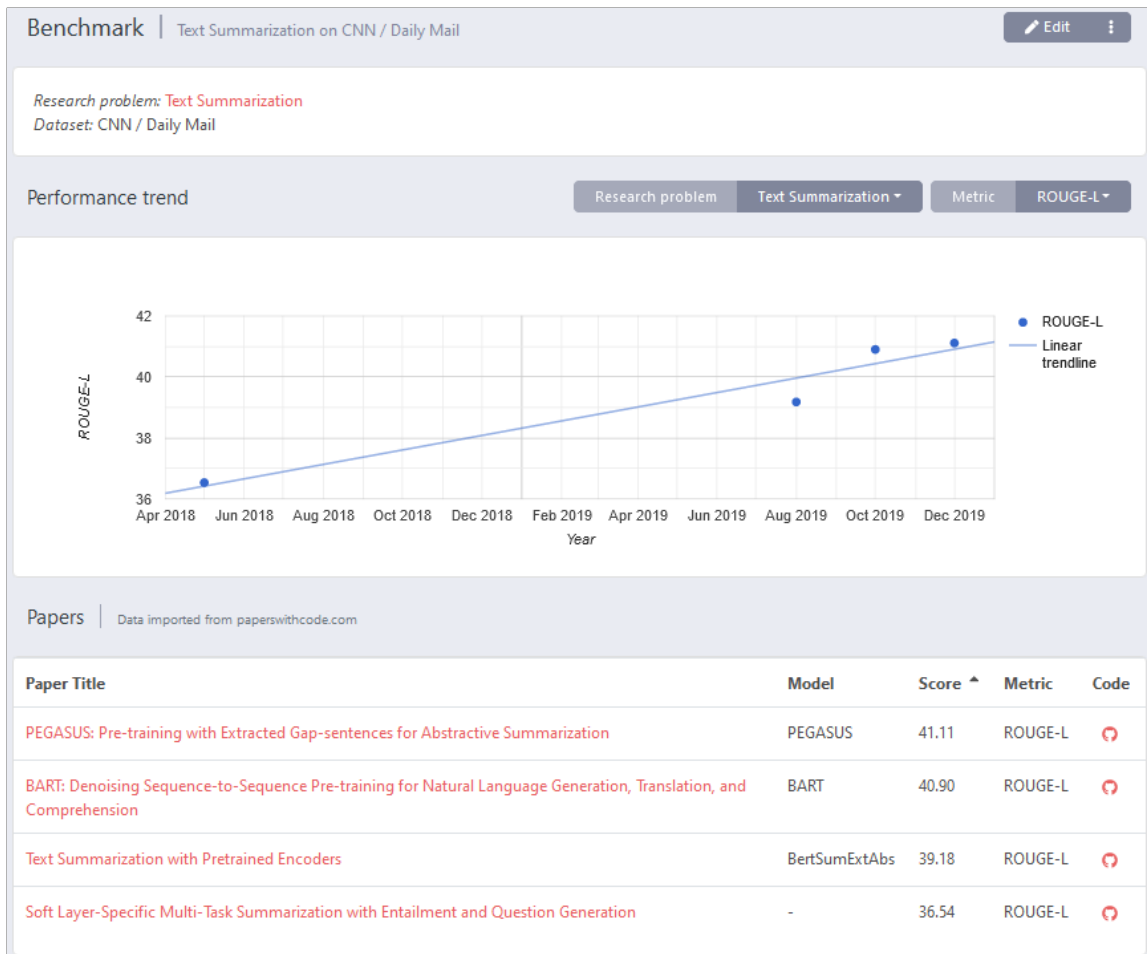


Figure 3.7: A benchmark page in the ORKG for the *Text Summarization* research problem on the *CNN/Daily Mail* dataset showing different systems evaluated with the *Rouge L* metric. A list of all benchmarks are online at <https://www.orkg.org/orkg/benchmarks>.

contributions. This proposes a novel direction in the writing of academic articles leveraging in part machine-actionable data in the context of discourse text.

# Scholarly Knowledge Graphs

Information Science   Sören Auer   Muhammad Haris   Markus Stocker   Allard Oelen

## Introduction

One of the key challenges for sciences in the context of digitalization is how we can transform scholarly communication to leverage novel digital possibilities. Over the last centuries, the way we communicate scholarly results has hardly changed. Still, we produce textual essays, which we can now after the arrival of digital devices and the internet easier produce and distribute. However, the new means of digital technologies, such as semantic representation, dynamic content as well as data and knowledge integration are not yet exploited. Increasingly, we see the inadequacy of scholarly communication through symptoms such as the proliferation of publications, the reproducibility crisis, or the deteriorating quality of peer review.

## Scholarly Identifier Systems

The basis for scholarly knowledge graphs are scholarly identifier systems. An identifier system defines how scholarly identifiers are coined and curated. It determines requirements on metadata, identifier persistency and versioning. Key identifier systems for scholarly communication include:

- [Document Object Identifiers](#) (DOI) as for example provided by DataCite ([Cruse, 2017](#))
- [ORCID](#) identifiers for researchers ([Ratner, 2012](#))
- [ROR](#) identifiers for research organizations ([Gould, 2019](#))

## Comparison of Scholarly Identifier Systems

Properties	<a href="#">DataCite: Lessons Learned on Persistent Identifiers for Research Data</a> 2017 - Contribution 1	<a href="#">ORCID: a system to uniquely identify researchers</a> 2012 - Contribution 1	<a href="#">Let's ROR Together: Building an Open Registry of Research Organizations</a> 2019 - Contribution 1	<a href="#">Persistent Identifiers for Research Organizations</a> 2020 - Contribution 1
Entity type	Scholarly artefacts	Researchers	Organizations	Organizations
Has research problem	Persistent Identification	Persistent Identification	Persistent Identification	Persistent Identification
Metadata schema	<a href="https://schema.datacite.org/">https://schema.datacite.org/</a>	<a href="https://github.com/ORCID/orcid-model/tree/master/src/main/resources/record_3.0">https://github.com/ORCID/orcid-model/tree/master/src/main/resources/record_3.0</a>	Basic metadata about an organization including name, alternate names, and location.	<a href="#">http://purl.org/net/ror-schema/1.0</a>
Number of entities	24000000	10000000	91000	91000

Figure 3.8: A SmartReview article created from structured surveys in the [ORKG](#). More information on creating smart reviews can be found here [https://www.orkg.org/orkg/help-center/article/6/SmartReviews\\_-\\_creating\\_review\\_articles\\_with\\_the\\_ORKG](https://www.orkg.org/orkg/help-center/article/6/SmartReviews_-_creating_review_articles_with_the_ORKG).

## Chapter 4

# Scholarly Contributions Similarity

As the first information retrieval service aspect in **ORKG**, this thesis addresses the similarity service over research contributions called **Simcomp**. It was first introduced by Jaradeh et al. [34] to help researchers view the most similar other **ORKG** contributions to the one they are currently browsing in a single page without the need to search for them. This feature is akin to a ‘most related pages’ feature in regular web pages. The difference in the **ORKG** is that the similarity is computed over *structured information* in contrast to web pages that has unstructured text; and the information focus is unique, i.e. scholarly contributions data. This offers two main benefits: 1) ease of browsing the **ORKG**; and 2) rapid survey creation process by selecting all the similar contributions bundled within a single view via the **ORKG** contributions comparison creator feature. Thus, in terms of the *Findable* data aspect of the **FAIR** Data Principles [75], the **ORKG Simcomp** service improves the findability of its research contributions.

While the standard information retrieval practices in scholarly digital libraries concern retrieving similar papers, slightly deviating from this norm, the **ORKG Simcomp** service computes similar research contributions. To this end, there are some particularities that need to be clarified at the outset. The **ORKG** is conceptually structured at the top-level w.r.t. an **Ontology**<sup>1</sup> in which a research paper node is a class that can contain one or more nodes about the research contributions of the paper. The **Simcomp** similarity service then discussed in the subsequent chapter is indeed based on the contribution nodes and not the paper nodes. As an example, **Figure 4.1** shows a paper structured in terms of its three different research contributions. Note also that two contributions from the same paper can be dissimilar from each other.

---

<sup>1</sup><https://gitlab.com/TIBHannover/orkg/orkg-ontology>

The remainder of this section is organized as follows. [Section 4.1](#) offers the technical details of [Simcomp](#). [Section 4.2](#) describes the novel gold-standard dataset created as part of this work to quantitatively evaluate the performance of the similarity service and enhancements also made in this work. [Section 4.3](#) analyzes the pros and cons of the existing service. [Section 4.4](#) describes the enhancements of this work to alleviate the service limitations. [Section 4.5](#) presents evaluations showing the impact of the enhancements made to [Simcomp](#). Finally, we conclude this part of the thesis with [Section 4.6](#).

## 4.1 Technical Description

This section presents the technical details of [ORKG](#)'s original [Simcomp](#) service that this thesis work aims to optimize.

[Simcomp](#) leverages an existing search engine viz. Elasticsearch. Elasticsearch [69] is a full-text, highly scalable, free, and open source search engine based on the Apache Lucene<sup>1</sup> library. It can store large volumes of schema-free data optimally using a NoSQL-Database [57] under the hood. Its [IR](#)-specific features are: it can compute similarity rankings over a set of documents; and it can generate an inverted index to keep track of documents enabling their lookup via keywords, which makes the retrieval process faster, easier, and more accurate. In other words, this latter feature called indexing is a data structure technique applied to optimize the performance of a database by looking up an index (or indices) before accessing the whole collection. This is akin to a lookup over the table of contents of a book to decide which chapter to read first. Some natural questions about the details may arise. Such as *i*) what method does Elasticsearch use to rank the documents and terms within an index?; And *ii*) how can an [ORKG](#) contribution be expressed in terms of what is considered an Elasticsearch document?

Elasticsearch uses the [BM25](#) algorithm [64], one of the most common text-retrieval algorithms, as its default ranking method. It's based on the Probabilistic Relevance Framework and state-of-the-art [TF-IDF](#)-like [60] retrieval ranking functions. As the word probabilistic suggests, the point here is computing the relevance score of a document (or set of documents) to a query. Equation 1 shows the [BM25](#) scoring formula between a document  $D$  and a query  $Q$  pair:

$$score(D, Q) = \sum_{i=1}^n IDF(q_i) * \frac{f(q_i, D) * (k_1 + 1)}{f(q_i, D) + k_i * (1 - b + b * \frac{|D|}{avgdl})} \quad (1)$$

---

<sup>1</sup><https://lucene.apache.org/>



Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge

November 2019 Databases/Information Systems Mohamad Yaser Jaradeh Allard Oelen Kheir Eddine Farfar Manuel Prinz  
 Jennifer D'Souza Gábor Kismihók Markus Stocker Sören Auer

Published in: Proceedings of the 10th International Conference on Knowledge Capture - K-CAP '19 DOI: 10.1145/3360901.3364435

Contribution Similarity Literature Comparison ORKG System

Research problems  Add to comparison  
 Similarity measures  
 Structured descriptions of research contributions

Contribution data

Architecture	No label Allard Oelen Kheir Eddine Farfar Mohamad Yaser Jaradeh
Description	Find similar research contributions inside ORKG and suggest them to the user
Utilizes	TF/IDF

Similar contributions

- 13% A Unified Semi-supervised Framework for Author Disambiguation in Academic Social Network Contribution 1
- 13% Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge ORKG System
- 13% Incremental unsupervised name disambiguation in cleaned digital libraries. Contribution 1

Compare these contributions

Provenance Timeline  
 Added on 19 Dec 2019  
 Contributors: Kheir Eddine, Jan Göpfert, Alexander Rodrigues Silva

Figure 4.1: Front-end view of the paper <https://www.orkg.org/orkg/paper/R8186> structured in 3 contributions (shown in different tabs). Contribution Similarity, Literature Comparison and ORKG-System. The research problems and data of the contribution “Contribution Similarity” (highlighted in red) are shown; as well as 3 of its similar contributions (in gray boxes).

where  $q_i$  is the  $i$ -th query term.  $f(q_i, D)$  is the  $q_i$ 's term frequency in the document  $D$ .  $|D|$  is the length of  $D$ .  $avgdl$  is the average document length in the entire documents collection.  $k_1$  and  $b$  are **Hyperparameters** that affect the term frequency saturation

and document length to average document length ratio, respectively.  $IDF(q_i)$  is given by equation 2:

$$IDF(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right) \quad (2)$$

where  $N$  is the total number of documents in the collection and  $n(q_i)$  is number of documents containing  $q_i$ . Knowing the scoring function, we now have an overview of how everything is being calculated and could imagine how Elasticsearch's indexing process looks like. An index is like a bag-of-words of all documents' terms, that are counted for each single document ( $f(q_i, D)$ ) and over all documents ( $n(q_i)$ ). These numbers are counted while indexing and should be updated when a new document comes into a precomputed index.

Concerning the second question about how an **ORKG** contribution can be equivalently seen as an Elasticsearch document, this can be explained as follows. Recall that a contribution is structured in the **ORKG** as a set of triple statements (**S**, **P**, **O**). The default process of making a document from these statements is simple. The labels of the subject, predicate, object triple are concatenated, then each triple is concatenated to the next triple expressed as its labels, and finally they are concatenated with the paper's title. In the **ORKG** context, search queries are also constructed in the same way since they are in fact contributions to which other similar contributions are to be computed. Reserved characters<sup>1</sup> when encountered need to be escaped at query time for Elasticsearch.

The components described should allow one to intuit how Elasticsearch works as a standalone service and its applicational relevance over **ORKG** contributions. Its **ORKG**-specific operational workflow as **Simcomp** is presented in Figure 4.2. First, the application is launched which then calls the initialization service. This service creates a document for every contribution in the **ORKG** and indexes it. At query time, the contribution viewed in the **Front-end** by an **ORKG** user is set as the query to the Elasticsearch index. Its document is looked up in the precreated index where **BM25** relevance scores [64] per query indexed document pair and ranked results can be returned. The Elasticsearch index is periodically updated to incorporate new contributions added to the **ORKG** or edits made to existing contributions.

**Simcomp** [51] offers RESTful Web APIs [63] using the Flask<sup>2</sup> Framework based on **Python**. The source code is hosted at the **TIB-Hannover** GitLab Repository.<sup>3</sup>

---

<sup>1</sup>[https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-query-string-query.html#\\_reserved\\_characters](https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-query-string-query.html#_reserved_characters)

<sup>2</sup><https://flask.palletsprojects.com/en/2.0.x/>

<sup>3</sup><https://gitlab.com/TIBHannover/orkg/orkg-similarity>

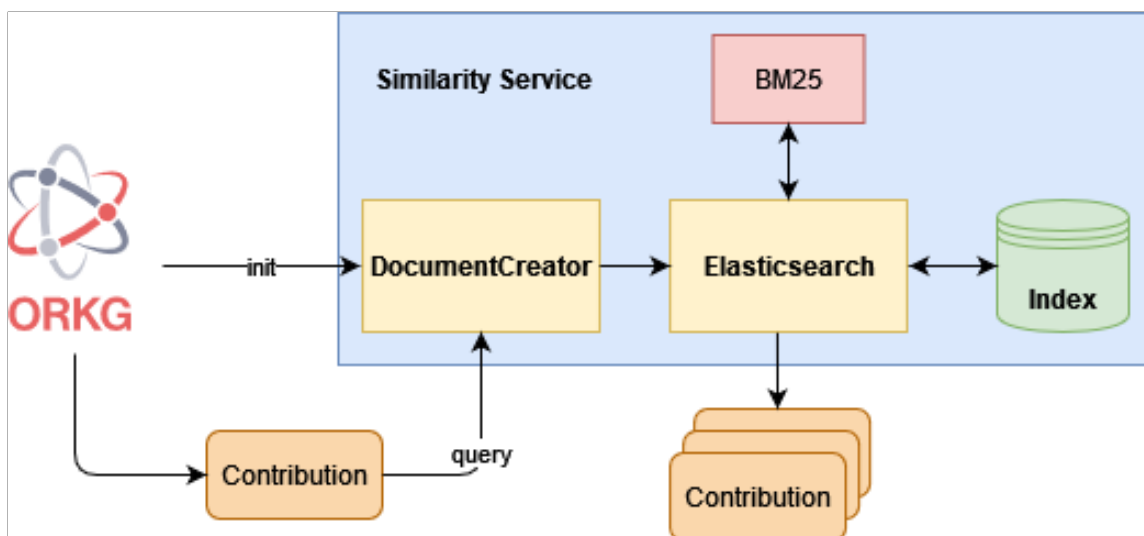


Figure 4.2: Similarity service operational workflow. Arrows show the data flow

## 4.2 ORKG Similarity Service Evaluation Corpus

In Computer Science, it is an established practice to create benchmarks, i.e. a predefined set of objects containing the parameters tackled in a problem over which performances are measured. In the context of the **Simcomp** service, a benchmark should thus be established. By measuring the service or its enhancements against the benchmark its effectiveness can be quantitatively gauged. Thus as the benchmark for **Simcomp** evaluations, in this work, a gold-standard evaluation corpus is created where the query contribution documents are manually assigned a set of contribution documents that are its true similar candidates. To create this evaluation corpus, a criteria put in place was that it needed to cover diverse scientific fields encompassing those that the Elasticsearch index is based on to ensure fair evaluation of the service. In this section, more about constructing this corpus is discussed.

Creating the evaluation corpus was not straightforward. Pairs of query contributions and their corresponding similar contributions had to be manually annotated. But which set of contributions should be selected as queries? The **ORKG** contains, at the time of this writing, a daily growing collection of 8866 contributions. It was not practical to look up all of the **ORKG**'s contributions to select the best candidates as queries. Instead, we restricted the search space to just the **ORKG** featured-comparisons<sup>1</sup> originally created by Oelen et al. [51]. Note that

<sup>1</sup><https://www.orkg.org/orkg/featured-comparisons>

the featured-comparisons are those that are specifically selected since they combine several comparable and, importantly, well-structured semantic representations of research contributions. For creating our evaluation corpus of contributions, the individual contributions were isolated from these featured-comparisons. Specifically, a SPARQL<sup>1</sup> query (shown in listing 4.1) was implemented on the ORKG's RDF-Dump<sup>2</sup> for the IDs of the featured-comparisons together with their contributions' IDs and research fields as a response.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX orkgp: <http://orkg.org/orkg/predicate/>
PREFIX orkgc: <http://orkg.org/orkg/class/>

