Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Master Thesis in Computer Engineering

How to create persistent, human- and machine-readable citations of Web Rankings

Master Candidate

Alessandro Lotta

Student ID 2054786

Supervisor

Prof. Gianmaria Silvello

University of Padova

Academic Year
2022/2023

Abstract

Data citation is a practice that has been a concern for many years and in the last decades has gained a growing interest due to the increasing importance of data in scholarly communication and business field. The practice of citing data aims to give credit and attribution acknowledging the contributions of data creators, collaborators, scientists, and institutions, while also ensuring the reproducibility of data by guaranteeing long-term access to these resources. In this context, an emerging topic revolves around the overlooked domain of reproducible web rankings, a branch of information retrieval that focuses on capturing information related to searches conducted on search engines and systems, and generating citations based on this data. This topic finds applications in various domains, including scientific research and decision-making processes. This thesis introduces the "ranking citation" model developed and made available in the form of a Chrome extension named "Unipd Ranking Citation Tool." It is constructed based on technologies from web development such as RDF graphs, ontologies, JSON languages, and Research Object, adhering to Linked Open Data and FAIR (Findable, Accessible, Interoperable, and Reusable) principles. The tool provided to users is capable of meeting the requirements necessary for reproducing web rankings, offering an automated method to produce both human- and machine-readable resources.

Contents

List of Figures

List of Tables

List of Code Snippets

List of Acronyms

API	Application Programming Interface
CSV	Comma-Separated Values
CLI	Command Line Interface
DOI	Digital Object Identifier
DOM	Document Object Model
JSON	JavaScript Object Notation
JSON-LD	JSON Linked Data
LOD	Linked Open Data
LODE	Live OWL Documentation Environment
NPM	Node Package Manager
OpenAIRE	Open Access Infrastructure for Research in Europe
ORCID	Open Researcher Contributor ID
PNG	Portable Networks Graphics
RDA	Research Data Alliance
RDF	Resource Description Framework
RO	Research Object
RO-Crate	Research Object Crate
ROSC	Research Objects for Scholarly Communication
UI	User Interface
URI	Unique Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium

1 Introduction

Citations are one of the fundamental components in the context of scholarly communication, serving as references to the sources of information and research to support the author's work and the propagation of knowledge. The advent of the Internet and the emerging World Wide Web have introduced new ways of sharing and accessing sources of information. As the web continuously evolves, data has gained central importance having a huge impact on all aspects relative to the web information. Experimental and observational data in the field of scientific research is now stored in scientific databases available on the web. This scientific data has all moved to digital content for most of the research disciplines. In recent years, there has been an emerging need to transition to open, data-rich and linked data environment following the FAIR (Findable, Accessible, Interoperable, Reusable) data principles, with the intent of facilitating the sharing and propagation of knowledge across different domains.

In this context increasingly driven by data, it is progressively becoming crucial to properly attribute citations to the data that is being referenced and used. Citation of resources has evolved into a new form of Data Citation, which is emerging as a critical concern for researchers, scientists, and data authors. The concept of data citation has its origins in the late 90s, although initially it wasn't adopted by researchers due to the lack of interest in implementing tools based on it [22]. In the last years, there has been a growing interest in defining the principles and structures for new data citation systems, driven by the increasing production of data and by a set of motivations identified by Silvello in [31]. Data citation practices acknowledge the efforts and contribution of every data creator and curator involved in the process of generating the research object, uniquely identifie all the resources related to it, enhance knowledge propagation and discovery, and enable the reproducibility of experiments. Additionally, the use of data citation helps to keep track of the impact that the research findings have on the community.

CODATA[1], Research Data Alliance (RDA)[2], Force-11[3], and Research Objects for Scholarly Communication (ROSC)[4] are some of the most important groups of researchers involved in the definition of data citation standards and best practices which culminated in the publication of the Joint Declaration of Data Citation Principles (JDDCP). This document provides the guidelines for data citation implementations, which are: importance, credit and attribution, evidence, unique identification, access, persistence, specificity and verifiability, and interoperability and flexibility. The development of data citation systems still has to take into account different requirements, such as the proper identification of datasets at different levels and the automatic generation of the citation.

An emerging field in the context of data citation is the citation of information retrieval (IR) rankings. Information retrieval is a field of study focused on finding and presenting relevant information from large datasets, while as ranking we intend the ordering of items based on their relevance and usefulness to a specific query. This topic was introduced by the RDA Data citation working group which exposes the applications of and reasons to make retrieval result rankings reproducible [24].

The thesis focuses on a narrower scenario in the citation of IR rankings that is the citation of web rankings, which specifically refers to the ranking of websites on search engines. The vast volume of data constantly created and updated on the web results in a dynamic and continuous flow of information that is subject to constant change. As a consequence, search engines and search systems update their results in response to the queries used for the research, which results in non-static rankings. Several factors affect how these rankings are determined, such as the relevance of the results to the prompted query, keywords and links, new information, and security and speed of the page. Web

[1]https://codata.org/about-codata/
[2]https://www.rd-alliance.org/about-rda
[3]https://force11.org/info/about-force11/
[4]https://www.w3.org/community/rosc/

rankings play a significant role both in research studies and in the business field. For example, they can be applied to extract subsets of data from search platforms such as Google Scholar, Google Search, and Scopus Elsevier, but also from search engines implemented in social media like Twitter. They are also valuable for supporting decision-making processes, illustrating previous study absence, and providing context for a study.