SELECT ?comp ?contr ?paper ?field
  WHERE {
    ?comp a orkgc:Comparison, orkgc:FeaturedComparison ;
          orkgp:compareContribution ?contr .
    ?paper orkgp:P31 ?contr ;
          orkgp:P30 ?field.
  }
```

Listing 4.1: SPARQL query to retrieve featured-comparisons and their respective contributions and research fields.

In all, 26 featured-comparisons were obtained consisting of 292 contributions. From the 292, 100 were randomly chosen as the evaluation corpus. Because there were already a known problem by the baseline similarity service, that is, for some contributions it does not respond with any similar candidates, we decided to have the 100 instances with the property that 50 of them have similarities in the response, where the other 50 do not. Further, to satisfy the corpus creation criteria, we ensured that the contributions were distributed over various research fields. Figure 4.3 shows the distribution of the “with response” 50 contributions in the evaluation corpus over 6 research fields. The statistics of the “without response” 50 contributions are summarized in Table 4.1.

By now the test instances and their actual values (by default the top-5 similarities are retrieved) were constructed and the next step will be manually annotating the instances. It is important to annotate in a systematic way, so one can assume the reliability in the annotations. As discussed earlier, instead of looking for similar candidates in the entire ORKG by the manual annotator, we bound the search space

---

<sup>1</sup><https://www.w3.org/TR/rdf-sparql-query/>

<sup>2</sup><https://orkg.org/orkg/api/rdf/dump>

for each test instance by leveraging the featured-comparisons. The search space  $S$  of test set’s instance  $I$  is defined by [Equation 3](#):

$$S(I) = C(I) \cup Sim_{k=10}(I) \quad (3)$$

where  $C(I)$  are the compared candidates of  $I$  and  $Sim_{k=10}(I)$  are the top-10 similar candidates returned by the similarity service. An appropriate number of service’s results were used to not only restrict the search space to the featured-comparisons, but to some other contributions in the [ORKG](#). Note that the default number of results is  $k=5$ . The number was increased to  $k=10$  on the local machine in order to enlarge the search space and evaluate the service properly (see [Section 4.5](#)). In the meanwhile, we differentiated between homogeneous and heterogeneous instances. A homogeneous case is, for instance, a test contribution<sup>1</sup> is very similar to its compared contributions<sup>2</sup> and they only differ by one or very few number of properties. In this case all homogeneous candidates were considered as annotations for this test contribution. The heterogeneous case is self-explaining the opposite, therefore, only the top-1 candidate was considered as an annotation. Overall there are 62 homogeneous and 38 heterogeneous instances.

To sum up, the test set consists of 100 instances divided in two partitions, namely, “with response” and “without response” with 50 instances for each. We also distinguish the annotation type of one instance between homogeneous instances with several annotations and heterogenous instances with exactly one annotation, i.e. each instance of the test set is manually annotated with 1- $n$  annotations as expected similarities, where  $n$  is the number of all homogeneous candidates of a test instance.

Research Field	# Test Set Instances
Environmental Sciences	7
Climate	27
Virology	11
Artificial Intelligence	5

Table 4.1: Test set’s “without response”-part distribution over research fields.

<sup>1</sup><https://www.orkg.org/orkg/resource/R39049>

<sup>2</sup><https://www.orkg.org/orkg/comparison/R39082>

### 4.3. Analysis of the Default **ORKG** Contributions Similarity Service

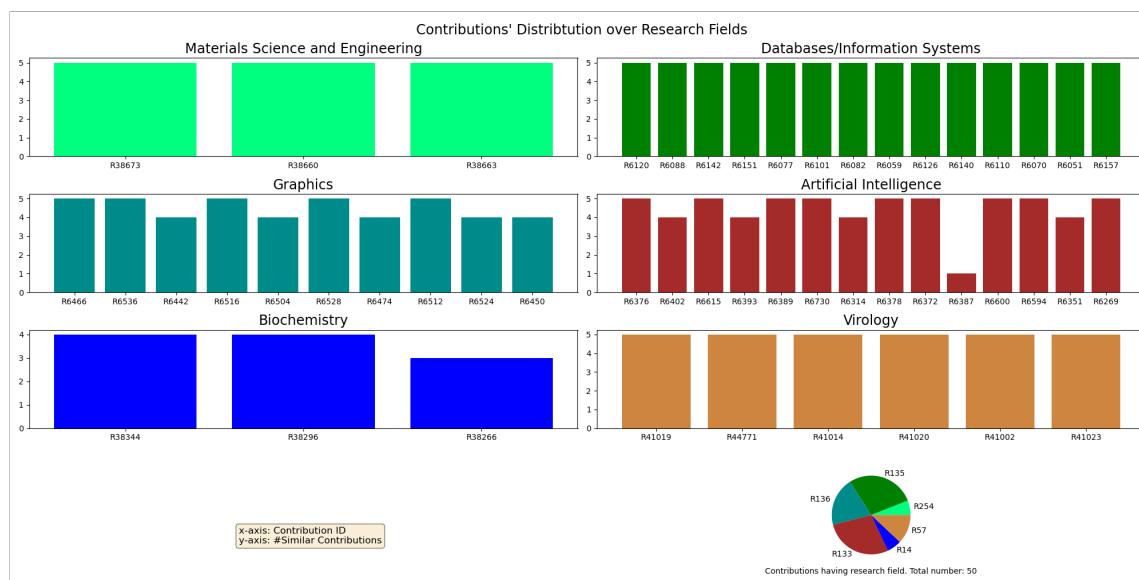


Figure 4.3: The “with response” evaluation corpus contributions distribution over research fields. Each subplot is a research field, the x-axis is the query contribution instances, and the y-axis is the number of similar contributions returned by the similarity service sharing the same research field as the query. The query contributions with lower number of similar contributions sharing the same research field either got only that set of contributions in the response (e.g. R6387) or got contributions from research fields similar to its own (Biochemistry and Virology e.g. R38266).

## 4.3 Analysis of the Default **ORKG** Contributions Similarity Service

In this section, the default **ORKG** similarity service is reviewed. Conducting such a review enables one to identify the precise engineering enhancements to improve the service effectiveness. For this, each step of the operational workflow shown in **Figure 4.2** was analyzed and tested for possible failures. The service was analyzed for unexpected results against a set of randomly selected contributions as queries and their expected results. This was considered the service development set and was not part of the evaluation corpus created in this work.

### 4.3.1 The DocumentCreator Module

In this step, the expected outcome, as described earlier in [Section 4.1](#), is a concatenation of the labels of the contribution’s triples and the respective paper’s title. The statements and the paper’s title are fetched directly from the [ORKG](#)’s data stored in the Neo4j<sup>1</sup> database using two [Cypher](#)<sup>2</sup> queries. The first query should have the contribution’s statements as response after expanding all connected nodes to the contribution’s node recursively until the label nodes are reached. At this point, the question arose whether the query could be cyclic in the case an object in the subgraph references the same contribution subject’s node. The answer was no, since the used procedure<sup>3</sup> in the query discards such cycles and prevents infinite loops. The second query should have the paper’s title as response, but it does not because of a source code bug and instead returns “None” as response. An example document is shown in [Listing 4.2](#). There was also redundancy noted in the generated document as subjects and predicates were repeated if there is, for example, a list of objects for a specific predicate. These repetitions could misleadingly bias the score of one document over another in the querying process, since the [BM25](#) [64] ranking function counts the occurrences of terms in a document. Another observation was that the objects’ labels could be long text descriptions containing relevant as well as noisy irrelevant terms for the contribution itself. Including such texts made it hard to control the semantic score for that contribution as a meaningful number of similarity to other contributions. Thus such descriptive labels were undesirable in the document.

```
None Contribution 1 Evidence Abstract Contribution 1 Evidence
Author name Contribution 1 has research problem Author name
disambiguation Contribution 1 uses similarity Cosine
Contribution 1 Uncertainty F Contribution 1 Performance
metric F1 Contribution 1 deals with Homonyms problem
Contribution 1 Limitations If relationship information is
not available then this method would not perform well
Contribution 1 Performance metric Pairwise accuracy
Contribution 1 Performance metric Precision Contribution 1
Evidence References Contribution 1 dataset Self designed
Contribution 1 Evidence Title words Contribution 1 Method
Unified Probabilistic framework with Markov random fields
Contribution 1 approach Unsupervised Learning Contribution
1 Evidence Year Contribution 1 Evidence venue
```

<sup>1</sup><https://neo4j.com/>

<sup>2</sup><https://neo4j.com/developer/cypher/>

<sup>3</sup><https://neo4j.com/labs/apoc/4.0/graph-querying/expand-subgraph/>

---

Listing 4.2: The document representation of the contribution R6088 in the default Contributions Similarity Service.

### 4.3.2 The Elasticsearch Module

In this step, the **NLP** pre-processing of the labels text applied by Elasticsearch was examined. It makes use of the Standard Analyzer,<sup>1</sup> consisting of a Standard Tokenizer<sup>2</sup> which is grammar-based and a Lowercase Token Filter<sup>3</sup> that lowercases the given text. **Listing 4.3** gives an example of applying the standard tokenizer on a synthesized example text. Some observations are *i*) stop words are not filtered and *ii*) hyphenated words are not compounded. For instance, “Co-authors” should be considered as “Coauthors”, while “linked-data” can be considered as “linked” “data”.

Further, in the default service querying step, after retrieving the similar contributions, all contributions with a similarity percentage greater than 80% are being discarded. This threshold was chosen empirically, assuming that very similar contributions are identical and identical ones cannot be similar. This is actually a strict assumption, since each contribution has a unique identifier in the **ORKG** and therefore, there are in principle no identical contributions. Note that the similarity percentage is computed by:

$$\text{SimilarityPercentage}(D, Q) = \frac{\text{score}(D, Q)}{\text{score}(Q, Q)} \quad (4)$$

where the score function is given by equation 1.

```
text: The author and his co-authors wrote a paper about linked
      -data
tokens: "The", "author", "and", "his", "co", "authors", "wrote
        ", "a", "paper", "about", "linked", "data"
```

Listing 4.3: A tokenization example using the Elasticsearch’s standard tokenizer.

---

<sup>1</sup>[https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-standard-analyzer.html#\\_definition\\_4](https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-standard-analyzer.html#_definition_4)

<sup>2</sup><https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-standard-tokenizer.html>

<sup>3</sup><https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-lowercase-tokenfilter.html>



When noting the Elasticsearch responses to queries, it returns all indexed documents sorted by the **BM25** score. The question then is, should there be a lower-bound threshold? Consider that the **ORKG**'s **Front-end** displays the similarity percentage between the browsed and each of the retrieved contributions. This places the decision in the hands of the end-user to include a returned contribution in a comparison. The resulting conclusion to this end is that discarding less similar contributions limits the user's choices so it is preferred to display them anyway, as opposed to no results, despite low similarity.

### 4.3.3 The Default **ORKG** Contributions Similarity Service Response Analysis

Apart from investigating the technical details of the individual default **Simcomp** modules, there is also a need to analyze the responses for meaningfulness from the service to randomly selected **ORKG** contribution queries. This is discussed next w.r.t. the top-ranked query response:

1. ***Survey contribution returned as top-ranked response to query contribution system surveyed.*** The top-ranked result for contribution R6402 (with paper's title "Natural language questions for the web of data" and research problem "Question answering systems") is R9469 (with paper's title "Survey on Challenges of Question Answering in the Semantic Web" and research problem "Ambiguity in QA"). R9469 surveys various systems including the contribution R6402. The systems surveyed share similar structures, therefore, the query terms occur multiple times in the R9469 structured contribution description, which explains its high rank.
2. ***Elasticsearch supports only surface term similarity matches without semantic reasoning capacity.*** The responses to R112273 (with paper's title "Klimaschutzszenario 95 (KS95)" and research problem "Future energy and emission scenario predictions") have almost nothing semantically in common with it. The term "Assumptions" in R112273 is repeated many times in the query and it is included in the first result R31051. Having a frequent term in the query increases its importance in turn positing it as a similar candidate. Note then that this highlights the term-based dependency of Elasticsearch over deeper semantics which it does not possess.
3. ***Different user-specified structuring of two contributions with nearly the same contribution information.*** The common terms of R6466 (with

paper’s title “Visualizing Populated Ontologies with OntoTrix” and research problem “Graph-based visualization systems”) to the query contribution R25719 (with the same paper’s title and research problem “Data Visualization”) were structured one level deeper in the hierarchy by the user.<sup>1</sup> Whereas R25719 contains exactly the same terms one-level higher as structured by a different user. In principle, both are two different contributions, since there is a unique identifier for each, but they are semantically identical thus they wouldn’t be good candidates to compare in an **ORKG** comparison. Nevertheless the default service response in this query context is sound.

4. ***Loss of valid similar contribution candidates due to the upper-bound 0.8 threshold of the default **Simcomp** service.*** The paper of the contribution R41020 consist of 6 contributions that are similar to each other and mainly differ by the city location, where the COVID-19 cases are being reported. R41020 reports the cases in “Hubei, China” and the response of the baseline R41025 reports the cases in “International (46 countries)”. While R41025 is a valid similar contribution result to R41020, there are other valid candidates that should be higher ranked since they share a greater similarity with the query. For instance, R41019 reports the cases in “Rest of Hubei, China”, which indeed shares a greater similarity with the query. The reason this result is not returned in the default service is because its similarity measures over 0.8 which in the default service is set as the discarding upper-bound. Note that the 3 mentioned contributions belongs to the same paper with title “Unique epidemiological and clinical features of the emerging 2019 novel coronavirus pneumonia (COVID-19) implicate special control measures” and have the same research problem “COVID-19 case fatality rate”.

## 4.4 Thesis Implementation of **Simcomp** Enhanced

After analyzing the default service workflow, a summary of the service limitations addressed in this thesis are as follows. 1) The DocumentCreator Module output will be re-engineered such that repetition of terms in the existing method of traversing contributions will not occur, thus the side-effect of biasing similarity scores of the contribution document will be averted; 2) the paper title no response will be fixed; 3) the upper-bound 80% threshold will be dropped; 4) objects with lengthy descriptive labels will be dropped; and 5) the default Elasticsearch **NLP** pre-processing w.r.t. hyphenation and stop words filtering will be addressed.

---

<sup>1</sup>The graph hierarchy of the contribution

First, the new DocumentCreator module implemented as part of this thesis is described. [Figure 4.4](#) shows an illustration with a contribution’s snippet from the [ORKG](#). It works in the following steps.

1. The statements (S, P, O) of the contribution are fetched from the Neo4j database and joined with slash characters to compose triples as raw paths.
2. Considering the contribution’s title - “Contribution 2” in the example - the root and as long as there are still paths that do not start with the root, the DocumentCreator iterates over all paths, takes the object of the current path, looks for paths starting with that object, replaces the object with those paths and finally removes them. This step is prone to cyclic contribution graphs, therefore, the implementation of this work discards such cycles and ensures not getting stuck in an infinite loop.
3. The composed paths have in principle the same structure as a file system, if one considered the contribution’s title as root directory, the intermediate resources as subdirectories and the labels as files. The document creator removes the repetitions properly and forms a [YAML](#)-structured document.
4. All whitespace characters are trimmed to form a document similar to the one of the baseline.
5. Both the paper’s title as well as the paper’s research field are set as prefix to the document. [Listing 4.4](#) shows the simplified [YAML](#)-structured document of the R6088 contribution, which can be compared with [Listing 4.2](#).

```
A Unified Probabilistic Framework for Name Disambiguation in
Digital Library Databases/Information Systems Contribution
1 Evidence Year Author name Title words Abstract References
venue Limitations If relationship information is not
available then this method would not perform well Method
Unified Probabilistic framework with Markov random fields
Performance metric F1 Pairwise accuracy Precision
Uncertainty F approach Unsupervised Learning dataset Self
designed deals with Homonyms problem has research problem
Author name disambiguation uses similarity Cosine
```

Listing 4.4: The document representation of the contribution R6088 using the [YAML](#)-structured document creator

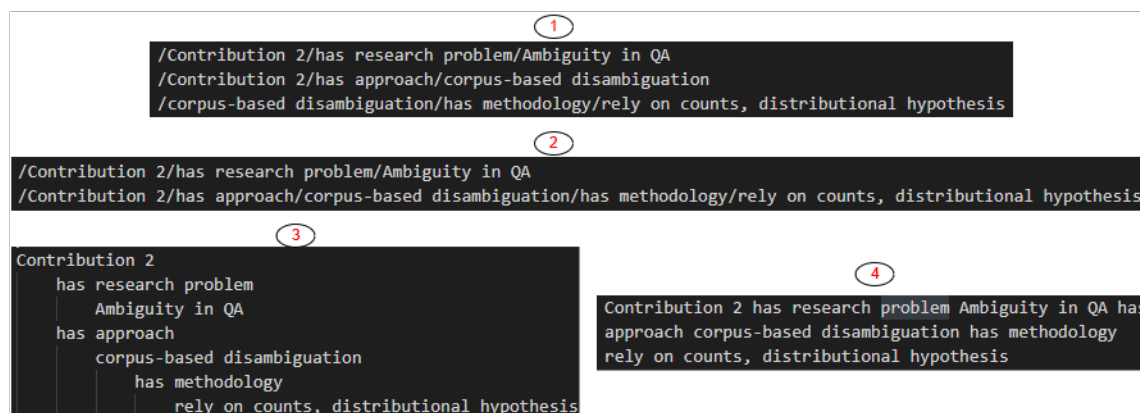


Figure 4.4: YAML-structured improved document creation process with example statements of the R9469 contribution. The red circled numbers are the order of the steps.

Next, w.r.t. dropping descriptive object labels from the document created for Elasticsearch as a representation of contributions. This is done by setting an upper-bound permissible phrase length threshold. [43] reports that 90% of information is when sentences averaged 14 words in English. Since the DocumentCreator targets phrases and not sentences, half the length i.e. 8 words is fixed as an upper-bound for valid phrases. Thus any phrases longer than 8 words are not included in the contribution resulting document.

Finally, the **NLP** pre-processing modules are addressed. Hyphenated words are handled with the Whitespace Tokenizer<sup>1</sup> by defining a custom analyzer for Elasticsearch, to make the hyphen handling take effect. Table 4.2 shows occurrence cases of hyphens in a text and how they are handled. Further, a custom stop word filter is added to the Elasticsearch’s analyzer. This work used the English stop words<sup>2</sup> from **NLTK**<sup>3</sup>.

Last and as a technical modification to speed up searching the index, the indexing process now indexes contributions are by their research field, i.e. one index is created for each research field, as opposed to indexing all contributions in a single index.

<sup>1</sup><https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-whitespace-tokenizer.html>

<sup>2</sup><https://gist.github.com/sebleier/55428#gistcomment-2596130>

<sup>3</sup><https://www.nltk.org/>

Case	Before Handling	After Handling
alphanumeric-alphanumeric	Linked-data IPSL-CM5	Linked data IPSL CM5
alphanumeric*-alphanumeric	U-Velocity Co-author 12-December	UVelocity Coauthor 12December
alphanumeric-alphanumeric*	QALD-5 C-Mn	QALD5 CMn
numeric-numeric	1990-1995	1990-1995

Table 4.2: Hyphen occurrence cases in a text with examples before and after handling. alphanumeric\* is any alphanumeric text with a maximum length of 2.

## 4.5 Evaluations and Discussions

With the enhancements in place, this section discusses evaluations in depth. The following list shows the enhancements applied per implementation version, where each<sup>1</sup> version was called on the **ORKG**'s Neo4j-Dump from *23.06.2021* and returned 10 responses:

- v0: The default **Simcomp** service as the baseline system.
- v1: v0 is fixed. For instance, the discarding threshold is suppressed and other error-prone places in the source code that could lead to a program crash while calling the initialization service are fixed.
- v2: v1 + paper's title and research field are attached to the created document.
- v3: v1 + discarding of long literals and escaping the reserved special characters.
- v4: v1 + **YAML**-structured document instead of the baseline's document (repetitions reduction). The document creation process includes escaping the reserved special characters.
- v5: v4 + paper's title and research field are attached to the created document.
- v6: v5 + discarding of long literals.
- v7: v6 + hyphen handling in combination with configuring the Elasticsearch's tokenizer as well as filtering the **NLTK** stopwords.

<sup>1</sup>Except v0, which returned no results in the "without response" part.

- v8: v4 + creating an Elasticsearch’s index for each research field.

Recall that our test set described in [Section 4.2](#) has specific categorical groupings which themselves are leveraged in different evaluation scenarios. Consider that they are in four categories: the homogeneous set, the heterogeneous set, the “with response” contribution query set, and the “without response” contribution query set. In addition, a qualitative analysis over a small blind set similar to [Subsection 4.3.3](#) will be discussed.

### 4.5.1 The “with response” and “without response” Results

The first idea was to evaluate the two 50 query instances sets created regardless of the homogeneity criteria between the query and their annotated paired contributions as valid responses. The evaluation formula is shown in [Equation 5](#). It is a simple counting formula inspired after the standard recall metric that adds 1 whenever the top-10 responses to a query include at least one of the annotated valid response contributions. Note that while loosely inspired after recall, it does not adhere strictly to the definition of recall as we custom define the numerator to measure the percentage of test queries that get a valid response in their top-10 automatically returned responses from [Simcomp](#).

$$\text{Valid Response Approximation} = \frac{1}{|Q|} \sum_{q=1}^{|Q|} f(q, k) \quad (5)$$

where  $k$  is the response set from [Simcomp](#),  $Q$  is the test set of queries, and  $f(q, k)$  is a function that evaluates to 1 if top-k retrieved results includes at least one relevant document illustrated in [Equation 6](#):

$$f(q, k) = \begin{cases} 1, & \text{if } \text{Valid}(q) \cap \text{Response}(q, k) \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

where  $\text{Valid}(q)$  is the set of valid response contributions as defined in the gold-standard set for query  $q$  and  $\text{Response}(q, k)$  is the set of top-k contribution responses from [Simcomp](#) for query  $q$ .

The results of applying the “Valid Response Approximation” metric on the test set are shown in figures [4.5](#) and [4.6](#) for the “with response” and “without response” sets, respectively. In the “with response” set, all the enhanced [Simcomp](#) versions (all bars except the first blue bar) clearly outperform the default [Simcomp](#) service in all result groups. Between the enhancements themselves, one sees similar performances

between the **Simcomp** versions for the top-5 to top-10 result groups. In the top-1 block, however, v4 clearly stands out outperforming the other versions followed by v8. This implies that indeed our intuitions for removing the threshold and implementing a new method for creating documents is empirically verified are worthy enhancement contributions to the service (v4). In the “without response” set each top-k block has a different version that outperforms the others. This we ascribe to the crude nature of our “Valid Response Approximation” metric as offering only an approximate idea of performance. Note that since no information about response ranks are incorporated in it, it is only an approximate measure of performance. A precise measure should consider the ranks of relevant document in the results list. Thus subsequently for the heterogeneous and homogeneous sets’ evaluations the **MRR** and the **MAP** metrics in **Subsection 4.5.2** and **Subsection 4.5.3** are respectively leveraged.

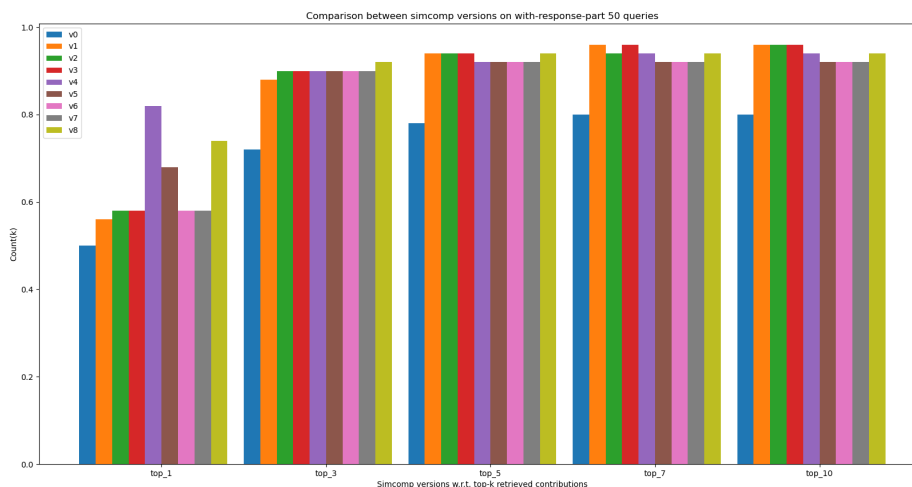


Figure 4.5: Applying the “Valid Response Approximation” metric over the “with response” part of the test set. The colors indicate the implementation version. Bar blocks indicate top-k results.

## 4.5.2 Heterogeneous Set Results with Mean Reciprocal Rank

The Reciprocal Rank (**RR**) measure [19] calculates the reciprocal of the rank at which the first relevant document was retrieved. **Equation 7** shows the **MRR** metric which

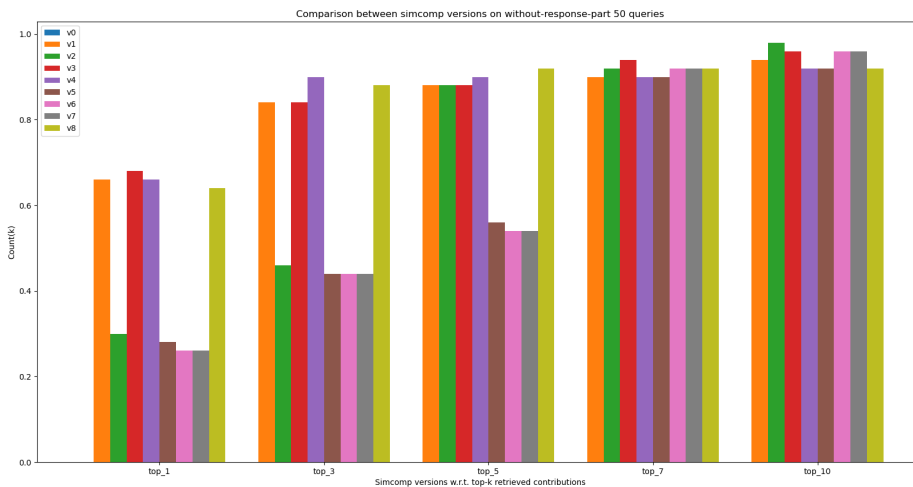


Figure 4.6: Applying the “Valid Response Approximation” metric over the “without response” part of the test set. The colors indicate the implementation version. Bar blocks indicate top-k results. The blue bars of v0 are not to be seen, since the baseline returns no results in this part of test set, hence the name.

sums the reciprocal ranks of queries and normalizes them by the number of queries:

$$MRR = \frac{1}{|Q|} \sum_{q=1}^{|Q|} \frac{1}{rank_q} \quad (7)$$

where  $Q$  is the set of the heterogeneous queries and  $rank_q$  is the rank position of the first relevant document for the  $q$ -th query.

To justify how **MRR** is a better choice than the “Valid Response Approximation” metric, both metrics were applied to evaluate the heterogeneous set consisting of 38 queries. The results are shown in Figures 4.7 and 4.8 for the two metrics, respectively. To offer an intuition, **MRR** accounts for the rank as well as the number of true results and performs a tradeoff between the two. On the other hand, the crude metric we devised only accounts for the true results returned in the top-k set which is its limitation since it has no notion of ranks. Consider the case when 4 results are returned in the top-10 and 5 results are returned in the top-10 but ranked low, the **MRR** metric can account for the difference in the results, but the crude metric cannot. Further, it will simply evaluates the two cases which may not be the best



desired outcome for an information retrieval where it is important that the relevant results are top-ranked. Note from our exemplars contrasting the two metric results in Figures 4.7 and 4.8, the **MRR** metric clearly indicates the best version in any scenario of the top-k groups.

Per **MRR** results depicted in Figure 4.8, v4 outperforms all other versions with a score around 80% followed by v8. Versions 6 and 7 have no difference regardless the number of results retrieved implying that handling the hyphens and filtering the stop words do not affect the performance of the contributions similarity service. Adding the paper’s title and research field is a noisy input for **YAML**-structured documents on the one hand when versions 4 and 5 are compared, but on the other hand it does slightly enhance the performance for the baseline’s structured documents when versions 1 and 2 are compared.

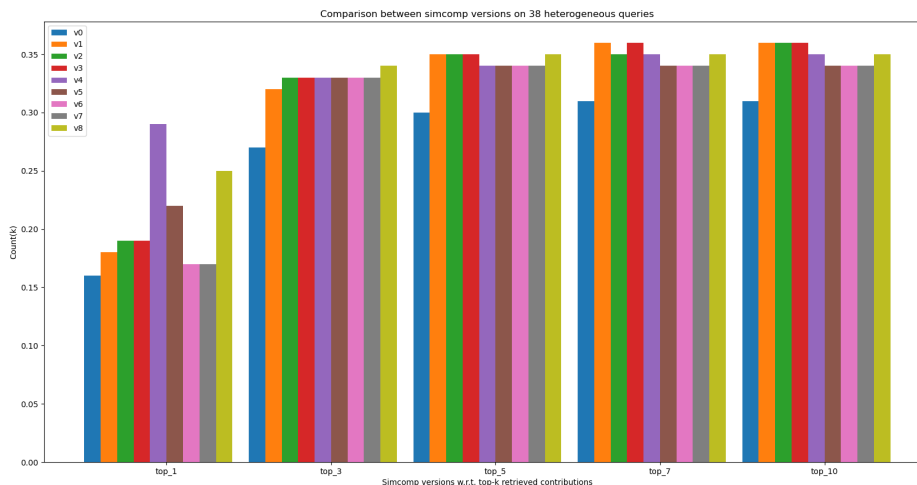


Figure 4.7: Applying the “Valid Response Approximation” metric over the heterogeneous instances of the test set. The colors indicate the implementation version. Bar block groups indicate top-k results.

### 4.5.3 Homogeneous Set Results with Mean Average Precision

While **MRR** accounts for rank position of the relevant document in the retrieved results, the Mean Average Precision (**MAP**) [44] not only accounts for the rank

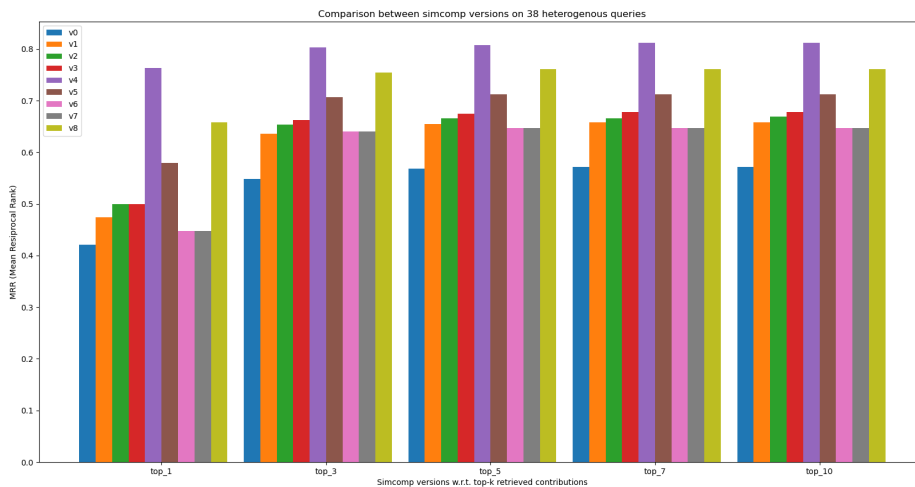


Figure 4.8: Applying the **MRR** metric over the heterogeneous evaluation set. The colors indicate the implementation version. Bar block groups indicate top-k results.

position but also for the number of relevant documents that have been retrieved in the top-k results. Thus for the 62 homogeneous test set queries, the **MAP** metric is better suited than **MRR**. Recall that in the homogeneous set there are multiple equally valid contribution responses in the gold-standard to a query. This is in contrast to the heterogeneous set where there is clearly just one most similar contribution response candidate. Thus evaluating with **MAP** which offers an idea of how many of the true responses are returned in the response set rather than just whether the one most similar response is high ranked is more meaningful for the homogeneous set. The **MAP** metric is given by [Equation 8](#).

$$MAP = \frac{1}{|Q|} \sum_{q=1}^{|Q|} AveP(q) \quad (8)$$

where  $Q$  is the set of queries and  $AveP(q)$  is the Average Precision of query  $q$  and is given by [Equation 9](#):

$$AveP(q) = \frac{1}{|Relevant(q)|} \sum_{k=1}^N P(q, k) * rel(k) \quad (9)$$

where  $Relevant(q)$  is the set of relevant documents for query  $q$ ,  $N$  is number of retrieved results,  $P(q, k)$  is Precision at  $k$  for query  $q$  and  $rel(k)$  is an indicator function that evaluates to 1 if the item at rank  $k$  is a relevant document, zero otherwise. The Precision at  $k$  for query  $q$  is given by Equation 10

$$P(q, k) = \frac{|Relevant(q) \cap Retrieved(q, k)|}{|Retrieved(q, k)|} \quad (10)$$

where  $Retrieved(q, k)$  is the set of  $k$  retrieved documents for query  $q$ .

The results of applying the MAP metric over the different versions is shown in Figure 4.9. It is again evident that version 4 outperforms the others in the top-3 results and has a slight lower MAP score than versions 1 and 3 in the top-10 case. But since users are mostly interested in the top-3 similar contributions, version 4 remains justified. Similar to the heterogenous case, v6 and v7 do not have any difference. Paper’s title and research field are noisy input for both YAML and baseline structured documents and discarding the long literals enhances the performance a little further but still has a better performance than that by version 1 and 4.

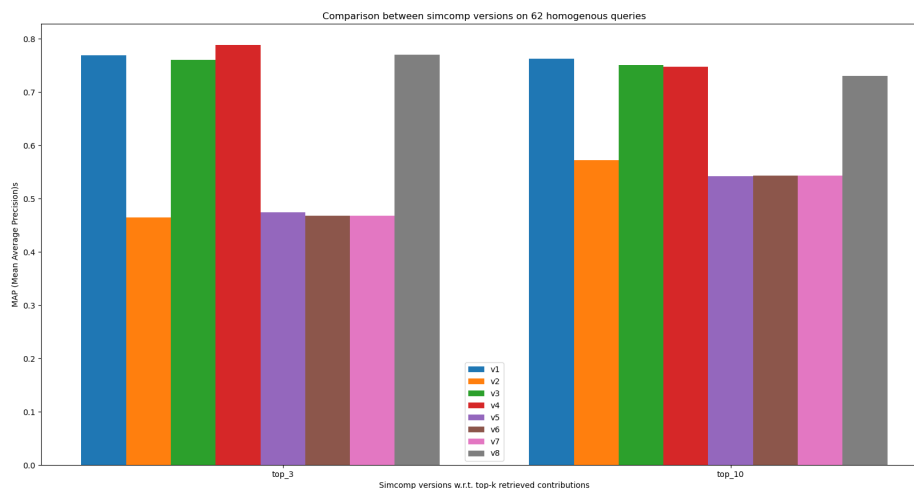


Figure 4.9: Applying the MAP metric on the homogeneous instances of the evaluation set. The colors indicate the implementation version. Bar blocks indicate top-k results. The v0 bar is not visible since the baseline returns no results in most of the homogeneous instances of the test set.

## 4.5.4 Qualitative Analysis

### Random **ORKG** Contributions

In addition to the test set discussed in [Section 4.2](#), this work discusses 4 additional random chosen contributions from the **ORKG**, their similar results have been queried by all implemented versions, in order to acquire a qualitative understanding after having different modifications. Note that the end-to-end response analysis conducted in [Subsection 4.3.3](#) was done before the modifications to gain an intuition of the existing problems in the default **Simcomp**. Each version was called on the **ORKG**'s Neo4j-Dump dated 21.07.2021. The queries and top-1 results are shown in [Table 4.3](#) and discussed next. We recommend the reader to read each of the following points side by side with the respective column of the mentioned table:

1. **R135270** (with paper's title "On nonlinear forced vibration of nano cantilever-based biosensor via couple stress theory" and research problem "Theoretical considerations on the performance of nanocantilevers as biosensors"): For this query v0 returns no results and other versions return either R135263 (with paper's title "Modeling the material structure and couple stress effects of nanocrystalline silicon beams for pull-in and bio-mass sensing applications") or R135267 (with paper's title "Optimization of a nano-cantilever biosensor for reduced self-heating effects and improved performance metrics"). Both contributions have the same research problem as the query and are structurally and semantically similar to the query and are included in the top-3 results of all versions. The difference is how each version counts the terms in its point of view. For instance, v6 and v7 ignores long literals, which form the majority of the query in this case.
2. **R135171** (with paper's title "Domain Adaptive Transfer Learning on Visual Attention Aware Data Augmentation for Fine-grained Visual Categorization" and research problem "Image Classification"): Versions 0, 1 and 2 return an error when queried that a special reserved character is not handled and need to be escaped to be included in the query. This issue is solved by the special character handling and the new DocumentCreator. All other versions respond with R134925 (with paper's title "Sharpness-Aware Minimization for Efficiently Improving Generalization" and the same research problem as the query) as most similar instance, which is reasonable, since both contributions are about image classification models.
3. **R135183** (with paper's title "Domain Adaptive Transfer Learning on Visual Attention Aware Data Augmentation for Fine-grained Visual Categorization")

and research problem “Fine-Grained Image Classification”): Versions 0, 1 and 2 have the same previous problem. Both top-1 responses R135098 (with paper’s title “Sequential Random Network for Fine-grained Image Classification” and research problem “Image Classification”) and R135180 (that comes from same paper as the query but has the research problem “Image Classification”) are semantically reasonable, and therefore, similar to the query. Version 4 includes both contributions in its results. Versions 5, 6 and 7 prefer R135180 because it belongs the same paper.

4. **R135476** (with paper’s title “An Ontology-Based Approach for Curriculum Mapping in Higher Education” and research problem “Ontology Learning”): Structurally, both R135479 (with paper’s title “A learning object ontology repository to support annotation and discovery of educational resources using semantic thesauri” and research problem “ontology creation”) and R136012 (with paper’s title “Ontology-Based Personalized Course Recommendation Framework” and research problem “Ontology mapping”) are similar to the query, but only R135479 is semantically similar to the query, since they are about educational topics. R136012 was as usual chosen because of the similarity in the paper’s title. The contribution R8466 is a very good example showing that restricting the search space to one research field could be a bad limitation. Since the **BM25** scoring function only considers terms and documents frequency without checking any semantic meanings and version 8 restricts the search space, R8466 was highly scored. Whereas the more similar ones are not assigned to the same research field. Furthermore, browsing the top-10 results produced by v4 shows that all results are (partially) meaningful to this query since all results are in one way or another about ontology-based approaches in the education field and their similarity percentages are in the range between 17.2% and 55.8%.