The objective of this thesis is to establish a methodology for reproducing web rankings for the purpose of citing them. To accomplish this task, the thesis presents the implementation of a system that is able to generate citations of web rankings by introducing a new 'ranking citation' model. The tool presented has been developed as a Chrome extension named *Unipd Ranking Citation tool* and is based on the use of FAIR and Linked open data (LOD) principles. By following these principles the tool enables seamless interconnection with other resources available on the network, making data accessible and promoting transparency, reproducibility, and open access to the research objects.

The extension provides an accessible user interface that is activated as soon as the user performs a query search on one of the supported platforms, including Google Scholar, Google Search, Scopus, Bing, and Twitter. The generated web rankings are captured by the extension, which starts a process for generating the citation. This process involves different phases namely data gathering, creation of the output resources, upload and publication on Zenodo, and the construction of the textual citation. In synthesis, the whole process consists of capturing all the relevant information displayed on the result page, creating the output resources (RDF graph, screenshot, and the related metadata) from this data, uploading these resources on a specific online service, and publishing them ensuring persistency and accessibility. The final phase involves the creation of the textual citation which includes the metadata relative to the author, collaborators, affiliation, title of the research, publication date, and the identifier of the resource. All the citations generated are displayed in the popup of the extension, where they can be retrieved, copied, or removed.

The development of the tool was directed into an open data approach, for this reason, the Zenodo service was chosen to publish the research objects.

Regarding the technological aspect, the extension is built upon the RDF graph structure and JSON-LD (Linked Data) language, which is used as a serialization method for storing the RDF triples. The RDF triples describe the information captured using the subject-predicate-object formulation and are integrated with the Ontology model (*Ranking Citation Ontology*). The ontology provides extensive documentation for the classes, properties, and attributes utilized. The classes contained in the ontology are used to model the essential components of the web rankings and those involved in the creation of the citation. The *System* class represents the platform used to make the research and it can vary between search engines dedicated to scholar publications (Google Scholar, Scopus), or integrated in social networks (Twitter), or general purpose search engines (Google Search, Bing). The *Search Query* describes the query prompted by the *User* and executed by the System in use. The Search Query produces the *Ranking Snapshot* which represents the result page displaying the resulting rankings and the other visible information. The results are organized and ordered implementing *rdf:List*, and they are modeled with the *Search Result* class.

An important component of the extension is the integration of the Research Objects, in the form of RO-Crate. This is a tool introduced by the ROSC group in 2013 with the intent of providing a way to package research artefacts in a way that is both human and machine readable. Research Objects are introduced in the extension with the `ro-crate-metadata.json` file, which allows to define the structure of the deposit and to package the output resources for the generation of the citation.

The developed tool is able to accomplish the task of providing a way of citing a web ranking, with an automatic process that guarantees consistency and completeness. The tool was also developed to be as easy as possible to configure and utilize, indeed it was built as a Chrome Extension, upon the Chrome Extension CLI framework, so that everyone is able to run it in their own browser.

The thesis is organized in the following way.

The second chapter is dedicated to the presentation of the research topic by introducing the concept of data citation and its significance. It describes past contributions in the field, its evolution over time, and its main concerns and principles. Moreover, it highlights the importance of citing web rankings. Chapter 3 delves into the core technologies and methods involved in the creation of the ranking citation model, focusing on data structures, RDF graphs, the Ontology model, and Research Objects. The following chapter explains the steps involved in the generation of the actual citation. This includes the process of data gathering, the upload and publication of files in Zenodo, and the construction of the textual citation. Chapter 5 provides an in-depth explanation of the architecture of the developed tool, starting from the foundational components, such as the Chrome extension framework, and following the workflow of the extension to describe its details. The last chapter showcases one example of usage of the tool, facing the steps of installation and setup, and describing the user interface and experience while providing screenshots of the tool for better comprehension of the context.

2 Background

2.1 Data Citation

Various definitions are proposed for the concept of data citation: "A key practice underpinning the recognition of data as a primary research output rather than as a by-product of research" [11], "formal ways to ground the research findings in a manuscript, upon their supporting evidence, when that evidence consists of externally archived datasets." [6]. A definition that is generally accepted is "Data citation is the practice of referencing data products used in research. A data citation includes key descriptive information about the data, such as the title, source, and responsible parties." [10]. This is the definition of data citation given by the United States Geological Survey (USGS), an agency founded by the United States government, dealing with various scientific disciplines including biology, geography, geology, and hydrology.

Data citation is not a new concept, Parsons et al. [22] in a recent review from 2019 describe the history of data citation from the late 1990s when this concept emerged, and its evolution through the years. They state that it has gained increasing interest over the past few decades due to recent advances in technologies and driven by the need for automated citation generation methodologies. The article also highlights that the basic principles have been established but there are remaining concerns that need to be evaluated regarding human credit and machine accessibility to the data.

In [8] Crosas et al. review the evolution of data citation standards and practices in a similar way. Additionally, they outline the benefits of citing data, including the enhanced ability to link data with the publication referring to it, enabling new forms of scholarly publishing, promoting interdisciplinary research, and facilitating replication and extension of previous research. Moreover, they also present an example of a possible citation format, similar to the one used for referencing publications following the BibTeX citation style:

> *Author(s), Year, Dataset Title, Global Persistent Identifier, Data Repository or Archive, version or subset*

In the article from 2018 [31], Silvello delves deep into the aspects concerning the fundamentals of data citation, providing an overview of various contributions and approaches in this field. The article emphasizes the importance of treating data as first-class research objects and highlights the need for a general agreement on data citation rules and practices. The paper analyses what are the main motivations and the needs for citing data. A great incentive is given by the increasing production of data in research findings, together with the introduction of new tools for managing, accessing, analyzing, and sharing data happening in the latest years. The main motivations for data citation can be identified in:

- Data attribution
- Data connection
- Data discovery
- Data sharing
- Data impact
- Reproducibility