Thus, one can conclude that versions 3 and 4 could always return the best top-1 result. But, version 4 still has the advantage, that it includes both similar results that other versions have in the case of the query R135183.

### Revisiting the Response Analysis

In general, the results of both **MRR** and **MAP** metrics show that version 4 outperforms its competitors, but there is a need to investigate further whether its results are interpretable. Revisiting the response analysis **Subsection 4.3.3** is a good comparison for that matter. **Table 4.4** shows a comparison between versions 0, 1 and 4 w.r.t. the top-1 result and it’s interpreted as follows:

Version \ Query	R135270	R135171	R135183	R135476
v0	None	Error	Error	R135479
v1	R135263	Error	Error	R135479
v2	R135263	Error	Error	R136012
v3	R135263	R134925	R135098	R135479
v4	R135263	R134925	R135098	R135479
v5	R135263	R134925	R135180	R136012
v6	R135267	R134925	R135180	R136012
v7	R135267	R134925	R135180	R136012
v8	R135263	R134925	R135098	R8466

Table 4.3: Top-1 results of the different implementation versions over the blind set’s queries.

1. R6402: v4 returns R6320 as a first result which is a contribution that have the same structure as R6402. The survey is not being returned, since the number of repetitions have been plainly reduced.
2. R112273: v1 has improved the performance by fixing the software failures, since R112367 is very similar to the query. But v4 chose R112384 as the most similar candidate, because of the term “power”, which occurs in both R112273 and R112384 exactly 2 times whereas it occurs only 1 time in R112367. Having this term as a literal for the predicate “Includes technologies” is more important than counting the repetitions.
3. R6466: v4 returns in this case a contribution that has the same hierarchy as the query and is semantically not identical yet similar to the query.
4. R41020: Both v1 and v4 return the same result, since the problem in this case was obviously because of discarding similar candidates with similarity percentage greater than 80%.

### Versions Responses Discussion

By comparing versions 4 and 5 one notices that the paper’s information is a noisy input. On one hand v5 performs well in case that the top-1 similar contribution has by accident a similar paper’s title as the query itself. On the other hand it prefers sibling contributions from the same paper. In the case of contribution R6466, v5

responds as v1 with contribution R25719, which comes from the same paper (recall the hierarchy problem described above).

For version 6 the results have also shown that discarding the long literals makes the performance even worse in comparison with version 5 meaning that long literals are generally informative and should still be included. Query R6101 is a good example of how both v5 and v6 seek contributions with similar paper’s title and research field, but v6 ignores the long literals and wins in this case. V5 and v6 respond with R6140 and R6070, respectively.

Another point is handling the hyphen in combination with customizing the Elasticsearch’s tokenizer and filtering the stop words. Those ideas do not add any further modification in the results.

The last method applied is having indices per research field which is implemented in version 8. The results show a slight difference in comparison with v4, which could be due the limited test set size, and with a potential data growth in the **ORKG** it might perform better than v4. But in fact, it restricts the search space by excluding contributions assigned to another research field than the query’s, which can be in some cases unreasonable, since users are human beings and could incorrectly classify one paper to a another similar research field. For instance, papers on “Covid-19” could be classified to “Immunology of Infectious Disease”, “Biochemistry” or even to “Virology” research fields.

In addition to the analysis and evaluation, this work measures how confident is each version by its returned results, with the help of the similarity percentage derived from the **BM25** score. The average similarity percentage for a version is given by **Equation 11** and the results shown in **Figure 4.10** points out that v4 outperforms its competitors.

$$AvgSimilarity = \frac{1}{|Q|} \sum_{q \in Q} \frac{\sum_{d \in D} SimilarityPercentage(d, q)}{|D|} \quad (11)$$

where  $Q$  is the set of queries,  $D$  is the set of retrieved results and  $SimilarityPercentage(d, q)$  is defined in equation 4.

## 4.6 Conclusion

Evaluating **IR** systems quantitatively is deemed an important step to concretely determine system performance. To this end, one of the contributions of this thesis chapter is constructing an evaluation corpus as a human-annotated gold-standard to

## 4.6. Conclusion

---

Query \ Version	v0	v1	v4
R6402	R9469	R9469	R6320
R112273	R31051	R112367	R112384
R6466	R25719	R25719	R6458
R41020	R41025	R41019	R41019

Table 4.4: Revisiting the response analysis. Comparison between v0, v1 and v4 w.r.t. the top-1 result.

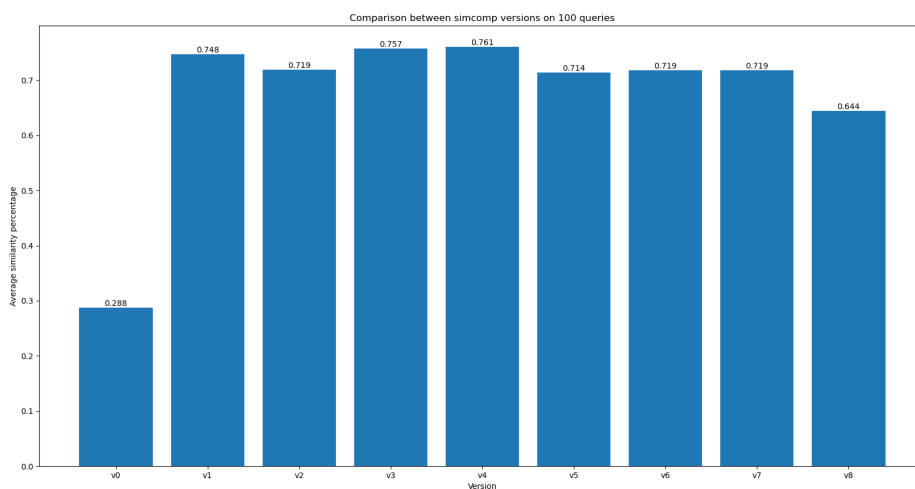


Figure 4.10: Comparison between simcomp versions w.r.t. average similarity percentage.

evaluate the **ORKG** contribution similarity service **Simcomp**. The corpus was created after insights were gleaned on the various characteristics of the **ORKG** contribution data. To this end the corpus was compartmentalized in four parts: 1) a homogeneous evaluation set - consisted of query contributions that had other contributions very similar to it and therefore were equally matched similar to the query; 2) a heterogeneous evaluation set - consisted of query contributions which clearly another winner similar contribution response standing out from other candidates; 3) a “with response” set - consisted of query contributions having similarities in the response of the default **Simcomp** service; and 4) a “without response” set - consisted of the



query contributions not having similarities in the response of the default **Simcomp**; Where sets 1 and 2 overlap with sets 3 and 4 having 100 test instances in total.

A detailed service response analysis was performed on the default **Simcomp** service. Its working was analyzed and technical shortcomings were noted. Based on the observations made, various enhancements were implemented to **Simcomp**. In all, eight different versions were created that included one or more combinations of the enhancements. E.g., a new DocumentCreator module to represent the contributions was implemented, the NLP preprocessing steps were tailored to better suit the **ORKG** labels, etc.

Finally, detailed evaluations were conducted. In this step, the evaluation corpus described at the beginning of the thesis chapter was leveraged. With it, the default **Simcomp** service could be evaluated w.r.t. the various enhancements made in this thesis. At least three different evaluation metrics were leveraged, viz. an approximate evaluation measure, **MRR**, and **MAP** each suited to the specific evaluation scenario. Furthermore, a qualitative analysis has been applied on a different small set to ensure the performance after applying the different implementation versions. The evaluations' results have shown that version 4 outperforms by having a **YAML**-structured documents its competitors numerically and qualitatively as well. It seeks for similar contributions in the entire **ORKG**'s dataset and tries to conserve the same document structure of the query. Additionally, by considering the **BM25** scoring function as similarity measure, the scores of the average similarity percentage show that version v4 is the most confident implementation by its retrieved results in comparison with the others.

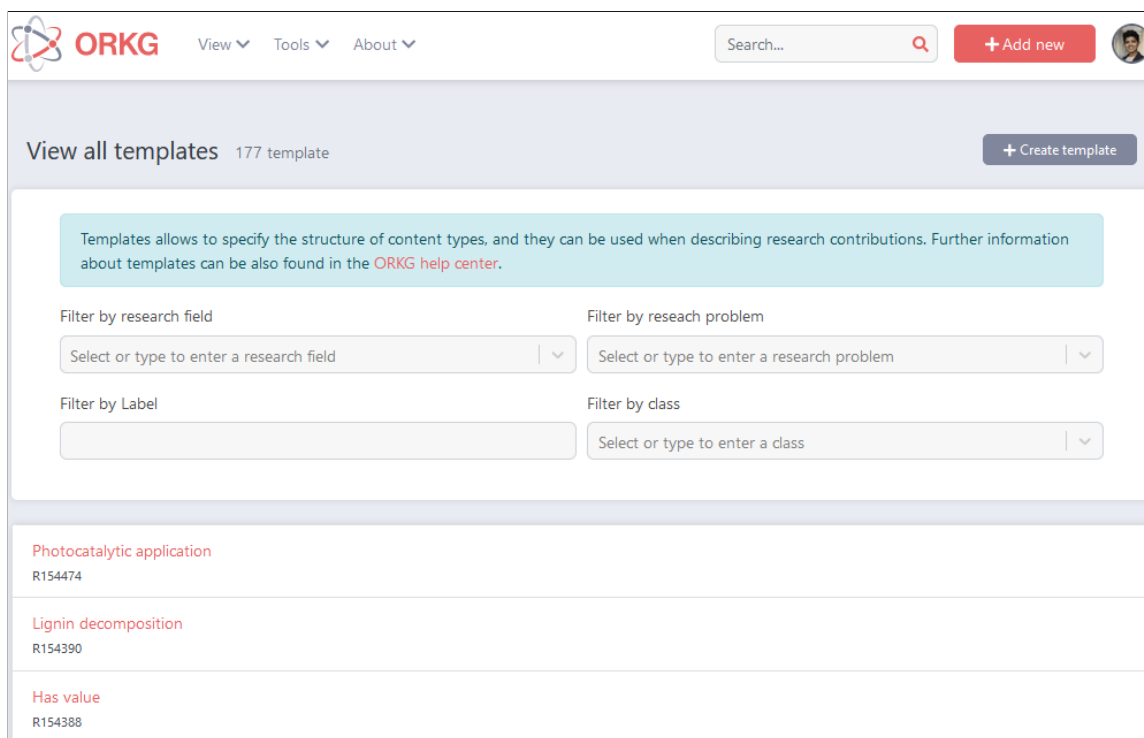
# Chapter 5

## Templates Recommendation

The next information retrieval service of the Open Research Knowledge Graph (ORKG) implemented in this thesis is the Templates Recommendation service. As described earlier in [chapter 3](#), templates are form-based semantic graph patterns that enable authors to structure their contribution data in a standardized manner. They are implemented based on repeated contribution pattern observations made across papers often focusing on a single research problem. E.g., *Basic reproduction number estimate* template <https://www.orkg.org/orkg/template/R40006>. As such the Templates Recommendation Service tackles the following information retrieval scenario. For a new incoming paper, based on the paper title and abstract content, it recommends a template to structured the paper’s contributions if one exists in the ORKG templates knowledge base.

As briefly introduced in [chapter 3](#), a template is a structured object representing domain-specific entities, their properties and relationships. Generally, a template represents something in the real world, and specifically, in the context of the ORKG, it represents a research contribution for a research problem. It may or may not relate directly to a specific span of text in an article. For example, the Covid 19 case fatality rate estimates reported as numbers in the Abstract of a paper as the paper’s main findings could lead to a Case Fatality Rate template (e.g., <https://www.orkg.org/orkg/template/R46273>) being created. Templates must be defined by a domain expert who fully understands the problem scope and therefore has a wholesome conceptual mental image of the properties it entails from direct exposure to the research study. Other users then either with growing or no awareness of a research problem are cautioned away from creating templates since they may not be able to specify all the problem properties and their chances of specifying incorrect or an incomplete set of problem properties are higher.

To create a new Template or to view the existing templates, a user must visit the <https://www.orkg.org/orkg/templates> page on the **ORKG Front-end**. See **Figure 5.1**. On this page, they can search among existing templates. Four filtering functions are offered: 1) Filter by research field - a template can be optionally assigned to a specific **ORKG** research field, thus, this feature when used lists all templates if they exist from particular user-specified research field(s); 2) Filter by research problem - a template can also be optionally assigned to a specific research problem, thus, akin to the earlier filtering function, by enlisting this feature templates assigned the same research problem can be listed; 3) Filter by label - allows filtering based on substring matches of the template name; and 4) Filter by class - allows filtering based on the specified class name. Next, to create a new template, the **ORKG Front-end** offers a user-friendly interface to specify a template: 1) *Description* (**Figure 5.2**); 2) *Properties* (**Figures 5.3** and **5.4**); and 3) *Format*.



The screenshot shows the ORKG website's 'Templates' page. At the top, there is a navigation bar with the ORKG logo, 'View', 'Tools', and 'About' dropdown menus, a search bar, and a '+ Add new' button. Below the navigation bar, the page title is 'View all templates' with a count of '177 template' and a '+ Create template' button. A light blue informational box states: 'Templates allows to specify the structure of content types, and they can be used when describing research contributions. Further information about templates can be also found in the ORKG help center.' Below this, there are four filter sections: 'Filter by research field', 'Filter by research problem', 'Filter by Label', and 'Filter by class'. Each filter section has a text input field and a dropdown arrow. The 'Filter by research field' and 'Filter by research problem' fields contain the placeholder text 'Select or type to enter a research field' and 'Select or type to enter a research problem' respectively. The 'Filter by Label' field is empty. The 'Filter by class' field contains the placeholder text 'Select or type to enter a class'. Below the filters, there is a list of templates with the following entries: 'Photocatalytic application' (R154474), 'Lignin decomposition' (R154390), and 'Has value' (R154388).

Figure 5.1: The **ORKG** Templates page with functionality to create a new template (“Create template” button) or search for existing templates with the option of four Filters to narrow search results. The four available search filtering methods are by 1) *research field*, 2) *research problem*, 3) *label*, and 4) *class*.

The screenshot shows the ORKG web interface for creating a new template. At the top, there is a navigation bar with the ORKG logo, links for 'View', 'Tools', and 'About', a search bar, and a '+ Add new' button. Below this is a header for 'Create new template' with a 'Save' button. The main content area is titled 'Create template' and has three tabs: 'Description' (selected), 'Properties', and 'Format'. Under the 'Description' tab, there are several form fields: 'Name of template' (a text input), 'Target class' (a dropdown menu with the placeholder 'Select or type to enter a class'), and a section titled 'Template use cases' which contains three optional fields: 'Property' (a dropdown menu with the placeholder 'Select or type to enter a property'), 'Research fields' (a dropdown menu with the placeholder 'Select or type to enter a research field'), and 'Research problems' (a dropdown menu with the placeholder 'Select or type to enter a research problem'). Each optional field has a small explanatory text below it.

Figure 5.2: The ORKG Templates Creation page, “Description” tab. On this page, users must specify “Name of template” and “Target class” field values. The “Property”, “Research fields”, and “Research problems” fields are optional. Specifying them indexes the template per the specified property which systematizes template search and in the backend generates a hierarchy of templates.

Having offered a background of ORKG templates, this chapter describes the information retrieval service called Templates Recommendation engineered into the ORKG codebase. As a first step to building a full-fledged recommendation service, the thesis conducts a feasibility analysis on the current ORKG data snapshot to

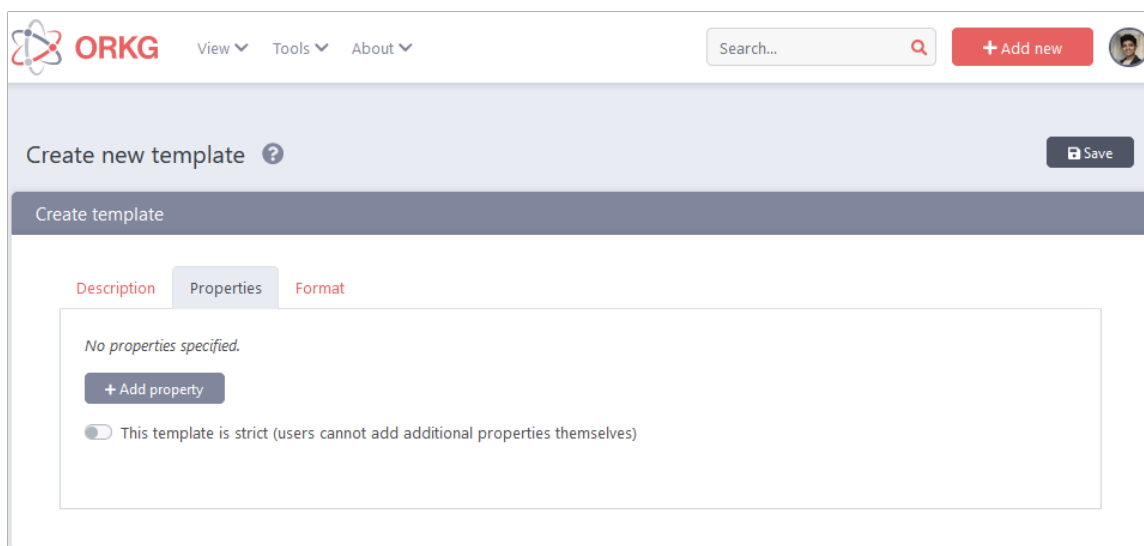


Figure 5.3: The ORKG Templates Creation page, “Properties” tab. On this page, users can add new properties via that “Add property” button. An example instance is show in Figure 5.4.

ensure that the service can be based on sound empirical parameter tuning methods and will report reliable evaluation results. The next section reports how the proposed Templates Recommendation Service was tested for its implementation feasibility.

## 5.1 Service Feasibility Analysis

Following the standard practice in empirical machine learning, the task is first defined as a dataset. This dataset would then serve an *information retrieval* objective. Based on whether the information retrieval objective is defined as an unsupervised (e.g., Elasticsearch) or a supervised (e.g., neural transformer models) method, the dataset can be used only to test a model (relevant in both unsupervised and supervised settings) or to train a model as well (relevant only in supervised setting).

First and foremost, a feasibility analysis was needed to check for two conditions: 1) the number of used templates in the ORKG, and 2) whether a sizeable proportion of the the ORKG papers were structured using the templates. If both conditions were satisfied, a diverse dataset representative of Science with adequate number of papers could be built for an information retrieval system. The feasibility analysis described next will discuss the characteristics of the dataset suited for service development.

Figure 5.4: The ORKG Templates Creation page, “Properties” tab. An example instance of adding a new Property called “Algorithm.” For each property, a *type* as an ORKG class or a predefined ontology class (e.g., SKOS <https://www.w3.org/TR/2008/WD-skos-reference-20080829/skos.html>, BFO <https://basic-formal-ontology.org/> ontologies) can be specified. The list of ontologies are supplied by TIB’s Ontology Lookup Service (OLS) <https://service.tib.eu/ts4tib/index>. The *cardinality constraints* for the minimum and maximum occurrences of a property within the contribution template can also be custom defined.

### 5.1.1 Task Dataset

First, a dataset must be compiled. To this end, a technical challenge was encountered. There is no relation between a paper  $p$  and the template  $t$  used by  $p$ , therefore, whether a paper was structured using a template had to be determined indirectly by means of heuristics. For this, a corpus pre-processing script was written to perform data matching from the ORKG’s RDF-Dump<sup>1</sup>. The script found used templates as

<sup>1</sup>Dump version: 13.08.2021.

follows. Create a dataset of contributions by retrieving contributions' properties up to the 2nd depth level. Drop all papers not written in English or whose abstracts could not be retrieved. For each **ORKG** template  $t$ :

1. Retrieve templates  $t$ 's properties.