Data attribution consists of giving the proper credit to scientists and researchers who produce and manage the data. Proper attribution is achieved by correctly specifying the people responsible for creating, curating, collecting, and analyzing the research data. Data connection and discovery express the need to relate the scientific papers with the data on which they are based and to identify this data with consistent methods. These features enhance access and consultation of the data and allow the expansion of the researcher's knowledge by reaching and discovering new resources. Data sharing is also an essential factor enabling data discovery and reuse, and it represents an incentive for collaboration and further studies on the research topic. The recognition of data impact is an essential component of data citation. Data impact is intended as a way to measure and discover how and where the data being cited is used, an aspect that is relevant also for funding agencies directing investments into the research field. The 'Make Data Count' project [7], through the COUNTER Code of Practice for Research Data [13], defines a way to count data downloads, which are considered the most valuable measures of data impact together with the citation count. The importance of these two aspects is identified in the research of Kratz and Strasser [17], which also highlights the importance of having defined and formal citation practices adopted by researchers. Data citation has also an impact on reproducibility, enabling other researchers to reuse the data and replicate the workflows for verifiability and validation purposes.

An essential incentive in the adoption of data citation for scholarly publications is the integration with Linked Open Data. The connection of Linked Open Data principles to the data citation model represents a significant factor strictly related to the motivations for citing data and in particular for data sharing, data connection, and data discovery.

In [30], Silvello addresses the topic of data citation in the context of Linked Open Data (LOD). The paper presents a methodology for effectively citing LOD subsets and discusses the impact of using this methodology, which is based on named graphs Resource Description Framework (RDF) quads semantics. Through a use case and application of this methodology, the paper demonstrates that "exploiting the LOD paradigm enables persistent, dereferenceable, variable granularity and human- and machine-readable citations of LOD subsets". This enhances the reproducibility, transparency, and knowledge sharing of research.

The need for more transparency, openness, and reproducibility in scientific research is also addressed in the journal article from 2015 ([19]) titled "Promoting an Open Research culture", which highlights the limited implementation of open practices in current research studies. Additionally, it introduces the Transparency and Openness Promotion (TOP) guidelines for shared and open practices across journals, including citation and replication standards, and data-sharing standards.

2.1.1 Principles of Data Citation

Over the last decade, various working groups have focused their efforts on establishing and implementing standards and best practices for data citation.

CODATA-ICTSI (International Council for Scientific and Technical Information), Research Data Alliance (RDA) Data Citation Working Group, Force-11 Data Citation Group, and Research Objects for Scholarly Communication (ROSC) (a World Wide Web Consortium (W3C) community established in 2014) are the most important among others. As stated in [3], these groups have considerable overlap in membership and collaborate with each other through joint activities and cross-memberships for the task of defining data citation principles, but they have different areas of interest. For example, RDA is interested in building infrastructures for data management, Force-11 is involved in reforming scholarly communication in a general context, while ROSC is concerned with "technical approaches for managing data, publications, software, and other objects created in scientific research" [3].

In [1, 25] are presented the principles defined by the various groups, and in the manifesto [18] is finalized the *Joint Declaration of Data Citation Principles* (JD-DCP) that consists in the list of eight principles obtained from the collaboration of the task groups:

- **Importance**: Data should be considered legitimate, citable products of research. Data citations should be accorded the same importance in the scholarly record as citations of other research objects, such as publications
- **Credit and attribution**: Data citations should facilitate giving scholarly credit and normative and legal attribution to all contributors to the data, recognizing that a single style or mechanism of attribution may not be applicable to all data.
- **Evidence**: In scholarly literature, whenever and wherever a claim relies upon data, the corresponding data should be cited.
- **Unique Identification**: A data citation should include a persistent method for identification that is machine actionable, globally unique, and widely used by a community
- **Access**: Data citations should facilitate access to the data themselves and to such associated metadata, documentation, code, and other materials, as are necessary for both humans and machines to make informed use of the referenced data.
- **Persistence**: Unique identifiers, and metadata describing the data, and its disposition, should persist—even beyond the life span of the data they describe
- **Specificity and Verifiability**: Data citations should facilitate identification of, access to, and verification of the specific data that support a claim. Citations or citation metadata should include information about provenance and fixity sufficient to facilitate verifying that the specific timeslice, version

and/or granular portion of data retrieved subsequently is the same as was originally cited.

- **Interoperability and Flexibility**: Data citation methods should be sufficiently flexible to accommodate the variant practices among communities, but should not differ so much that they compromise interoperability of data citation practices across communities.

Many of these principles are also part of the reasons explaining the importance of data citation for the research community. Among them, Identification is an essential feature that allows us to uniquely identify datasets and research outputs by assigning each one an alphanumeric string. Methods and techniques for referencing this data are required to be specific and guarantee persistence over time. The most used identifiers include Archival Resource Key (ARK), Universal Unique Identifier (UUID), Research Resource Identifiers (RRID), and Digital Object Identifier (DOI). These identifiers are globally unique alphanumeric strings assigned by a registration agency and provide a persistent link to the location of the resource, ensuring long-term accessibility and referencing. It is worth noting that various identifiers have or are being developed for other research objects and entities. Among them, the Open Researcher Contributor ID (ORCID)[1] was developed for assigning an identifier to individual researchers, authors, and contributors of scholarly communication. The Research Organization Registry (ROR) Community[2], is developing persistent identifiers assigned to research organizations.

2.1.2 Implementation and system requirements

The implementation of a data citation system needs to take into account different considerations and concerns.

In the context of unique identification, the use of persistent identifiers (PID) such as DOI is not enough to solve all the identification problems. The reason for this is that a citation system needs to be able to uniquely identify and cite accordingly different levels of details and subsets of data [9], the same happens for bibliographic citations where it is possible to both cite entire documents or cite only single pages and even single passages. By adopting variable granularity practices, data citation systems enable researchers to precisely reference specific

[1]https://orcid.org/
[2]https://ror.org/

portions of the dataset. This is particularly important for interdisciplinary research, where different disciplines could be in need of referencing only small and different sections of data. Different solutions have been proposed to overcome the granularity problem: using Research Resource Identifiers (RRIDs) assigned to scientific resources [2], assigning a PID with a unique key to identify every single XML node exploiting a rule-based citation system [5], assigning PIDs to queries [26] which allows access and retrieval of a specific data subset, and adopting the nanopublication model [14] for research data.

Nanopublications "are the smallest unit of publishable information" and "are expressed in a knowledge graph format that is formal and machine-interpretable"[3]. They are implemented with the RDF language and are composed of three elements: the **assertion** which constitutes the main content of the nanopublication, the **provenance** describing how the assertion has been generated with its methods, and the **publication info** containing all metadata about the nanopublication. Fabris et al.[12] proposed the nanocitation framework for citing single nanopublications and an open-source system[4] that enables the automatic creation of citation text snippets and landing pages.

Another fundamental concern required in the implementation of a data citation system is the need for an automatic process for the generation of the citation [4, 26], which should include the necessary and relevant metadata information and a consistent format and style. Completeness and consistency of data citations can be achieved by providing tools and software that enable automatic citation functionalities to users who are not experts in this field. Some scientific databases and services already include this functionality. An example is the Zenodo open repository, which allows researchers to deposit any research material and assigns a DOI to each upload. The service also enables the user to choose the format in which the citation should be exported, including BibTeX, CSL, DataCite, Dublin Core, and JSON/JSON-LD.

Other relevant concerns regarding citation systems include fixity, which refers to the ability to keep the data accessible and unchanged over time. This involves developing systems that implement a versioning mechanism and safeguarding against losses and corruption of data. Finally, data citations should be able to adapt to the different data types, models, and formats implemented in

[3]`https://nanopub.net/`

[4]`http://nanocitation.dei.unipd.it/`

different domains.

Various solutions have been implemented to accomplish the task of data citation. The Dataverse[5] project is an open-source software web application that allows researchers to share, preserve, cite, explore, and analyze research data. It is supported by the Institute for Quantitative Social Science (IQSS) at Harvard University. Among its features, it offers the possibility to cite the dataset in a standardized and automatic way, allowing researchers to get credit and recognition for their work. Additionally, it incorporates the Universal Numerical Fingerprint (UNF) mechanism, which consists of a PID to specific digital objects or subsets of data and allows to identification of data with different granularities. Dataverse has a lot of features but is not able to handle dynamic datasets and the main disadvantage is that it only allows citing the data that has to be manually gathered and stored in a Dataverse repository.

Another relevant solution is represented by the Whole Tale project [6]. This is an "NSF-funded Data Infrastructure Building Block (DIBBS) initiative to build a scalable, open source, web-based, multi-user platform for reproducible research enabling the creation, publication, and execution of tales" [33]. These tales are Research Objects encapsulating both data and code, together with the entire software environment in which the research is conducted. Additionally, tales are completed with their metadata information. This project provides a good example of the efforts directed at finding ways to package data and generate a citable and re-executable Research Object. Nonetheless, this has to be combined with other tools for generating the citation from the packaged resources.

The article [23] proposed an approach to persistently cite arbitrary subsets in relational databases, a method based on the use of deterministic timestamped queries and incorporates a database schema with a versioning system to handle evolving data. This system, however, doesn't provide a suitable means to automatically generate human and machine-readable citations. Moreover, as mentioned above, in [30] Silvello proposed a method for producing human and machine-readable data citations based on the citation of RDF subgraphs through the use of named graphs. This approach focuses on the specific subject of Linked Open Data providing the descriptive metadata and uniquely identifying the cited object. Additionally, it presents a solution to the problem of variable

[5]`https://dataverse.org/best-practices/data-citation`
[6]`https://wholetale.org/`

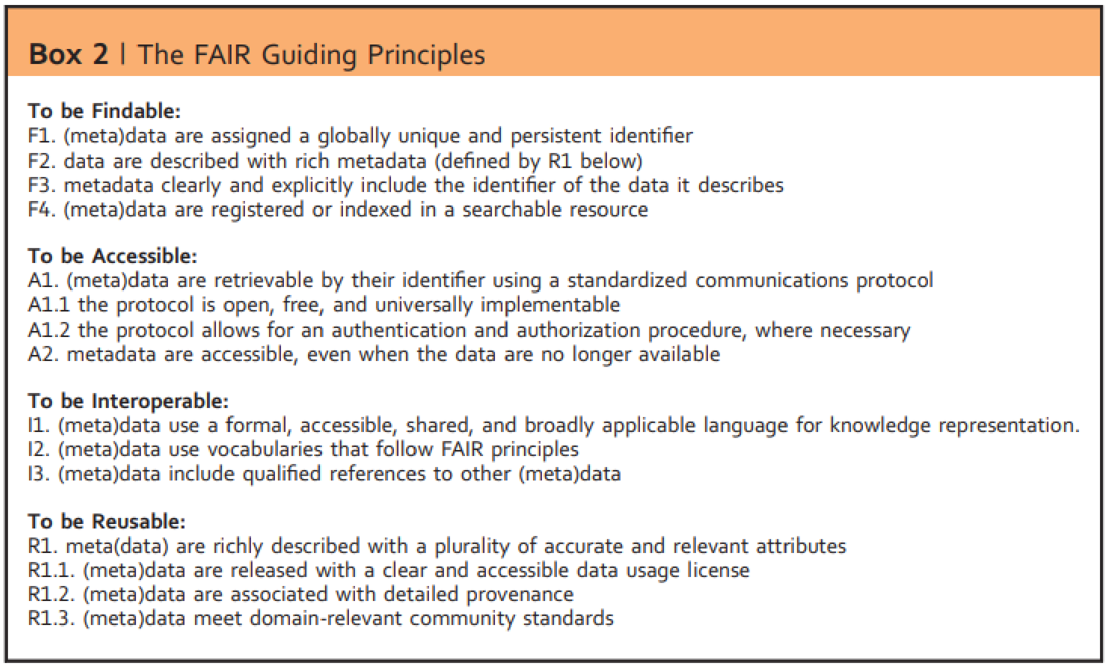
granularity but the composition of the references is not done automatically.