2. Compare  $t$  with all contributions in the pre-created dataset and consider a contribution  $c$  as templated, if the 1st **or** the 2nd depth level of the contribution graph fulfills one of the following conditions:
  - (a) If all  $t$ 's properties are of optional cardinality, at least 3 properties are used by  $c$ .
  - (b) If set  $A$  of  $t$ 's properties is mandatory,  $A$  has to be used by  $c$  **and**  $|A| \geq 2$ .
3. Drop templates used by more than 100 contributions. With this all generic templates that apply in almost all **ORKG** contributions are dropped as candidates for automated probabilistic recommendation and instead are incorporated in deterministic rules for template suggestions. E.g., the generic 'Research Problem' template which holds true in almost all **ORKG** contributions.
4. Finally, drop templates that are used by less than 2 papers as they would not be good prediction candidates since they are not represent by a sizeable number of papers.

Note that thresholds used in the script were defined empirically. The threshold of 2 defined in step 4 ensures that at least 1 paper will be included in training and test partitions, respectively, within supervised empirical evaluations. The papers' abstracts were fetched with the help of secondary external service APIs, viz. *Crossref*<sup>1</sup> and *Semantic Scholar*<sup>2</sup> using the papers' DOI and/or their titles. This will be used subsequently as the query string to the Templates Recommendation Service. This was implemented in a service wrapper called *MetaService* that is responsible for fetching a paper's abstract from the mentioned external services. The abstract language was determined with the help of the pre-trained text classifier *fastText*<sup>3</sup> [37]. **Table 5.1** roughly shows the distribution of the papers over the templates. A total of 264 template structured papers could be identified across 23 different templates. A minimum of 2 papers and a maximum of 56 papers were structured with a single template. A detailed explanation of this dataset is discussed subsequently.

---

<sup>1</sup><https://www.crossref.org/>

<sup>2</sup><https://www.semanticscholar.org/>

<sup>3</sup><https://github.com/facebookresearch/fastText>

Template's ID	Template's Title	# Papers
R138668	Psychiatric Disorders AI Overview	56
R35087	Time interval	33
R76795	Crowd Intelligence	28
R70247	BioAssay	28
R40006	Basic reproduction number estimate	20
R77101	Standard's template	20
R48214	Global Mean Sea Level Rise Projections	13
R76209	Team Subtask Contribution Description	10
R108239	KM-Oriented Enterprise Modeling Approaches	9
R107684	Health Persuasion	8
R38248	Task Dataset Metric Score	8
R38147	Comparisons with Existing Machine Learning (ML) Models	5
R108008	Business architecture	4
R46273	Case fatality rate estimate	4
R74758	Experiment	2
R69198	Scientific Concept Extraction Dataset	2
R54009	Climate sensitivity	2
R46269	Case fatality rate estimate value specification	2
R108555	Machine Learning Experiment	2
R54875	Climate response	2
R49194	Occupant's perception and behaviour	2
R51438	Common Template	2
R48000	Problem	2
Total	-	264

Table 5.1: Distribution of templated papers over 23 templates.

### 5.1.2 Dataset Analysis

Further analysis was performed in order to gain more insights about the dataset distribution over the templates' research fields, the papers, and templates and papers combined. Figures 5.5 and 5.6 show the independent distribution of templates and papers over their labeled research fields, respectively. Figure 5.7, on the other hand, depicts how papers are distributed over templates' research fields and vice versa. Note that a template can have zero, one, or more research fields, which can be



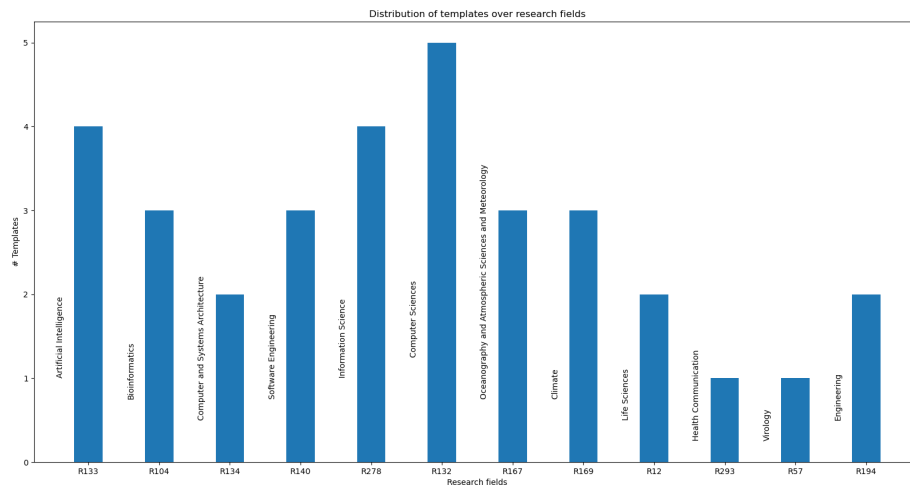


Figure 5.5: Distribution of templates over research fields.

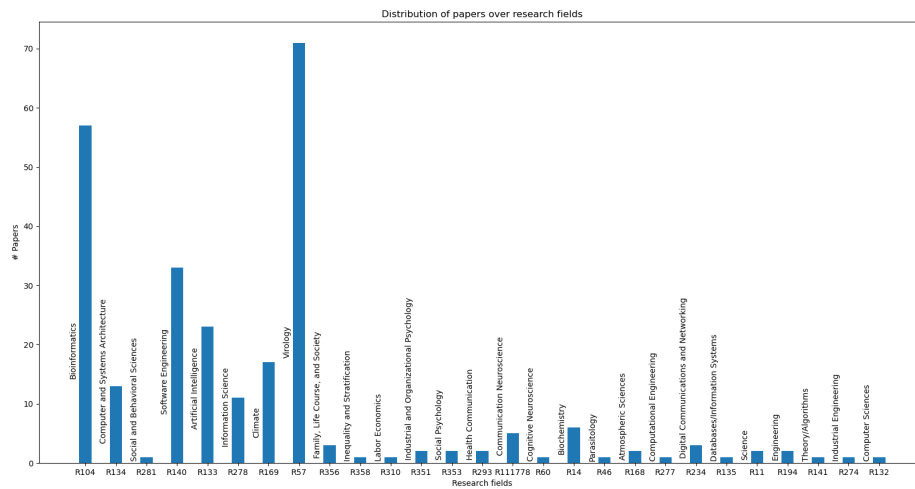


Figure 5.6: Distribution of papers over research fields.

identified in the raw **ORKG** data by tracing the predicate `TemplateOfResearchField`<sup>1</sup>.

<sup>1</sup><https://www.orkg.org/orkg/property/TemplateOfResearchField>

## 5.1. Service Feasibility Analysis

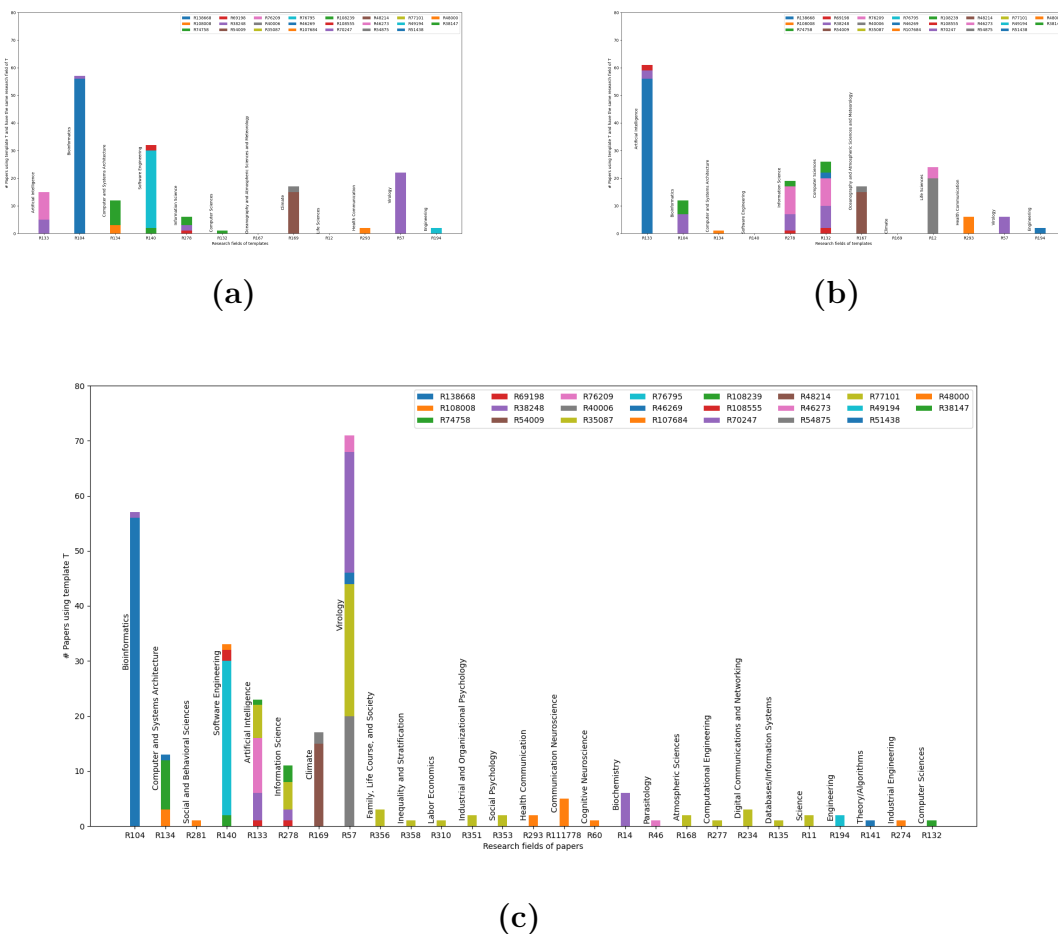


Figure 5.7: (a) Distribution of papers that belong to the same template's research field over the templates' research fields. (b) Complementary of (a): papers that do not belong to the same research template's research field (c) Distribution of papers using templates over the papers' research fields.

Further, the **ORKG** data needed to be realistically and comprehensively represented in our empirical evaluations. This means that not just templated papers but also papers untemplated ones had to be included. Thus, 264 new untemplated papers were, in addition, randomly selected from the **ORKG** data dump. The neutral papers were retrieved from the same research fields as the templated papers to ensure a similar thematic representation, however as a counter untemplated paper example. Additionally, a few papers were selected from other research fields that did not have

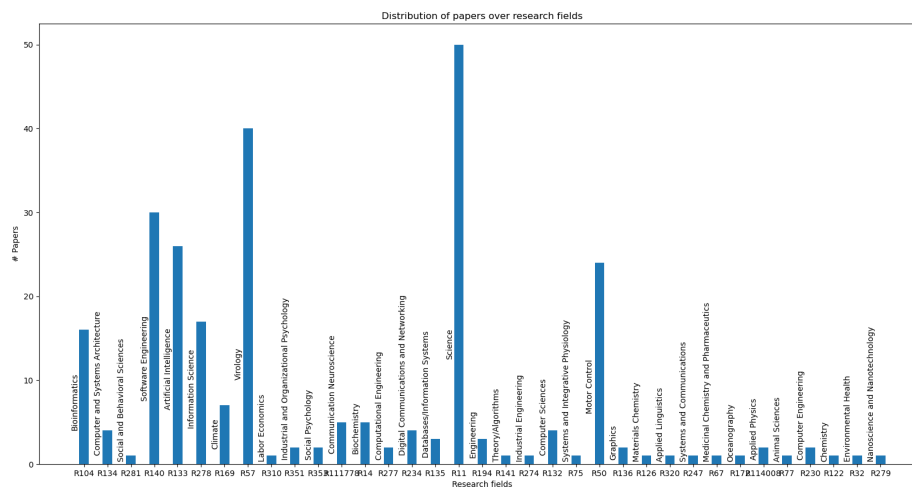


Figure 5.8: Distribution of neutral papers over research fields.

a templated counterparts. Figure 5.8 shows the neutral papers distribution.

Finally, given 264 templated papers over 23 templates and 264 untemplated papers overall across 43 unique research fields gave us a sizeable enough corpus over which to test and train an IR service. Thus our conclusion was an affirmative feasibility indication for IR service development.

## 5.2 Service Implementation

In this work, two approaches were proposed to recommend a template given a paper in the ORKG, namely, retrieving relevant templates by using Elasticsearch or inferring the correct template using Natural Language Inference (NLI) based on SciBERT [13] embeddings. There are a few important information items relevant around IR services. One of the foremost ones being *a query*. A query, the phrase posed to the IR service that the service is designed to satisfy. For the Templates Recommendation Service, a query is the title and abstract of a paper combined and the service itself should return the best semantically matched template for structuring the paper’s contribution. Next, pertaining to the service implementation, technical details of the corresponding service implementation are provided and the method for creating the dataset folds or partitioning for service development is described.

### 5.2.1 Unsupervised Approach with Elasticsearch

**Ranking Function.** For the unsupervised approach, Elasticsearch is used to retrieve the most relevant templates given a query. The implementation is engineered in a similar fashion to the Contributions Similarity Service presented in [chapter 4, Section 4.1](#). As a quick recap, Elasticsearch ranks responses w.r.t. a query by leveraging the [BM25](#) relevance scoring function which is a term-frequency based metric.

**Service Setup.** The implemented service behaves as a wrapper to the builtin Elasticsearch engine. In contrast to the workflow shown in [Figure 4.2](#), the DocumentCreator module is replaced with a MetaService for fetching the paper’s abstract; The input contribution ID is replaced with a paper title and its corresponding DOI; The similar contributions in the output are replaced with recommended templates. A relevant question here is how does the Elasticsearch index for Template Recommendation differ than that for similar Contributions retrieval? Technically, each document in the index is a concatenation of template  $t$  and the paper  $p$  structured by  $t$  with the document ID “ $ID(t);ID(p)$ ” (e.g. “R77101;R74026”) in case of a templated paper, and only the paper in case of a neutral paper (not used by any template) with the document ID “ $ID(p)$ ” (e.g. “R74026”). In inference time, the service recommends those templates that are included in Elasticsearch’s retrieved results and having the document ID format “ $ID(t);ID(p)$ ”. Saying papers are queries indeed implies that a paper represented by its title and abstract text, whereas in the terminology of templates, a template is represented by its title and properties’ labels. The described workflow is illustrated in [Figure 5.9](#) and it mostly corresponds to the workflow discussed earlier in [chapter 4](#).

**Building the Index.** Since we are currently not able to retrieve the templated papers automatically because of the missing relation between a paper and a template in the [ORKG](#), we have to build the index using the dataset we created. For new incoming templated papers we shall consider indexing them together with their templates. For now and as long as the relation is absent, the templated papers can be retrieved based on the heuristics and the index can only be built and updated manually. Building the index with 400 papers takes almost 30 seconds.

### 5.2.2 Supervised Approach with SciBERT

**Machine Learner.** Our machine learning system is the state-of-the-art, bidirectional transformer-based SciBERT [\[13\]](#) that is a variant of BERT [\[22\]](#) pre-trained on

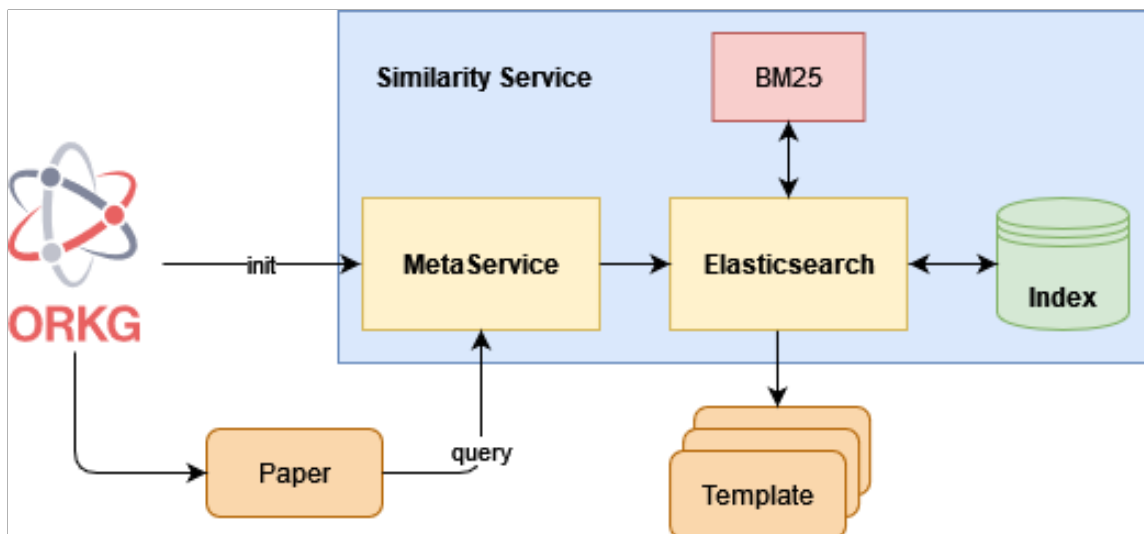


Figure 5.9: Unsupervised Approach - Templates Recommendation Service operational workflow. Arrows represent the data flow.

millions of scientific articles. We use the SciBERT inference architecture. For this, each data instance  $(t, p; c)$  corresponds to a paired template  $t$  with a paper  $p$  and the pair is assigned an inference class  $c \in \{entailment, neutral, contradiction\}$ . Each  $(t, p)$  are treated as sentence pairs separated by the special [SEP] token; the special classification token ([CLS]) remains the first token of every instance. Its final hidden state is used as the aggregate sequence representation for inference and is fed into a linear classification layer. Therewith, we define an **NLI** (*Natural Language Inference*) problem, where each  $(t, p)$  is the (*premise, hypothesis*) tuple and the outcome is defined as one of the three classes in  $c$ .

**Service Setup.** The implemented service behaves as a wrapper for the trained classifier. We introduce the service workflow by means of Figure 5.10. The service wrapper expects a paper represented by its title and optionally its DOI, and produces a list of similar templates to the given paper. The first step in the wrapper is to fetch the paper’s abstract using the MetaService. After that,  $N + 1$  sequences are created and fed into the trained classifier for inference, where  $N$  represents the number of templates the classifier has been already trained on, i.e. each known template is concatenated with the given paper and fed together as a sequence into the classifier. The classifier predicts the most likely class  $c$ . The additional premise is the empty template that represents the neutral case. The service then filters the  $N + 1$  outputs

and only returns the entailed templates. The best expected result for a given paper is to be contradicted by the classifier with  $N$  templates and entailed with the respective template, if it is a relevant one, or to be predicted as a neutral paper with the empty template, if there is no template for the paper to be described with.

**Training the Service.** For training the classifier we use a TPU runtime environment on *Google Colab*<sup>1</sup> [14] and fine-tune the pre-trained AllenNLP SciBERT [13] uncased model<sup>2</sup> with the PyTorch [52] implementation. We train the model for 10 epochs with batch size of 16 and max sequence length of 512 tokens. As an optimizer we utilize Adam [47] with initial learning rate of  $2 \cdot 10^{-5}$  and  $\epsilon = 10^{-6}$ . We also apply a step decay learning rate scheduler [28]. The cross entropy loss function is applied as criterion. We also applied early stopping on the validation loss. The training stopped with the mentioned settings at the 4th epoch after around 3.5 minutes and was done with training and validation accuracies of 95% and 93%, and with training and validation losses of 13% and 15%, respectively. The corresponding diagrams are shown in Figure 5.11. The mentioned numbers produced the best model as we performed multiple trainings on different data splits described in the next section.

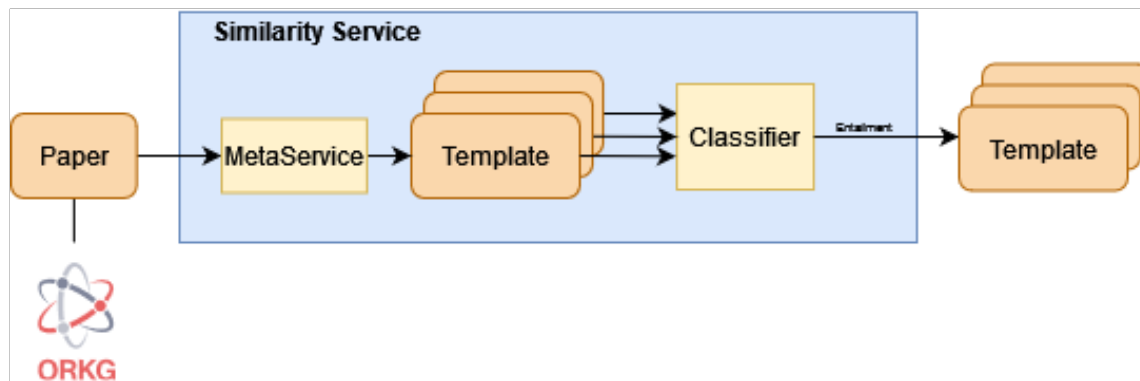


Figure 5.10: Supervised Approach - Templates Recommendation Service architecture. Arrows represent the data flow

<sup>1</sup><https://colab.research.google.com>

<sup>2</sup><https://github.com/allenai/scibert>

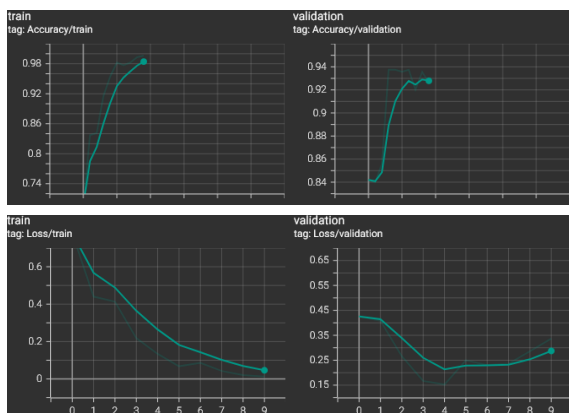


Figure 5.11: Supervised Approach - Corresponding training (left) and validation (right) accuracy (top) and loss (bottom) values over the training epochs.

## 5.3 Service Evaluation

### 5.3.1 Experimental Setup

In this subsection, the method for creating the training and test dataset partitions together with the evaluation metrics for the experiments are discussed.

**Training Dataset.** The training data partition consists of paired instances of templates and papers. This set combines randomly a total number of 23 valid templates and the empty template with 400 papers. For data instances representing templated papers (entailments), we have an average of 9 papers per template with a minimum of 1 and a maximum of 32 papers. We selected the training instances as follows:

1. A training instance is a text sequence tuple represented by
  - (a) a premise = template = template’s title and properties’ labels, and
  - (b) a hypothesis = paper = paper’s title and abstract.
2. We decide to select 75% of the entailments and 75% of the neutrals for the training portion of the dataset.  $(t, p)$  pairs were randomly selected from the dataset, so that all instances are unique and all templates were included. Note that  $t$  is the empty template for a neutral instance.
3. The training contradictions are synthesized by concatenating papers with templates borrowed from other structured papers. In order to preserve the classes

balance in the training set, we choose the following two thresholds for synthesizing the contradictions. Note that contradictions are not part of the Elastic-search index.

- (a) #Entailed contradictions for a template  $t = \lceil \#papers \text{ using } t * \frac{1}{ec} \rceil$ .
- (b) #Neutral contradictions for each neutral paper  $p = nc$ .

Where  $ec \in \{3, 4, 5, 6\}$  and  $nc \in \{2, 4, 5\}$  are different synthesizing sets of thresholds that were combined by applying the Cartesian Product [67]. Each combination produced in principle a different dataset used for model training.

4. For each class, we select 10% of the instances for the validation set. Note that there is no validation set splitting in case of the unsupervised approach.

**Test Dataset.** For creating the test set we select the remaining data instances as test entailments and neutrals. We ensure there is no test dataset leakage within the training dataset, in particular w.r.t. the contradiction training instances where templates are randomly paired with other papers that they do not structure. Thus note that none of the neutral or entailment papers in the test dataset have appeared in the training dataset. Further, and conversely, none of the papers used in the training dataset appear in the test data. The test dataset has 52 *entailments* and 64 *neutral* instances, respectively.

The corresponding split dataset statistics are shown in [Table 5.2](#).

**Evaluation Metrics.** Each approach was evaluated with the standard accuracy, precision, recall and f-measure metrics, which are defined in equations 12, 13, 14 and 15, respectively. Consider that this differs from the **MRR** and **MAP** metrics used in [chapter 4](#). In this setting, note that each paper is assigned only one true template. Therefore, the accuracy and f-measure metrics were better suited. Note that for extended evaluations, the **MRR** metric can also be applied. For clarity, we offer our definitions for TP (True Positives), FP (False Positives), TN (True Negatives) and FN (False Negatives) used in the calculations.

- TP: total instances correctly classified as entailments or neutrals.
- FP: total instances incorrectly classified as entailments or neutrals.
- TN: total instances correctly classified as contradictions.
- FN: total instances incorrectly classified as contradictions.



$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (12)$$

$$Precision = \frac{TP}{TP + FP} \quad (13)$$

$$Recall = \frac{TP}{TP + FN} \quad (14)$$

$$F - measure = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (15)$$

-	Training (supervised)	Validation (supervised)	Training (unsupervised)	Test
Entailment	180	20	200	52
Neutral	180	20	200	64
Contradiction	736	84	0	0
Total	1096	124	400	116

Table 5.2: Templates Recommendation - Dataset splits using Entailed Contradictions  $ec = 0.4$  and Neutral Contradictions  $nc = 4$  thresholds for synthesizing the contradicted training instances.

Two further remaining questions must be clarified w.r.t. the experimental setup. *i)* How does one differentiate between contradictions and neutrals when the supervised service returns only entailments? And *ii)* how does the concept of contradiction instances apply in the experimental setup for the unsupervised approach, when Elasticsearch is not indexed for contradictions?