The existing solutions and approaches present important contributions to the field of data citation but they don't address the specific subject of web rankings. In the citation of web rankings, it is necessary to have a tool that is able to collect information about the whole search environment including the results, their ordering, the user information, the system, and the query, and once this data is collected the tool should generate the reference from it. All of this has to be done in an automatic way, which is helpful in reducing errors and avoiding the repetition of the gathering process for each result and for big quantities of data. From this data, the tool should also be able to produce both human and machine-readable resources facilitating human comprehension and maintaining an optimal efficiency for machine operations. Moreover, due to the dynamic nature of the web, a versioning system is needed to keep track of the changes in different time instants and to replicate past versions of the data captured.

Overall, the solutions presented above have a positive impact on data citation but they are not able to provide all the listed features required for web rankings in a single tool.

2.2 FAIR Data principles

Data sharing is an important advantage provided by data citation practices. The research community benefits from data sharing which promotes transparency, encourages collaboration, enables better decision-making, and facilitates knowledge discovery. In this context, a group of academics and private stakeholders reunited in a workshop named 'Jointly Designing a Data Fairport' held in the Netherlands in 2014. The conclusions obtained from the workshop resulted in "The FAIR Guiding Principles for Scientific Data Management and Stewardship" [35]. The article was reviewed, curated, and published by a dedicated group composed of members of the FORCE11 community in 2016, in the journal called 'Scientific Data', an open-access, online journal for descriptions of scientifically valuable datasets, and research that advances the sharing and reuse of scientific data. The FAIR Data principles present a set of concise, high-level, domain-independent guidelines to support the reuse of scholarly data, in response to the need to define the goals and desiderata of good data management and stewardship. These principles are not only applied to the data itself,



Box 2 | The FAIR Guiding Principles

To be Findable:
F1. (meta)data are assigned a globally unique and persistent identifier
F2. data are described with rich metadata (defined by R1 below)
F3. metadata clearly and explicitly include the identifier of the data it describes
F4. (meta)data are registered or indexed in a searchable resource

To be Accessible:
A1. (meta)data are retrievable by their identifier using a standardized communications protocol
A1.1 the protocol is open, free, and universally implementable
A1.2 the protocol allows for an authentication and authorization procedure, where necessary
A2. metadata are accessible, even when the data are no longer available

To be Interoperable:
I1. (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.
I2. (meta)data use vocabularies that follow FAIR principles
I3. (meta)data include qualified references to other (meta)data

To be Reusable:
R1. meta(data) are richly described with a plurality of accurate and relevant attributes
R1.1. (meta)data are released with a clear and accessible data usage license
R1.2. (meta)data are associated with detailed provenance
R1.3. (meta)data meet domain-relevant community standards

Figure 2.1: A description of the FAIR principles taken from [35]

but also to the algorithms, tools, and workflows that are involved with that data. These guidelines support both manual and automated deposition, exploration, sharing, and reuse on a wide range of technologies and implementations.

FAIR stands for Findable, Accessible, Interoperable, and Reusable. These are the four core concepts extracted from the original meeting, they are reported in Figure 2.1.

A great number of stakeholders benefits from the application of these guidelines: researchers who want to share, get credit, and reuse each other's data; professional data publishers offering their services; software and tool builders for data analysis and reusable workflows; funding agency with an increasing interest for long term data stewardship; and the research community for advancing and discovery of knowledge concerning data. These principles provide great advantages also in tasks of data retrieval and analysis assigned to computational agents and machine operators, allowing discovery, access, and integration of scientific data and scholarly objects.

2.3 Citing Web Rankings

The task of citing web rankings poses a significant challenge for which no solution has been proposed until this thesis. Web rankings are generated by search engines that are running on the web and provide websites, documents, and other data related to the query prompted by the user. The RDA Data Citation Working Group recognizes the citation of information retrieval rankings as a relevant and emerging concern in the field of scientific research [24]. The group emphasizes the necessity to implement a mechanism that allows reproducing retrieval result rankings, which are continuously updated due to the addition, alteration, and deletion of their documents. The challenge that is posed consists of preserving the state of the system and the relevant information relative to the elements in the ranking list.

Reproducible rankings find applications in various domains including scientific experiments, business intelligence reports, online learning systems, and auditable decision-making processes. In scientific experiments and research, it is sometimes needed to extract subsets of data or documents from digital publication database systems like Google Books and Medline, or from social media sites, Twitter feeds, and Wikipedia pages, where content is frequently updated. Business intelligence reports rely on reproducible rankings to justify and support decision-making, especially in the context of press monitoring and social media surveillance. Similarly for rigorous auditing in contexts like patent retrieval, reproducible rankings ensure transparency and fairness by enabling evaluation of the quality and relevance of the retrieved information and addressing potential biases and errors.