Answering both questions requires considering the evaluation from a user perspective, who interacts with a service, neither with a classifier nor with a term-based index and expects either an appropriate template or nothing. Recall that both REST API services respond with a list of recommended templates, which is the only material that we can rely on in our experiments and answers our first question. We elaborate the different cases of a service response with examples. *Case 1 - Nothing to recommend:* An empty list in the response indicates a neutral classification, which means the **ORKG** has no template that can be recommended for the given inquiry. This also means the service has implicitly classified all templates as contradictions (TN) and the empty template as neutral (TP). *Case 2 - Templates to recommend:* A

list with different templates in the response indicates an entailment classifications for those templates (could be TP or FP) and contradiction classifications for all other not returned templates (could be TN or FN). Having the expected template at rank 1 of the list results in 1 TP and  $l - 1$  FP, where  $l$  is the length of the returned list. To particularly answer the second question, which actually only affects computing the standard accuracy while it considers TN, we can assume that the unsupervised service also does filter the results and return only entailments. With this assumption we could calculate TN and consequently the standard accuracy exactly as we do for the supervised service. Equation 16 shows the exact definition of the standard accuracy in our terminology.

$$Accuracy = \frac{\sum_{q \in Q} N - l + k}{Q * N} \quad (16)$$

Where  $Q$  is the number of test queries,  $N$  is the number of templates,  $l$  is the length of the results list and  $k$  is a constant with value 1 if there is a hit at rank 1 of the response, otherwise 0. Note that  $k$ ,  $N - l$  and  $Q * N$  are equivalent to TP, FP and TP + TN + FP + FN shown in Equation 12, respectively.

To apply a fair comparison one last assumption is yet needed, since that the unsupervised service cannot control the number of returned results (recall the BM25 score in chapter 4), whereas the supervised service can, which gives it an advantage in terms of its interpretability. Basically we assume that both services return a result list with a maximum length of 1. Thus, the variable  $l$  in Equation 16 can be either 0 or 1 that implies an equivalent behavior in both services. Not applying the latter assumption would increase the FP in the unsupervised case and consequently penalize that approach.

### 5.3.2 Results and Discussion

The threshold combinations of  $ec = 0.4$  and  $nc = 4$  results in the optimal supervised system. This optimal system results juxtaposed with the unsupervised approach's results are shown in Figure 5.12. Further, a comparison of training/inference performance times are also shown in Table 5.3. A detailed evaluation of the supervised model selection can be found in Table 5.4. Comparing the supervised and unsupervised approaches, the supervised approach outperforms Elasticsearch in all measures with a slight difference in terms of accuracy and by 10% in terms of f-measure. Having an equal recall and precision in Elasticsearch's results means that the service incorrectly classifies contradictions as it incorrectly classifies neutrals or entailments. This relates again to the problem that Elasticsearch always returns results. This

issue does not happen by the supervised approach, since it can distinguish between the classes, which makes it easier to interpret.

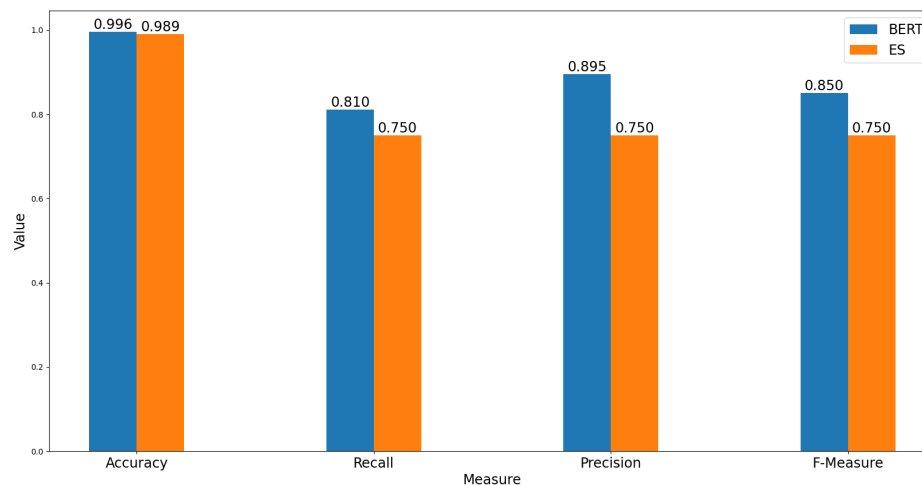


Figure 5.12: Evaluation comparison between supervised (blue) and unsupervised (orange) approaches using different evaluation metrics.

-	Supervised	Unsupervised
Training	8m (on TPU)	3.6s
Inference	32s	1.5s

Table 5.3: Comparison between the performance of the supervised and unsupervised approaches on a CPU. The abstract’s fetching time is not considered.

## 5.4 Conclusion

In this chapter, a novel Information Retrieval (IR) task is discussed in the context of the **ORKG** Digital Library. The IR service handles recommending the semantically relevant templates for structuring research contributions given a paper title and abstract as query for which the templating patterns exist. For the unrepresented semantic types, i.e. for those paper domains without a defined structuring

ec/en	Accuracy	Recall	Precision	F-measure
0.3/2	0.968	0.250	0.250	0.250
0.4/2	0.986	0.663	0.669	0.666
0.5/2	0.990	0.741	0.761	0.751
0.6/2	0.986	0.681	0.681	0.681
0.3/4	0.992	0.732	0.809	0.769
0.4/4	<b>0.996</b>	0.810	<b>0.895</b>	<b>0.850</b>
0.5/4	0.984	0.568	0.605	0.586
0.6/4	0.988	<b>0.836</b>	0.757	0.795
0.3/5	0.986	0.655	0.672	0.663
0.4/5	0.993	0.793	0.844	0.817
0.5/5	0.995	0.818	0.879	0.848
0.6/5	0.991	0.767	0.787	0.777

Table 5.4: Supervised Approach - Model selection w.r.t. entailed contradictions (ec) and neutral contradictions (nc).

template, the service is expected to return no templates as a result. Several conceptual and definitional considerations had to be made to put in place such a service implementation.

The first consideration was a detailed analysis of the **ORKG** data dump to identify the number of templated papers and based on size it was determined that service developments and experimental evaluations were feasible. For the service itself, i.e. to recommend an existing **ORKG** template to a user at the point of adding a new paper, two different approaches were experimented within this thesis. An unsupervised approach which utilized Elasticsearch and a supervised approach based on SciBERT [13], the state-of-the-art transformer-based machine learning system. The second consideration was conceptualization of a suitable evaluation task dataset. This involved introducing “entailment”, “neutral” and “contradiction” instances. Neutrals were a specific class introduced that covered all papers not structured by an **ORKG** template. Finally, our evaluation dataset consisted of 528 papers structured by 23 templates and distributed over 41 research fields. The third consideration was service development itself which entailed building a new Elasticsearch index for untemplated papers and training an **NLI** model. The training instances of the supervised classifier was the same as those used to build the Elasticsearch index. The test corpus is also shared for both the proposed approaches and we ensured that its instances were unseen during the learning step. Furthermore, we defined an **NLI**

problem with a classifier built upon the pre-trained SciBERT model. To this end, we synthesized the training data for the supervised approach to cover all classes related to the **NLI** problem. Finally, for each of the approaches a **Back-end** service was implemented, described and evaluated on accuracy, precision, recall and f-measure. We showed that the supervised approach outperforms the unsupervised one in all measures and fails in respect of timing performance perspectives.

In all, despite the interpretability advantage of the supervised approach, we believe that Elasticsearch is simpler (the index can be dynamically updated) and more efficient (recall **Table 5.3**) to be maintained and integrated in the current **ORKG** assuming the existence of a relation between a paper and its template. As future work, we suggest the same proposed classifier to be trained and evaluated on a larger corpus of templated papers, since that as the **ORKG** evolving, more semantic interpretations might be required in order to recommend a more accurate result.

# Chapter 6

## ORKG Grouped Predicates Recommendation

### 6.1 Introduction

Scholarly Knowledge Graphs (**SKGs**) represent the contents of research literature as subject-predicate-object triples [8], thereby enabling comprehensive analyses that identify e.g. benchmark performance rankings with machine-interpretable system scores triples in automatically computed Leaderboards <https://www.orkg.org/orkg/benchmarks>. The Open Research Knowledge Graph (**ORKG**) with its specific focus on capturing *scholarly contributions* [9] falls within the wide **SKG** construction context [1, 42, 48] that include various other aims such as capturing bibliographic metadata or research data as structured knowledge. In an **SKG**, generally, a research knowledge aspect is formalized as (**S, P, O**) triples, where pairs of entities are related to each other by predicates [26]. By integrating triples from a variety of sources, knowledge graphs can be used to perform computational analysis on the comprehensive body of knowledge. E.g., **SKG** of biomedical knowledge contributions enable tracking disease-disease or disease-drug relations [18].

This chapter discusses an **IR** service over the semantic construct of predicates in **ORKG** triples. This semantic construct was visited already in **chapter 5**, as the building blocks of the aggregated graph patterns of templates. But isolated as individual items within the formalism of triples in **KG**, “a predicate is seen as a property that a subject has or is characterized by.”<sup>1</sup> E.g., every **ORKG** contribution resource as a subject has the property “has research problem” as a predicate. The

---

<sup>1</sup>[https://en.wikipedia.org/wiki/Predicate\\_\(grammar\)](https://en.wikipedia.org/wiki/Predicate_(grammar))

object of this predicate is then specific instances of one or more research problems tackled in the contribution.

COVID Bioassays

A comparison of key semantic properties and their values of Covid-related Bioassays.

April 2020

Properties	Design and Synthesis of Dipeptidyl Glutaminy Fluoromethyl Ketones as Potent Severe Acute Respiratory Syndrome Coronavirus (SARS-CoV) Inhibitors 2006 - Contribution 1	Design, synthesis and crystallographic analysis of nitrile-based broad-spectrum peptidomimetic inhibitors for coronavirus 3C-like proteases 2013 - Contribution 1	Specific Plant Terpenoids and Lignoids Possess Potent Antiviral Activities against Severe Acute Respiratory Syndrome Coronavirus 2007 - Contribution 1
Deposit Date	2010-05-24	2013-11-07	2010-05-25
Has	BAO semantic description	BAO semantic description	BAO semantic description
Has assay format	Vero CCL-81 cells and human colon carcinoma cell lines CaCo2 cell-based format toxicity format	enzyme format in vitro format protein format	Vero E6 cells cell-based format cytopathogenic format
Has assay method	cell viability evaluation	fluorescence resonance energy transfer	cytopathic morphology evaluation
Has assay phase characteristic	minimum Essential Eagle medium supplemented with 2% FBS	homogeneous phase	Dulbecco's modified Eagle's medium (DMEM) supplemented with 10% fetal bovine serum (FBS)
Has assay title	Inhibitory activity against SARS coronavirus FFMI replication in vero (CCL81) cells	Inhibition of SARS coronavirus 3C-like protease pretreated for 5 mins before substrate addition by fluorimetry	Inhibition of SARS coronavirus-induced cytopathogenicity in Vero E6 cells at 20 uM
Has concentration unit	micromolar	micromolar	micromolar
Has confirmatory assay	confirmatory assay	confirmatory assay	literature-derived assay
Has detection instrument	MMT cell-proliferative Kit 1 (Roche)	EnVision 2101 Multilabel Plate Reader	
Has detection method	MTT	fluorescence changes fitted to a single-exponential decay	inverted phase contrast microscopy
Has endpoint	EC50	IC50	EC50

Figure 6.1: The comparison feature of the ORKG as a survey of Bioassays

The question we ask in this chapter is how can an ORKG user benefit from the vast store of predicates information when structuring their paper for those predicates that are not part of any predefined templates? This is a necessary service to prevent users from reframing their contributions from scratch in the scenarios where this can be avoided. Essentially, the ORKG relies on *the wisdom of the crowd* [68] w.r.t. the collective intelligence that arises when imperfect judgments are aggregated. Specifically, w.r.t. the ORKG this would mean tapping into collective intelligence where systematic patterns were observed when structuring paper contributions. A natural question here is how can these systematic patterns be observed other than within templates? Recall the *Comparisons* feature discussed in chapter 3. Within Comparisons, several research contributions sharing similar properties as aggregated as

a survey. E.g., as depicted in the [Figure 6.1](#) comparison survey over three different Bioassays based on their similar properties. This serves as a distinct example of aggregated crowd intelligence. While such comparisons are good candidates for the creation of templates, there is no strict one-to-one mapping between predicates in comparisons made into templates. In other words, templates are not necessarily defined from comparisons although they are worthwhile candidates for templates design. Furthermore, even the converse case does not hold, i.e. not all contributions structured with templates are part of comparisons. This is because a template can be used to structure just one contribution whereas the comparison feature necessarily requires at least two contributions to be applicable.

This thesis in its three contributing chapters has developed [IR](#) services involving the [ORKG predicates](#) in some capacity or another. For instance, in [chapter 4](#), similar contributions were suggested where an Elasticsearch engine indexed contributions and ranked them w.r.t. a query contribution. In [chapter 5](#), an Elasticsearch engine was indexed on predicate groups within templates minus the values of the predicates. The predicate groups were then ranked w.r.t. incoming queries as papers to be structured for their contributions. Finally, this concluding chapter, in a similar vein to the previous chapter, reconsiders predicate groups as an [IR](#) service, except the predicate groups were not strictly defined as valid semantic patterns by a semantic modeling domain expert. Instead the predicate groups arise from [ORKG](#) comparisons. The subsequent sections provide details of this service including the service feasibility analysis to final implementation.

## 6.2 Service Feasibility Analysis

As done in the previous chapter, the task is first defined as a dataset. This dataset would then serve an *information retrieval* objective. First and foremost, systematic observations are offered over the [ORKG RDF](#) triples dump<sup>1</sup> w.r.t. two main organizational characteristics of its *predicates*: 1) when treated as individual data items, and 2) when grouped within thematic constructs of comparisons in a similar aggregating vein to templates albeit informally. Note that for some parts of the data construction in the following subsections we use the same modules used in [chapter 5](#), namely the *MetaService* and *fastText*<sup>2</sup> [37].

---

<sup>1</sup>dated 10.11.2021.

<sup>2</sup><https://github.com/facebookresearch/fastText>



### 6.2.1 Dataset of Single Predicates

**Dataset Construction.** A list of all ORKG predicates was constructed. This list was prefiltered for predicates that applied in every contribution by default, e.g. “has research problem”, and for predicates that did not obviously satisfy the role of defining an ORKG contribution triple. E.g., predicates relating to the paper bibliographic metadata. Note that predicate paths were maintained as is and no path substrings were included in this group. For instance, one possible predicate in the contribution shown in Figure 6.2 is “has benchmark;has evaluation;has metric”. The paths were fetched by leveraging a similar implementation discussed in Section 4.4 and were filtered across different phases: *i*) Content-nonspecific predicates were manually selected and dropped e.g. “see also”, “description”; *ii*) Predicates with different IDs and similar labels serving the same semantical purpose were manually selected and merged e.g. “F-measure” and “F1”; *iii*) Non-English predicates and papers as well as those papers whose abstracts could not be retrieved were dropped. The resulting list had 2011 predicates. Some insights of this data are provided next.

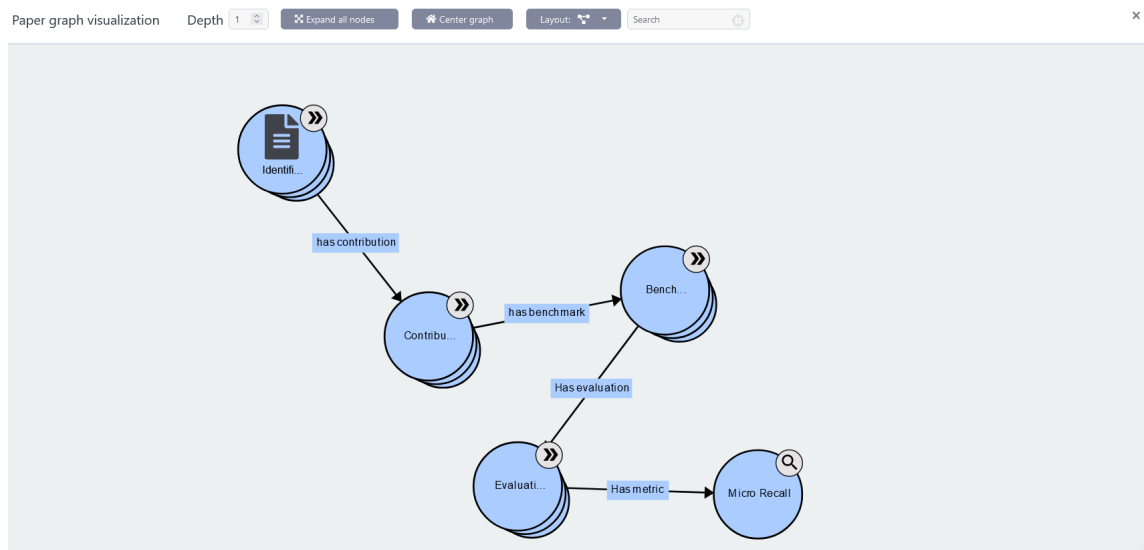


Figure 6.2: The ORKG Front-end graphical visualization for a scholarly contribution showing one path starting from the contribution node “Contribution” to the last resource node “Micro Recall”. All other paths are hidden.

**Dataset Analysis.** We provide visualizations of the predicates distributions over papers, research fields and research problems in figures 6.3, 6.4 and 6.5, respectively.

Table 6.1 provides insights of the distributions statistics. In total the data analysis shows that the 2011 retrieved predicates are diversely distributed over 140 and 752 research fields and problems, respectively. Having an average of 1.36 and 2.55 for research fields and problems per predicate indicates that the predicates are content-specific and cannot be generically used for structuring any paper. We deemed that a total of 5579 papers with 15.62 papers on average per predicate was a representative enough size of data that would suffice to train an automated learner.

-	Papers	Research Fields	Research Problems
Min/Predicate	2	1	0
Max/Predicate	931	29	57
Average/Predicate	15.62	1.36	2.55
Total	5579	140	752

Table 6.1: Dataset of Single Predicates - Statistics.

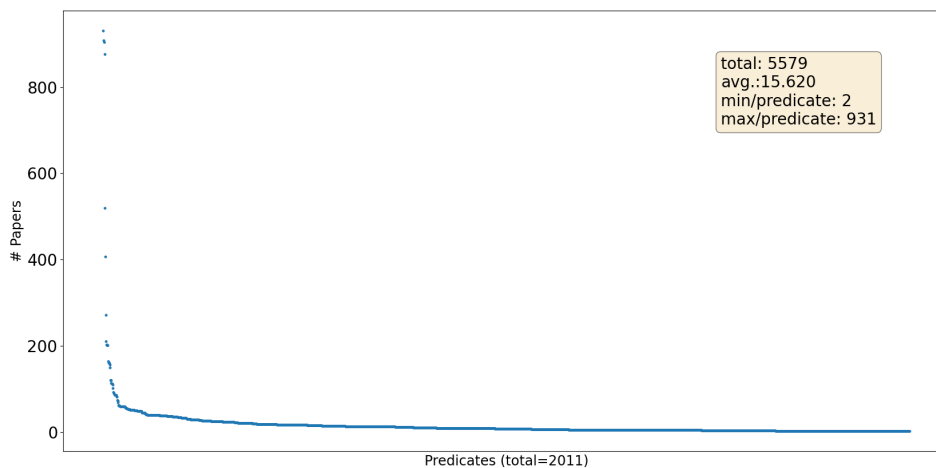


Figure 6.3: Dataset of Single Predicates - Predicates distribution over papers.

## 6.2.2 Dataset of Clustered Predicates

**Task Dataset Construction.** At the outset, we defined a criteria for predicates selection. This informed the dataset construction. First, we were interested in se-

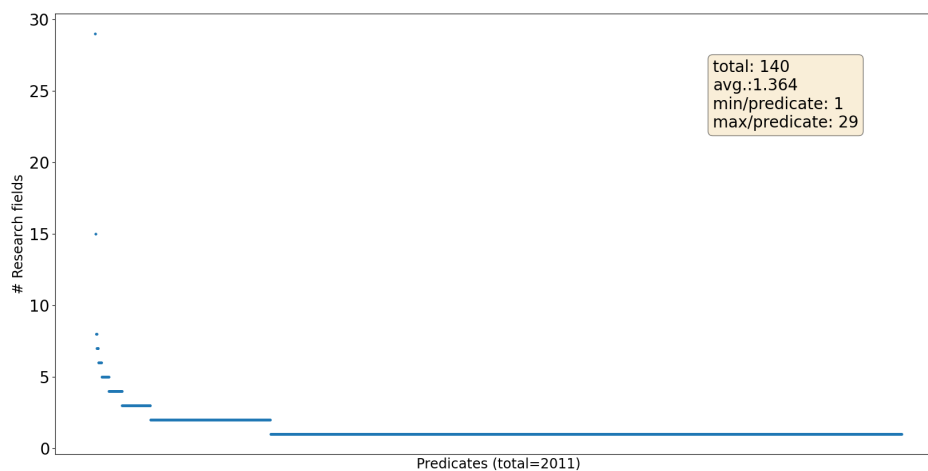


Figure 6.4: Dataset of Single Predicates - Predicates distribution over research fields.

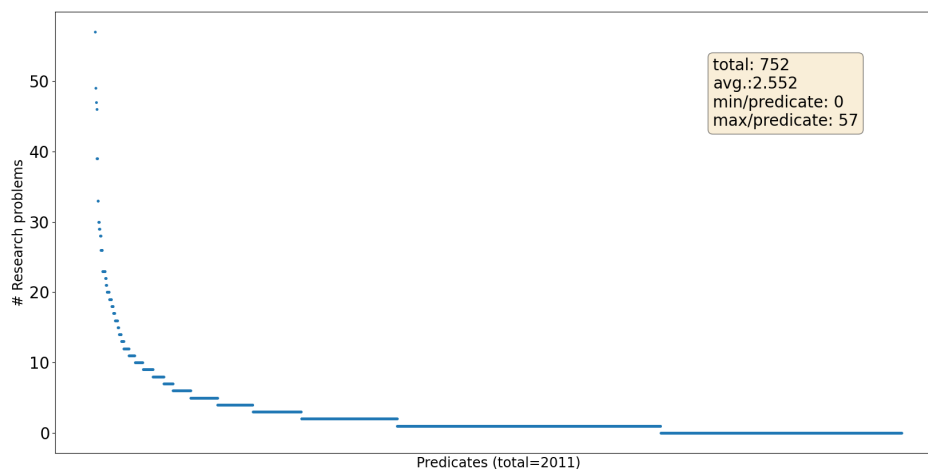