To address these challenges, the thesis proposes a “ranking citation” model developed in the form of the *Unipd Ranking Citation tool*, which enables researchers and users to automatically generate reproducible and persistent citations in the field of web rankings. The tool is built as a Chrome extension to facilitate its access and usage from any browser and for anyone who needs it in the above-mentioned contexts.

3 Technologies and methods

3.1 Citation Model

In the context of citation and cited data, adherence to the FAIR (Findability, Accessibility, Interoperability, Reuse) data principles [35] is essential. While these principles are generally applied to data in a broader context, they have particular importance in complementing the data citation principles, which include persistence, immutability, and reusability for both machine and human interpretability. To achieve these goals it is important to include both human- and machine-readable resources with the citation object.

Machine-readable data formats, which include Comma-Separated Values (CSV), RDF, and JavaScript Object Notation (JSON), are designed for devices and machines, and are usually complex to comprehend for humans. They enable rapid extraction and utilization without the need for human intervention. On the other hand, human-readable data aims to facilitate human usage and understanding, exploiting resources such as PDF documents containing natural language, or visual representations like images and tables to enhance comprehension.

To guarantee human readability, the tool incorporates a set of functions to capture single or multiple screenshots of the web pages displaying the rankings to be cited. These captured pages are then saved in Portable Networks Graphics (PNG) format, which has the advantage of implementing a lossless compression along with gamma and chromaticity correction, making it more suitable for

Web-based teaching and projects [34] compared to other image formats such as JPEG, and GIF. Additionally, it produces files of bigger size resulting in sharper and higher quality images overall.

The captured screenshots display all the information contained in the resulting pages, and the number of screenshots taken for a specific capture depends on the number of pages specified in the settings of the tool. For each supported search system, the information present in the screenshot includes the user query in the search bar, the resources and links resulting from the research, the filters included, and other relevant elements. These screenshots complementing the other output files are effective in providing visual confirmation of the accuracy and correct execution of data capturing. Moreover, they offer a rapid and easy-to-understand way of presenting the data captured.

For machine readability, the tool is able to extract information from web rankings and generate the output data in the form of an RDF Graph. This serves as a structured representation for efficient data processing and integration in machine-readable contexts.

In the following subsections are described the features incorporated in the citation tool.

3.2 RDF

RDF [27] is a standard model for data interchange and sharing across different sources and domains on the web. It serves as the foundation for the representation and organization of knowledge and, as already stated, RDF provides a standard model for representing and exchanging data in a machine-readable format. RDF was initially adopted as a recommendation by W3C in 1999, and subsequent specifications were presented. The RDF 1.1 specification was published in 2014 and is currently being adopted, while a new version (RDF 1.2) is in development.

The RDF graph is structured as a collection of interconnected nodes, representing the entities or resources, and edges, which are relationships that depict the properties among them. The RDF data is stored in triples, each one expressing a statement in the form of subject-predicate-object. The subject in the triple is a resource identified with a Unique Resource Identifier (URI) or a blank node. The object also indicates a resource identified with URI, but it can also be a literal or a blank node. The predicate is a resource connecting the subject and

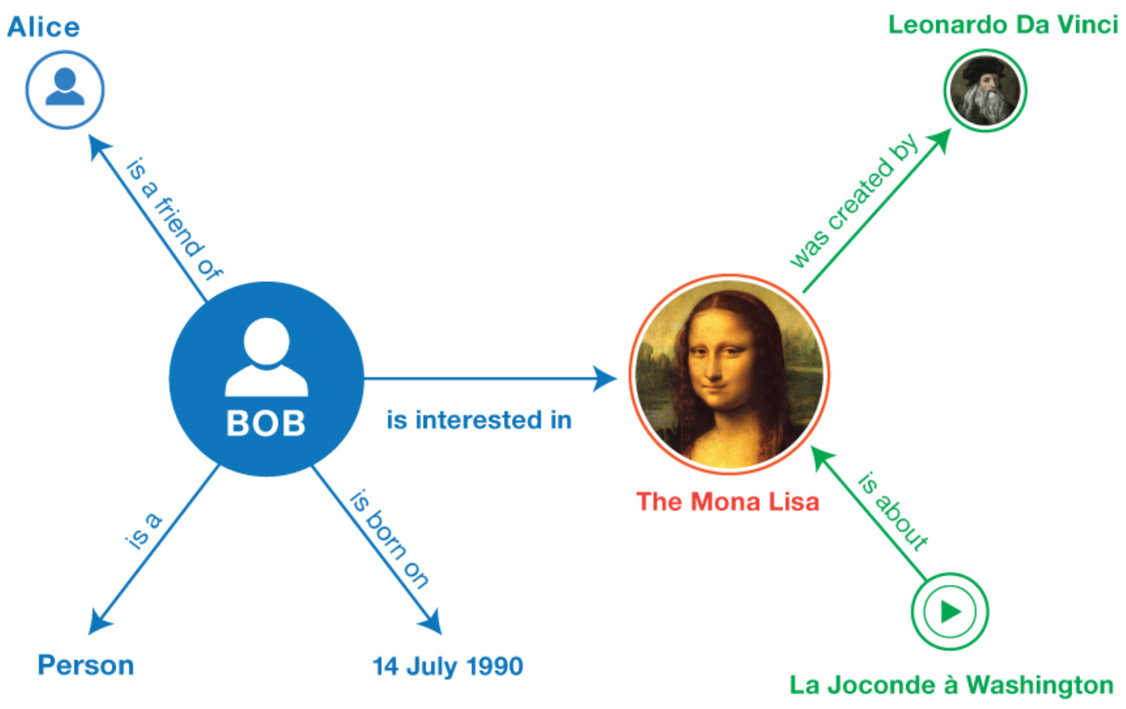


Figure 3.1: Example of an RDF graph taken from `https://www.w3.org/TR/rdf11-primer/`

object in a relation, identified again with a URI. Blank nodes are anonymous resources without a URI that cannot be used in a global scope on the Web but allow to represent logical reasoning within the local graph, such as ordering and representation of structured values. Literals are used to identify values such as numbers and dates. They are divided into plain literals and typed literals. Plain literals are strings combined with a language tag, used for natural language. Typed literals, instead, are strings combined with a datatype URI, which restrict and define the values to be used.

URIs are unique sequences of characters that identify a logical or physical resource used by web technologies. The use of URIs allows for addressing every resource in the graph, making them accessible across the web. Nodes from different graphs having the same URI are merged together. By utilizing globally unique identifiers, RDF graphs enable the integration and interlinking of heterogeneous datasets from various sources and allow for the representation of Linked Open Data.

RDF triples are represented and stored with types of serialization. One way to represent RDF graphs is with the use of graphical diagrams [], however, this is efficient for human comprehension but not understandable by machines. Other common ways are text representations, which include:

- N-Triples[1]: the simplest form of serialization for the RDF triples, using full URIs for the resources written between angle brackets (< >). Each triple terminates with a '.'.
- Turtle/N3[2]: an extension of N-Triples combined with Curie notation, providing abbreviations and namespaces to reduce the amount of text and enhance interpretability
- RDF/XML[3]: it's the first RDF serialization used. More verbose than the others due to the XML tree structured syntax, well suited for web infrastructures, but it is difficult to comprehend and use.
- JSON-LD[4]: modern approach preferred in web applications, combining JSON and Linked Data. It is also easy to read and use for those accustomed to working with JSON.

Other types of serializations used are N-Quads for named graphs and TriG, an extension of the Turtle format.

```
<http://dbpedia.org/resource/Bob_Marley> <http://www.w3.org
    /1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person>
     .
<http://dbpedia.org/resource/Bob_Marley> <http://www.w3.org/2000/01/
    rdf-schema#label> "Bob Marley"@en .
<http://dbpedia.org/resource/Bob_Marley> <http://www.w3.org/2000/01/
    rdf-schema#label> "Bob Marley"@fr .
<http://dbpedia.org/resource/Bob_Marley>
<http://www.w3.org/2000/01/rdf-schema#seeAlso> <http://dbpedia.org/
    resource/Rastafari> .
<http://dbpedia.org/resource/Bob_Marley> <http://dbpedia.org/ontology
    /birthPlace> <http://dbpedia.org/resource/Jamaica> .
```

Code 3.1: N-Triples serialization

```
@prefix dbr: <http://dbpedia.org/resource/>.
@prefix dbo: <http://dbpedia.org/ontology/>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.

dbr:Bob_Marley
    a foaf:Person ;
```

[1]https://www.w3.org/TR/n-triples/
[2]https://www.w3.org/TR/turtle/
[3]https://www.w3.org/TR/rdf-syntax-grammar/
[4]https://www.w3.org/TR/json-ld11/

```
    rdfs:label "Bob Marley"@en ;
    rdfs:label "Bob Marley"@fr ;
    rdfs:seeAlso dbr:Rastafari ;
    dbo:birthPlace dbr:Jamaica .
```

Code 3.2: Turtle serialization

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:ns0="http://dbpedia.org/ontology/"
         xmlns:foaf="http://xmlns.com/foaf/0.1/">

  <foaf:Person rdf:about="http://dbpedia.org/resource/Bob_Marley">
    <rdfs:label xml:lang="en">Bob Marley</rdfs:label>
    <rdfs:label xml:lang="fr">Bob Marley</rdfs:label>
    <rdfs:seeAlso rdf:resource="http://dbpedia.org/resource/Rastafari
   "/>
    <ns0:birthPlace rdf:resource="http://dbpedia.org/resource/Jamaica
   "/>
  </foaf:Person>