Figure 6.5: Dataset of Single Predicates - Predicates distribution over research problems.

mantic groups of predicates, i.e. predicates used to structure a paper. Second, the grouped predicates needed to show a repeated usage pattern to structure different

contributions at least a minimum number of times to be considered loosely as an evolving template pattern or as part of a valid semantic group. The **ORKG** comparisons provided a reference starting dataset to identify such predicate groups. All **ORKG** comparisons <https://www.orkg.org/orkg/comparisons> with 10 or more contributions were selected.<sup>1</sup> 10 was empirically selected as the threshold for the minimum number of contributions to be structured by a group of predicates for them to be considered as a valid semantic unit. Only those predicate groups were chosen for clustering that structured over 10 contributions. As in **Subsection 6.2.1**, the paths of the predicates were maintained. The paths were fetched by calling the scholarly contributions comparison service<sup>2</sup> provided by **Simcomp**, which compares based on the paths of the spanning tree of each contribution. For the data filtering we dropped those comparisons which have less than 2 papers and also dropped non-English and non-abstract papers as in the previous dataset. Next we provide some dataset analysis insights.

**Task Dataset Analysis.** In this part, we demonstrate the analysis of the dataset of clustered predicates in two respects, viz. data grouped by comparisons and data grouped by predicates. For each, we show the respective distribution over papers, research fields, and research problems in Figs. 6.6, 6.7 and 6.8, respectively. Furthermore, we show the distribution of comparisons over predicates and vice versa in **Figure 6.9**. Data statistics are summarized in Tables 6.2 and 6.3. The data analysis for the clustered predicates shows again a diverse distribution over different research fields and problems in total and a field-related distribution in average by considering both statistics groups. Thus comparisons can be seen as groups of clusters containing papers that share similar predicate candidates. 4060 total number of papers is for this dataset as well a proper number of data instances, on which a clustering algorithm can rely.

## 6.3 Service Implementation

The methodology will be inspired from related work on the Semantification of Biological Assays data in the **ORKG** [6]. A bioassay is, by definition, a standard biochemical

---

<sup>1</sup>While the following comparison <https://www.orkg.org/orkg/comparison/R142085> called “Smart city governance research categories analysis by references articles ver.2” has over 60 contributions, it is excluded from our data since its predicates were not meaningful standalone semantic relational data items. Same with <https://www.orkg.org/orkg/comparison/R156112>.

<sup>2</sup>[https://gitlab.com/TIBHannover/orkg/orkg-similarity/-/blob/master/comparison/compare\\_paths.py](https://gitlab.com/TIBHannover/orkg/orkg-similarity/-/blob/master/comparison/compare_paths.py)

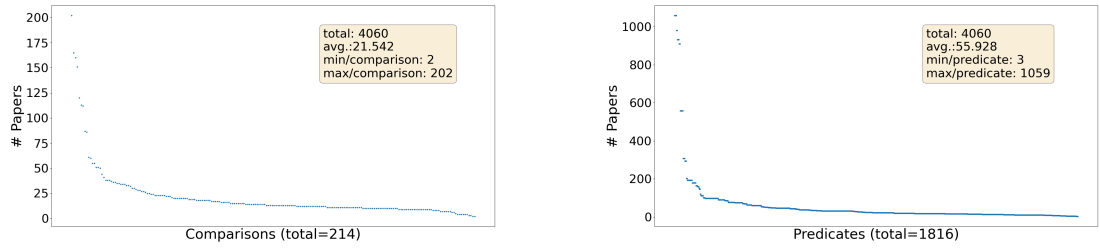


Figure 6.6: Dataset of Clustered Predicates - Comparisons distribution over papers (left) and predicates distribution over papers (right).

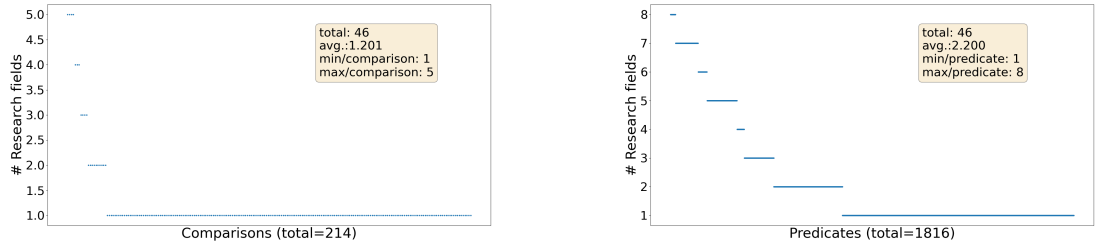


Figure 6.7: Dataset of Clustered Predicates - Comparisons distribution over research fields (left) and predicates distribution over research fields (right).

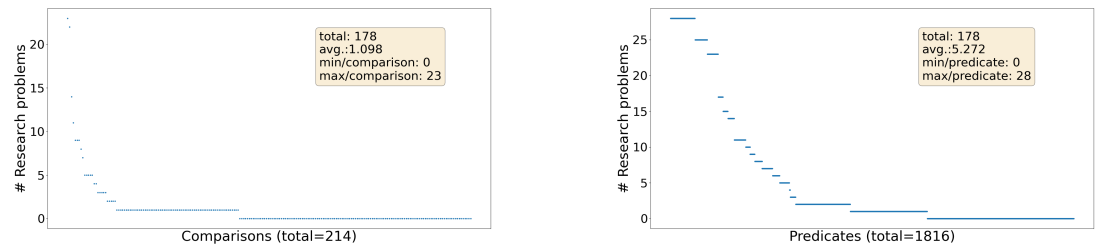


Figure 6.8: Dataset of Clustered Predicates - Comparisons distribution over research problems (left) and predicates distribution over research problems (right).

### 6.3. Service Implementation

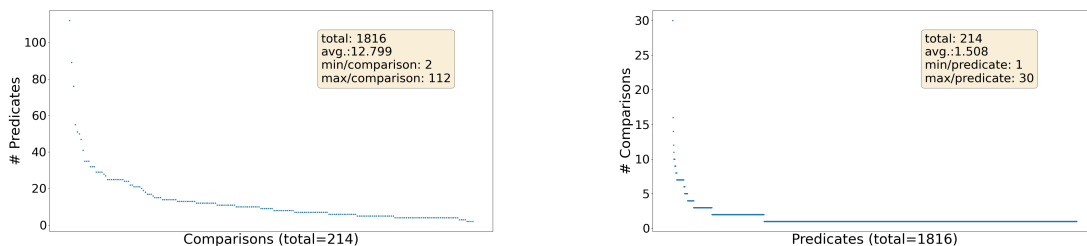


Figure 6.9: Dataset of Clustered Predicates - Comparisons distribution over predicates (left) and predicates distribution over comparisons (right).

-	Papers	Predicates	Research Fields	Research Problems
Min/Comparison	2	2	1	0
Max/Comparison	202	112	5	23
Average/Comparison	21.54	12.79	1.20	1.09
Total	4060	1816	46	178

Table 6.2: Dataset of Clustered Predicates - Statistics grouped by comparisons.

-	Papers	Comparisons	Research Fields	Research Problems
Min/Predicate	3	1	1	0
Max/Predicate	1059	30	8	28
Average/Predicate	55.92	1.50	2.20	5.27
Total	4060	214	46	178

Table 6.3: Dataset of Clustered Predicates - Statistics grouped by predicates.

test procedure used to determine the concentration or potency of a stimulus (physical, chemical, or biological) by its effect on living cells or tissues [32, 33]. Semantic description of assays are represented as logical annotations consisting of property and value pairs based on the BioAssay Ontology (BAO) [73, 2]. The set of the predefined property-value pairs annotated with the BAO as reference are indeed intended as the semantic equivalent of the unstructured bioassay text although this is not guaranteed without some information loss. Obtaining semantified bioassays enables their large-scale analysis in diverse systems since they are machine-interpretable. Similar in principle to the bioassays are then the **ORKG** structured contributions the semantic equivalents the information found in the unstructured text. Over bioassays, we

observed those with similar descriptions also had similar semantic representations. And semantification based on labels assigned to similar assays clusters showed performances above 80% in F-score [6]. This score showed that K-means over vectorized representations of the assays was effective in producing homogeneous clusters. Thus the idea of semantification based on clustering works in practice and remains to be tested on a different underlying dataset, e.g., a dataset of scholarly contributions which is the aim of this chapter.

### 6.3.1 K-means Clustering of Contribution Vectors

#### Scholarly Contribution Vectorization Functions.

- a. TF-IDF embeddings - We use the scikit-learn [53, 16] implementation<sup>1</sup> to convert the data corpus into TF-IDF [65] vectors depending on the text corpus. TF-IDF vectors are  $n$ -dimensional real-valued vectors representing a given text with the TF-IDF value for each possible term from the corpus, where  $n$  is the number of unique terms contained in the corpus.
- b. SciBERT embeddings - We feed forward the pre-trained AllenNLP SciBERT [13] uncased model<sup>2</sup> with the text corpus to output its final hidden state, which is then averaged by the tokens of each text sentence resulting in a vectorized text representation.

**Clustering Function.** In this thesis chapter we leverage the centroid-based clustering algorithm K-Means [35] to group equivalent scholarly contributions represented by their papers. K-Means is known as an effective and simple clustering algorithm with a space complexity of  $O(N(D + K))$  and time complexity of  $O(NDK)$ , where  $N$  is the number of data instances,  $D$  is the number of features (dimensions) and  $K$  is the number of centers or clusters. One limitation in K-Means is that  $K$  is expected as input to the algorithm, so that the algorithm randomly initialize the clusters' centers in its first step. In the second step, K-Means updates the points assignment to the clusters depending on a distance function and keeps re-updating the clusters until no more changes occur in the clusters or a predefined number of iterations is reached. The output of the algorithm is the data points grouped by clusters. For building the clusters we use in this thesis chapter the K-Means implementation<sup>3</sup>

---

<sup>1</sup>[https://scikit-learn.org/stable/modules/feature\\_extraction.html#tfidf-term-weighting](https://scikit-learn.org/stable/modules/feature_extraction.html#tfidf-term-weighting)

<sup>2</sup><https://github.com/allenai/scibert>

<sup>3</sup><https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

provided by scikit-learn [53, 16] and train it on vectorized papers represented by their titles and abstracts.

**Service Setup.** The idea of the predicates recommendation service is pretty simple. Since we have grouped similar papers into clusters, we can simply recommend a user those predicates used by all papers in the predicted cluster for an new incoming paper. Furthermore, by taking advantage of the dataset of clustered predicates discussed in Subsection 6.2.2 we already apply a predicates grouping by comparisons step, which allows us to recommend the predicates used by the comparisons included in the predicted cluster instead of the pure papers. The workflow of the service is illustrated in Figure 6.10. First, the service fetches the paper’s abstract using the MetaService as in the previous chapter. Then, the paper’s title and abstract are concatenated and vectorized using a vectorization function. The vector representation is then fed into the pre-trained clustering model, which outputs the predicted cluster and its assigned papers. Finally, we map the predicted papers to their comparisons and output the predicates used in the comparisons to the service caller.

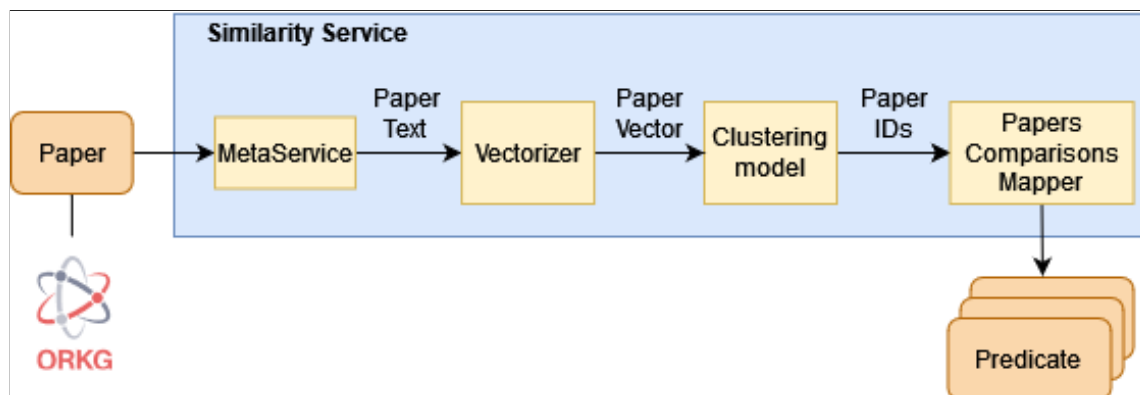


Figure 6.10: Grouped Predicates Recommendation Service architecture. Arrows indicate the data flow.

## 6.4 Service Evaluation

### 6.4.1 Experimental Setup

**Training and Test Datasets.** From each comparison group with more than 10 contributions, 70% of its papers were selected as the training dataset to build the



clustering model and the remaining 30% of its papers were reserved as test dataset to evaluate the predicate assignments. In other words the test dataset enabled evaluating whether homogeneous groups of papers could be generally obtained. In total our training set consists of 2857 papers distributed over 214 comparisons, whereas our test set consists of 1203 papers distributed over 180 comparisons. We ensured the uniqueness in the test papers, so that no test paper occurs in the training set; hence the leakage in the test comparisons.

**Evaluation Metrics.** The standard precision, recall and f-measure were computed given in equations 13, 14 and 15, respectively. The metrics are macro- as well as micro-averaged [7].

**Selecting K.** For selecting the best clustering model we rely on the results of the evaluation metrics after applying them on the several trained models. Since we already have our data grouped by comparisons and our training data consists of 2857 instances, so should  $k$  be in the range  $|C| \leq k \leq |P|$ , where  $C$  is the set of comparisons and  $P$  is the set of training papers. Considering the complex time and space limitations<sup>1</sup> of K-Means [35] we train the models across the range from 200 to 2050 with a step size of 50 resulting in 38 models for each of the vectorization methods.

**Average Purity.** In our cluster analysis we custom define the term *purity* as the percentage of clusters that group papers only from the same comparison they were distributed from. For instance, the purity function of comparison *comp* distributed over a set of clusters  $C$  is defined in Equation 17 as:

$$Purity(comp) = \frac{|\{c \in C \mid c \text{ groups papers only from } comp\}|}{|C|} \quad (17)$$

and the average purity is the sum of comparisons' purities normalized by the number of comparisons.

---

<sup>1</sup>Due to the necessity of increased RAM sizes during the learning step of K-Means we subscribed to Google Colab Pro+ <https://colab.research.google.com/signup> that offers a CPU environment with ~52GB RAM size.

## 6.4.2 Results and Discussion

### Quantitative Analysis

By applying the mentioned metrics on the different trained models with both vectorization methods we get the results shown in [Table 6.4](#). The evaluation results clearly point out that TF-IDF outperform the SciBERT vectorization method in all metrics and over all models. Comparing the models trained on TF-IDF vectors results into the models  $k = 1650$  and  $k = 1850$  with 0.706 macro-averaged F-measure and 0.665 micro-averaged F-measure as best models, respectively. While the macro-average is used to know how the system performs in overall, the micro-average weighs each test instance equally, i.e. it counts for accurate prediction on predicates level instead of on comparisons level. The models were further analyzed and the results are summarized in [Table 6.5](#) that shows a higher average purity value 64.5% for the model  $k = 1850$ . Thus, we select the latter model as the best performing one and compare its time complexity with a SciBERT model having the same number of clusters in [Table 6.6](#). Note that one vector representation has a length of 260016 and 768 for TF-IDF and SciBERT, respectively.

### Qualitative Error Analysis

More investigations are yet needed to indicate the reasons behind the noisy clusters, which can be done by observing the purity measure per comparison. A fact is, that comparisons partially share same papers. Once we consider this fact in computing the average purity by dropping duplicated papers from different comparisons we increase the value to 84.6% and to 78.4% for both models with  $k = 1850$  and  $k = 1650$ , respectively (recall [Table 6.5](#)). Another fact is, that noisy clusters - which have lower purity value or i.e. are distributed from multiple comparisons - semantically overlap. For instance, the comparison R140131 with title “Smart cities and cultural heritage” and purity value of 25% has 4 papers that have been distributed over 4 clusters (1 pure + 3 impure). These clusters are distributed from 6 comparisons in total that compare research about “Smart Cities”. A detailed overview of these comparison IDs and their titles is stated in [Table 6.7](#). Thus, having impure comparisons does not necessarily imply inaccurate predictions, it just could relate to the initial assumption that the comparisons existing in the ORKG purely group relevant research papers without semantic overlapping, which has led to the stated quantitative evaluation results.

-	Macro-Average						Micro-Average					
-	TF-IDF			SciBERT			TF-IDF			SciBERT		
Num. of Clusters	P	R	F1	P	R	F1	P	R	F1	P	R	F1
200	0.344	<b>0.957</b>	0.445	0.128	<b>0.678</b>	0.187	0.057	<b>0.953</b>	0.108	0.055	<b>0.634</b>	0.101
250	0.384	0.917	0.478	0.148	0.675	0.213	0.075	0.926	0.138	0.066	0.628	0.119
300	0.433	0.906	0.530	0.171	0.652	0.241	0.144	0.896	0.248	0.079	0.613	0.139
350	0.453	0.917	0.557	0.183	0.655	0.249	0.272	0.906	0.419	0.080	0.623	0.142
400	0.472	0.902	0.570	0.184	0.652	0.254	0.237	0.909	0.377	0.090	0.616	0.158
450	0.509	0.913	0.603	0.203	0.648	0.273	0.285	0.920	0.435	0.097	0.605	0.167
500	0.566	0.909	0.662	0.203	0.647	0.273	0.418	0.901	0.571	0.099	0.599	0.170
550	0.541	0.901	0.630	0.213	0.639	0.282	0.345	0.897	0.498	0.102	0.592	0.175
600	0.557	0.899	0.645	0.236	0.634	0.306	0.375	0.904	0.531	0.117	0.583	0.195
650	0.567	0.890	0.651	0.236	0.635	0.307	0.382	0.894	0.536	0.118	0.574	0.196
700	0.572	0.882	0.653	0.256	0.622	0.325	0.379	0.885	0.531	0.133	0.584	0.217
750	0.585	0.879	0.663	0.257	0.630	0.324	0.388	0.875	0.538	0.133	0.579	0.216
800	0.598	0.882	0.678	0.265	0.619	0.331	0.428	0.872	0.574	0.135	0.570	0.218
850	0.605	0.884	0.683	0.276	0.621	0.342	0.424	0.886	0.573	0.143	0.572	0.229
900	0.592	0.887	0.672	0.283	0.618	0.348	0.410	0.873	0.558	0.148	0.565	0.235
950	0.608	0.870	0.683	0.293	0.623	0.359	0.423	0.864	0.568	0.148	0.580	0.236
1000	0.605	0.864	0.680	0.298	0.608	0.361	0.424	0.858	0.568	0.158	0.566	0.247
1050	0.624	0.868	0.695	0.307	0.615	0.370	0.428	0.860	0.572	0.166	0.566	0.257
1100	0.632	0.868	0.701	0.319	0.608	0.376	0.447	0.838	0.583	0.174	0.576	0.267
1150	0.628	0.856	0.695	0.315	0.607	0.375	0.445	0.828	0.579	0.171	0.571	0.263
1200	0.618	0.859	0.685	0.327	0.605	0.383	0.408	0.842	0.550	0.173	0.561	0.265
1250	0.613	0.833	0.677	0.329	0.588	0.385	0.433	0.799	0.562	0.189	0.539	0.280
1300	0.630	0.832	0.695	0.339	0.599	0.394	0.507	0.813	0.625	0.191	0.558	0.285
1350	0.607	0.826	0.668	0.347	0.590	0.400	0.406	0.803	0.539	0.197	0.550	0.290
1400	0.595	0.827	0.661	0.353	0.593	0.404	0.400	0.790	0.531	0.201	0.551	0.294
1450	0.599	0.827	0.663	0.357	0.588	0.408	0.410	0.793	0.541	0.207	0.549	0.301
1500	0.594	0.820	0.657	0.368	0.583	0.414	0.397	0.808	0.532	0.206	0.535	0.298
1550	0.639	0.813	0.694	0.374	0.580	0.421	0.511	0.757	0.610	0.229	0.532	0.320
1600	0.576	0.794	0.636	0.385	0.584	0.433	0.376	0.757	0.503	0.237	0.538	0.329
1650	<b>0.650</b>	0.821	<b>0.706</b>	0.382	0.583	0.428	0.535	0.799	0.641	0.230	0.545	0.324
1700	0.622	0.791	0.677	0.397	0.584	0.439	0.514	0.769	0.616	0.250	0.536	0.341
1750	0.627	0.794	0.680	0.409	0.585	0.452	0.489	0.770	0.598	0.267	0.545	0.358
1800	0.594	0.773	0.652	0.403	0.572	0.443	0.509	0.738	0.603	0.256	0.529	0.345
1850	0.649	0.779	0.694	0.417	0.585	0.458	<b>0.593</b>	0.732	<b>0.655</b>	0.272	0.548	0.364
1900	0.632	0.773	0.679	0.419	0.576	0.457	0.564	0.723	0.634	0.275	0.534	0.363
1950	0.628	0.763	0.675	0.428	0.571	0.465	0.584	0.713	0.642	0.290	0.527	0.375
2000	0.624	0.750	0.666	0.439	0.574	0.475	0.552	0.699	0.617	0.292	0.525	0.376
2050	0.609	0.748	0.654	<b>0.447</b>	0.572	<b>0.479</b>	0.486	0.696	0.572	<b>0.319</b>	0.534	<b>0.399</b>

Table 6.4: Evaluation results of predicates clustering.

## 6.5 Conclusion

In this chapter, another **IR** task in the context of the **ORKG** is discussed. The task aims to retrieve semantically similar **ORKG**'s predicates to a user inquiry represented by a new incoming paper to help the users while reframing their contributions from scratch when possible. In contrary to the **IR** service tackled in **chapter 3**, this service's

-	k=1650	k=1850
Min Papers/Cluster	1	1
Max Papers/Cluster	30	24
Average Papers/Cluster	1.901	1.696
Min Clusters/Comparison	1	1
Max Clusters/Comparison	106	110
Average Clusters/Comparison	10.004	10.700
Min Comparisons/Cluster	1	1
Max Comparisons/Cluster	6	7
Average Comparisons/Cluster	1.297	1.237
Average Purity	0.590	0.645

Table 6.5: Statistics for clustering models with 1650 and 1850 clusters.

-	TF-IDF	SciBERT
Training	886.14s	1876.83s
Inference	5.57s	0.12s

Table 6.6: Comparison between the performance of the clustering using TF-IDF and SciBERT vectorization methods with  $k = 1850$ . The vectorization time is not considered.

ID	Title
R140131	Smart cities and cultural heritage
R141752	Smart city governance research categories analysis by references articles
R141782	Smart city governance research categories analysis by references articles
R146458	Enterprise architecture applications for managing digital transformation of smart cities
R145950	Ontology reuse in smart city ontologies
R27061	Overview of taxonomy of smart city definitions

Table 6.7: List of sample comparisons that overlap in clusters.

main database is the predicates stored in the **ORKG** as components of its (S, P, O) triples as such and not the aggregated ones that form templates.

To this end, systematic observations over the **ORKG** data had to be conducted to gain insights into the characteristics of the existing predicates and namely, *i*) when treated as individual data items, and *ii*) when grouped within comparisons as aggre-

gating components. For this we constructed and analyzed two datasets: the single predicates dataset in [Subsection 6.2.1](#) with 5579 papers and 2011 predicates; And the clustered predicates dataset in [Subsection 6.2.2](#) containing 4060 papers, 1816 predicates and 214 comparisons. In the process we ensured a diverse distribution over different research fields and problems.

The methodology of the implemented **IR** service was inspired from related work on the Semantification of Biological Assays data in the **ORKG** [6], which defined the task as a clustering problem. In this chapter we leveraged the dataset of clustered predicates grouped within comparisons to build clusters of comparisons' papers that can be mapped into comparisons' predicates at inference time (see [Figure 6.10](#) for the service workflow) The dataset were divided in 2857 and 1203 training and test papers, respectively. We applied TF-IDF [65] as well as SciBERT [13] embeddings to represent a paper's text represented by its title and abstract and then fed them to the centroid-based clustering algorithm K-Means [35] to build the clusters.

For selecting the best number of clusters we trained different models across the range from 200 to 2050 with a step size of 50 for both vectorization methods, resulting in 76 different models, and applied the evaluation metrics, precision, recall and f-measure macro- and micro-averaged, on them. The results clearly show that the models trained on TF-IDF vectors outperform the SciBERT ones. The results also show that  $k = 1850$  is the best number of clusters with F1-score of 0.655.

Considering the drawbacks of K-Means as clustering algorithm such as, e.g. centroid-based and a spherical-data distribution assumption, as well as the imbalance existing in our training set's labels represented by comparisons, a better performance can be achieved in future works by applying a better suited clustering algorithms, such as hierarchical-based ones like the *Agglomerative Clustering* [78], where papers of comparisons can be hierarchically distributed over clusters. Another way would be considering a better suited algorithm to deal with the high dimensionality existed in the TF-IDF vectors, as a vector representation in our training set has a length of 260016.

# Bibliography

- [1] SciGraph. <https://www.springernature.com/de/researchers/scigraph>. Accessed: 2021-11-02.
- [2] S. Abeyruwan, U. D. Vempati, H. Küçük-McGinty, U. Visser, A. Koleti, A. Mir, K. Sakurai, C. Chung, J. A. Bittker, P. A. Clemons, et al. Evolving bioassay ontology (bao): modularization, integration and applications. In *Journal of biomedical semantics*, volume 5, pages 1–22. Springer, 2014.
- [3] R. B. Allen. Supporting structured browsing for full-text scientific research reports. *arXiv preprint arXiv:1209.0036*, 2012.
- [4] R. B. Allen. Rich semantic models and knowledgebases for highly-structured scientific communication. *arXiv preprint arXiv:1708.08423*, 2017.
- [5] W. Ammar, D. Groeneveld, C. Bhagavatula, I. Beltagy, M. Crawford, D. Downey, J. Dunkelberger, A. Elgohary, S. Feldman, V. Ha, et al. Construction of the literature graph in semantic scholar. In *NAACL-HLT (3)*, 2018.
- [6] M. Anteghini, J. D’Souza, V. A. P. M. dos Santos, and S. Auer. Easy semantification of bioassays, 2021. URL <https://arxiv.org/abs/2111.15182>.
- [7] V. V. Asch. Macro-and micro-averaged evaluation measures [ [ basic draft ] ]. 2013.
- [8] S. Auer. Towards an open research knowledge graph, Jan. 2018. URL <https://doi.org/10.5281/zenodo.1157185>.
- [9] S. Auer, A. Oelen, M. Haris, M. Stocker, J. D’Souza, K. E. Farfar, L. Vogt, M. Prinz, V. Wiens, and M. Y. Jaradeh. Improving access to scientific literature with knowledge graphs. *Bibliothek Forschung und Praxis*, 44(3):516–529, 2020. doi: doi:10.1515/bfp-2020-2042. URL <https://doi.org/10.1515/bfp-2020-2042>.
- [10] K. Balog, L. Azzopardi, and M. Rijke. Formal models for expert finding in enterprise corpora. pages 43–50, 01 2006. doi: 10.1145/1148170.1148181.
- [11] K. Balog, L. Azzopardi, and M. Rijke. A language modeling framework for expert finding. *Information Processing & Management*, pages 1–19, 01 2009. doi: 10.1016/j.ipm.2008.06.003.

- 
- [12] S. Bechhofer, D. De Roure, M. Gamble, C. Goble, and I. Buchan. Research objects: Towards exchange and reuse of digital knowledge. *Nature Precedings*, pages 1–1, 2010.
- [13] I. Beltagy, K. Lo, and A. Cohan. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3606–3611, 2019.
- [14] E. Bisong. *Google Colaboratory*, pages 59–64. Apress, Berkeley, CA, 2019. ISBN 978-1-4842-4470-8. doi: 10.1007/978-1-4842-4470-8\_7. URL [https://doi.org/10.1007/978-1-4842-4470-8\\_7](https://doi.org/10.1007/978-1-4842-4470-8_7).
- [15] B. Brodaric, F. Reitsma, and Y. Qiang. Skiing with dolce: toward an e-science knowledge infrastructure. In *FOIS*, pages 208–219, 2008.
- [16] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.
- [17] J. Carroll and G. Klyne. Resource description framework (RDF): Concepts and abstract syntax. W3C recommendation, W3C, Feb. 2004. <https://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>.
- [18] H. Chen, L. Ding, Z. Wu, T. Yu, L. Dhanapalan, and J. Y. Chen. Semantic web for integrated network analysis in biomedicine. *Briefings in Bioinformatics*, 10(2):177–192, 03 2009. ISSN 1467-5463. doi: 10.1093/bib/bbp002.
- [19] N. Craswell. *Mean Reciprocal Rank*, pages 1703–1703. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9\_488. URL [https://doi.org/10.1007/978-0-387-39940-9\\_488](https://doi.org/10.1007/978-0-387-39940-9_488).
- [20] J. Dalton, L. Dietz, and J. Allan. Entity query feature expansion using knowledge base links. 07 2014. doi: 10.1145/2600428.2609628.
- [21] A. De Waard, L. Breure, J. G. Kircz, and H. Van Oostendorp. Modeling rhetoric in scientific publications. In *International Conference on Multidisciplinary Information Sciences and Technologies, InSciT2006*, 2006.
- [22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.
- [23] P. Donohoe, J. Sherman, and A. Mistry. The long road to jats. In *Journal Article Tag Suite Conference (JATS-Con) Proceedings 2015 [Internet]*. National Center for Biotechnology Information (US), 2015.

- [24] J. D'Souza, A. Hoppe, A. Brack, M. Y. Jaradeh, S. Auer, and R. 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*, pages 2192–2203, 2020.
- [25] F. Ensan and E. Bagheri. Document retrieval model through semantic linking. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017.
- [26] F. Manola and E. Miller. W3C.org. <https://www.w3.org/TR/rdf-concepts/#dfn-rdf-triple>. Online; accessed 30 November 2021.
- [27] S. Fathalla, S. Vahdati, S. Auer, and C. Lange. Towards a knowledge graph representing research findings by semantifying survey articles. In *International Conference on Theory and Practice of Digital Libraries*, pages 315–327. Springer, 2017.
- [28] R. Ge, S. M. Kakade, R. Kidambi, and P. Netrapalli. The step decay schedule: A near optimal, geometrically decaying learning rate procedure for least squares, 2019.
- [29] P. Groth, A. Gibson, and J. Velterop. The anatomy of a nanopublication. *Information Services & Use*, 30(1-2):51–56, 2010.
- [30] C. V. Gysel, M. de Rijke, and E. Kanoulas. Neural vector spaces for unsupervised information retrieval. *ACM Transactions on Information Systems*, 36(4):1–25, Oct 2018. ISSN 1558-2868. doi: 10.1145/3196826. URL <http://dx.doi.org/10.1145/3196826>.
- [31] A. Hars. Designing scientific knowledge infrastructures: the contribution of epistemology. *Information Systems Frontiers*, 3(1):63–73, 2001.
- [32] W. M. Hoskins and R. Craig. Uses of bioassay in entomology. *Annual review of entomology*, 7(1):437–464, 1962.
- [33] J. Irwin. Statistical method in biological assay. *Nature*, 172(4386):925–926, 1953.
- [34] M. Y. Jaradeh, A. Oelen, K. E. Farfar, M. Prinz, J. D'Souza, G. Kismihók, M. Stocker, and S. 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*, page 243–246, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450370080. doi: 10.1145/3360901.3364435. URL <https://doi.org/10.1145/3360901.3364435>.
- [35] X. Jin and J. Han. *K-Means Clustering*, pages 695–697. Springer US, Boston, MA, 2017. ISBN 978-1-4899-7687-1. doi: 10.1007/978-1-4899-7687-1\_431. URL [https://doi.org/10.1007/978-1-4899-7687-1\\_431](https://doi.org/10.1007/978-1-4899-7687-1_431).
- [36] A. E. Jinha. Article 50 million: an estimate of the number of scholarly articles in existence. *Learned Publishing*, 23(3):258–263, 2010.



- 
- [37] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April 2017.
- [38] J. M. Joyce. *Kullback-Leibler Divergence*, pages 720–722. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-04898-2. doi: 10.1007/978-3-642-04898-2\_327. URL [https://doi.org/10.1007/978-3-642-04898-2\\_327](https://doi.org/10.1007/978-3-642-04898-2_327).
- [39] J. Kim, X. Xue, and W. B. Croft. A probabilistic retrieval model for semistructured data. In M. Boughanem, C. Berrut, J. Mothe, and C. Soule-Dupuy, editors, *Advances in Information Retrieval*, pages 228–239, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-00958-7.
- [40] E. Landhuis. Scientific literature: information overload. *Nature*, 535(7612):457–458, 2016.
- [41] Y. Levy and T. J. Ellis. A systems approach to conduct an effective literature review in support of information systems research. *Informing Science*, 9, 2006.
- [42] N. Lewis, J. Wang, M. Poblet, and A. Aryani. Research graph: Connecting researchers, research data, publications and grants using the graph technology. In *eResearch Australasia Conference*, 2016. URL [https://eresearchau.files.wordpress.com/2016/03/eresau2016\\_paper\\_95.pdf](https://eresearchau.files.wordpress.com/2016/03/eresau2016_paper_95.pdf).
- [43] E. Lie. How many words make a sentence?, Feb. 2016. URL [https://techcomm.nz/Story?Action=View&Story\\_id=106](https://techcomm.nz/Story?Action=View&Story_id=106).
- [44] L. LIU and M. T. ÖZSU, editors. *Mean Average Precision*, pages 1703–1703. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9\_3032. URL [https://doi.org/10.1007/978-0-387-39940-9\\_3032](https://doi.org/10.1007/978-0-387-39940-9_3032).
- [45] X. Liu and H. Fang. Latent entity space: a novel retrieval approach for entity-bearing queries. *Information Retrieval Journal*, 18:473–503, 2015.
- [46] Z. Liu, C. Xiong, M. Sun, and Z. Liu. Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval, 2018.
- [47] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019.
- [48] P. Manghi, N. Manola, W. Horstmann, and D. Peters. An infrastructure for managing ec funded research output: The openaire project. *Grey Journal (TGJ)*, 6(1), 2010.
- [49] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, USA, 2008. ISBN 0521865719.
- [50] V. G. Meister. Towards a knowledge graph for a research group with focus on qualitative analysis of scholarly papers. In *SemSci@ ISWC*, pages 71–76, 2017.

- [51] A. Oelen, M. Y. Jaradeh, M. Stocker, and S. Auer. Generate FAIR literature surveys with scholarly knowledge graphs. *CoRR*, abs/2006.01747, 2020. URL <https://arxiv.org/abs/2006.01747>.
- [52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>.
- [53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [54] J. R. Pérez-Agüera, J. Arroyo, J. Greenberg, J. Iglesias, and V. Fresno. Using bm25f for semantic search. In *SEMSEARCH '10*, 2010.
- [55] S. Peroni. The semantic publishing and referencing ontologies. In *Semantic web technologies and legal scholarly publishing*, pages 121–193. Springer, 2014.
- [56] D. Petkova and B. Croft. Hierarchical language models for expert finding in enterprise corpora. page 8, New Jersey, USA, 2006. IEEE. ISBN 9780769527284.
- [57] J. Pokorny. Nosql databases: A step to database scalability in web environment. In *Proceedings of the 13th International Conference on Information Integration and Web-Based Applications and Services*, iiWAS '11, page 278–283, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450307840. doi: 10.1145/2095536.2095583. URL <https://doi.org/10.1145/2095536.2095583>.
- [58] J. Pound, P. Mika, and H. Zaragoza. Ad-hoc object retrieval in the web of data. pages 771–780, 01 2010. doi: 10.1145/1772690.1772769.
- [59] P. Rajpurkar, R. Jia, and P. Liang. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, 2018.
- [60] J. Ramos. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning, Vol. 242*. Piscataway, NJ, 133–142, Jan 2003.
- [61] H. Raviv, O. Kurland, and D. Carmel. Document retrieval using entity-based language models. pages 65–74, 07 2016. doi: 10.1145/2911451.2911508.

- 
- [62] R. Reinanda, E. Meij, and M. de Rijke. Knowledge graphs: An information retrieval perspective. *Foundations and Trends® in Information Retrieval*, 14(4):289–444, 2020. ISSN 1554-0669. doi: 10.1561/15000000063. URL <http://dx.doi.org/10.1561/15000000063>.
- [63] L. Richardson, M. Amundsen, and S. Ruby. *RESTful Web APIs*. O’Reilly Media, Inc., 2013. ISBN 1449358063.
- [64] S. Robertson and H. Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, Apr. 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL <https://doi.org/10.1561/1500000019>.
- [65] C. Sammut and G. I. Webb, editors. *TF-IDF*, pages 986–987. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8\_832. URL [https://doi.org/10.1007/978-0-387-30164-8\\_832](https://doi.org/10.1007/978-0-387-30164-8_832).
- [66] M. Sanderson. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4:247–375, 01 2010. doi: 10.1561/1500000009.
- [67] C. Sirangelo. *Cartesian Product*, pages 308–309. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9\_1259. URL [https://doi.org/10.1007/978-0-387-39940-9\\_1259](https://doi.org/10.1007/978-0-387-39940-9_1259).
- [68] J. Surowiecki. The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business. *Economies, Societies and Nations*, 296(5), 2004.
- [69] K. Tan. Basic elasticsearch concepts - elasticsearch tutorial.com, 2015. URL <http://www.elasticsearchtutorial.com/basic-elasticsearch-concepts.html>.
- [70] A. Tonon, G. Demartini, and P. Cudre-Mauroux. Combining inverted indices and structured search for ad-hoc object retrieval. *SIGIR’12 - Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 08 2012. doi: 10.1145/2348283.2348304.
- [71] C. Van Gysel, M. de Rijke, and E. Kanoulas. Learning latent vector spaces for product search. *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, Oct 2016. doi: 10.1145/2983323.2983702. URL <http://dx.doi.org/10.1145/2983323.2983702>.
- [72] C. Van Gysel, M. de Rijke, and M. Worring. Unsupervised, efficient and semantic expertise retrieval. *Proceedings of the 25th International Conference on World Wide Web*, Apr 2016. doi: 10.1145/2872427.2882974. URL <http://dx.doi.org/10.1145/2872427.2882974>.
- [73] U. Visser, S. Abeyruwan, U. Vempati, R. P. Smith, V. Lemmon, and S. C. Schürer. Bioassay ontology (bao): a semantic description of bioassays and high-throughput screening results. *BMC bioinformatics*, 12(1):1–16, 2011.

- [74] D. M. Weigl, D. E. Kudeki, T. W. Cole, J. S. Downie, J. Jett, and K. R. Page. Combine or connect: Practical experiences querying library linked data. In *Proceedings of the Association for Information Science and Technology*, volume 56, pages 296–305. Wiley Online Library, 2019.
- [75] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S. J. Lusher, M. E. Martone, A. Mons, A. L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M. A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, and B. Mons. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1):160018, Mar 2016. ISSN 2052-4463. doi: 10.1038/sdata.2016.18. URL <https://doi.org/10.1038/sdata.2016.18>.
- [76] C. Xiong and J. Callan. Query expansion with freebase. pages 111–120, 09 2015. doi: 10.1145/2808194.2809446.
- [77] C. Xiong, R. Power, and J. Callan. Explicit semantic ranking for academic search via knowledge graph embedding. pages 1271–1279, 04 2017. doi: 10.1145/3038912.3052558.
- [78] M. L. Zepeda-Mendoza and O. Resendis-Antonio. *Hierarchical Agglomerative Clustering*, pages 886–887. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. doi: 10.1007/978-1-4419-9863-7\_1371. URL [https://doi.org/10.1007/978-1-4419-9863-7\\_1371](https://doi.org/10.1007/978-1-4419-9863-7_1371).