</rdf:RDF>
```

Code 3.3: RDF/XML serialization

```
{   "@context": {
        "dbr": "http://dbpedia.org/resource/",
        "dbo": "http://dbpedia.org/ontology/",
        "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
        "foaf": "http://xmlns.com/foaf/0.1/"    },
    "@graph": [
        {   "@id": "dbr:Bob_Marley",
            "@type": ["foaf:Person"],
            "rdfs:label": [
                {"@value": "Bob Marley", "@language": "en"},
                {"@value": "Bob Marley", "@language": "fr"}
            ]
            "rdfs:seeAlso": [{"dbr:Rastafari"}],
            "dbo:birthPlace": [{"dbr:Jamaica"}]
        }
    ]
}
```

Code 3.4: JSON-LD serialization

Generally, the Turtle serialization is the preferred one in combination with the Curie notation, which allows to define bindings of URIs to single words. It uses the dot ('.') to end a triple, but it also adds other two elements, the semicolon (';') to add triples sharing the same subject, and the comma (',') to separate objects that have the same subject and predicate, without the need to write them on different lines.

In the case of the *Unipd Ranking Citation Tool*, the serialization method adopted is JSON-LD which is more suitable for Web applications, and easy to read by researchers who are not specialized in programming languages. Additionally, JSON-LD is designed to integrate with modern Web Application Programming Interface (API)s and allows for easy linking with other datasets thanks to its linked data nature.

RDF graph data can be manipulated with the use of SPARQL (SPARQL Protocol and RDF Query Language) query language [5], which recommendations are specified by the W3C consortium. SPARQL queries assume the format of Turtle syntax and are used to retrieve triple patterns along with their conjunctions and disjunctions. SPARQL also supports aggregation, subqueries, negation, creating values by expressions, extensible value testing, and constraining queries by source RDF graph.

3.3 Ontology Model

Building the RDF graph for the *Unipd Ranking Citation tool* involves the creation of the Ontology model that serves as the foundation for representing the specific domain of interest, i.e. web rankings and data citation.

The term ontology was first introduced in the philosophical field concerning the study of the concepts of being, existence, becoming, and reality [20]. This concept was then adopted by AI researchers to indicate computational models which enable certain kinds of automated reasoning[21]. Over time, the term evolved to refer to theories of modeled worlds and knowledge system components.

While the RDF graphs provide a framework for representing data in a flexible and extensible way, the purpose of ontologies is to define the semantics and

[5]https://www.w3.org/TR/sparql11-query/

formalization, enabling a more comprehensive and structured representation of the knowledge. Integrating the ontology model with RDF involves mapping the concepts, properties, and relationships to the actual data in the graph.

Ontologies describe the concepts, entities, and relationships within a given domain, providing a vocabulary of terms and a set of relationships to govern their usage. The same word adopted in different domains could assume different meanings, which are defined by its ontology. In the context of the *Unipd Ranking Citation tool*, the ontology helps in defining and establishing the meaning of terms like "Search Query," "System," "Ranking Snapshot," "User," and "Search Result," along with their interconnections and properties.

The key components that define the structure of the ontology model are individuals, relationships, attributes, and classes (Figure 3.2). Individuals or instances are the basic components of an ontology, they represent specific objects or instances in the domain of interest, including both concrete and abstract objects. Relationships, often referred to as Object Properties, are binary relations between individuals and classes to other individuals and classes. They are used to represent the ontology structure, and thanks to the `subClassOf` statement they enable to define hierarchies. Attributes or Data properties are used to describe the characteristics and values of individuals in the ontology, the values of the attributes are defined using specific data types. Furthermore, Classes, also known as concepts, are abstract collections of individuals and objects, defining concepts or categories within the domain, and describing the requirements for membership in the class. Classes can be organized in taxonomies (knowledge organized in parent/child structure) through the use of relationships between them. The ontology model also includes restrictions, rules, and axioms to express constraints, logical relationships, and inference rules. They allow to specify additional constraints within the ontology.

An important category of ontologies are the Foundational ontologies represent models of common objects (concepts) which are applicable across a wide range of domains. They can be used as building blocks and integrated into other ontologies to provide a more comprehensive description of the concepts facilitating the interoperability among ontologies aligned to the same foundational ontology. They also have the advantage that one doesn't need to 'reinvent the wheel', but can adopt the basic categories and relations, already defined, to represent the subject domain. Some examples of the most used foundational

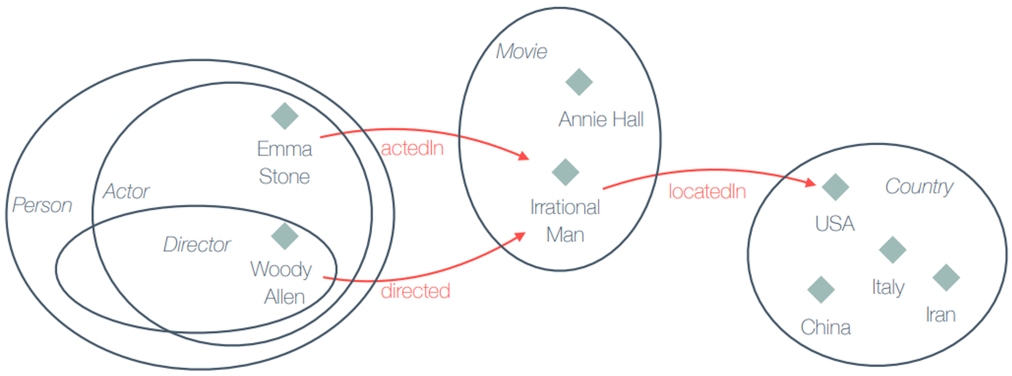


Figure 3.2: Representation of an example of an ontology. The diamonds represent the individuals, the arrows are the relationships, and the circles are the classes that group the individuals and have a hierarchical structure

ontologies are Dublin Core Metadata Initiative (DCMI)[6] a set of fifteen items to describe digital or physical resources, focusing on essential attributes like title, creator, subject, description, and date; Friend of a Friend (FOAF)[7] Ontology designed to represent social relationships and personal information, finds applications in social networking and in the creation of people's profiles; Simple Knowledge Organization System (SKOS)[8] designed to represent taxonomies and classification systems, allowing for the creation of hierarchical structures, associative links, and mappings between concepts.

3.4 JSON and JSON-LD

JSON [15] is a lightweight interchange format for storing and transporting data, commonly used to represent structured data. It is a format that at the same time is easy to generate and parse from a machine or device, and it is easy to read and understand also for humans. It is syntactically identical to the code for creating JavaScript Objects but independent from any programming language, nonetheless, many of the most used programming languages provide functions for parsing and generating JSON objects. The structure of a JSON object is composed of key-value pairs called properties, separated by commas and surrounded by curly braces '{}'. The key-value pairs consist of a key (string),

[6]https://www.dublincore.org/
[7]http://xmlns.com/foaf/0.1/
[8]https://www.w3.org/2004/02/skos/

and a value that can be of type string (strictly in double quotes), number, boolean, object, or array.

JSON Linked Data (JSON-LD) [16] is an extension of JSON, designed for representing Linked Data, which supports linking to other resources on the web. JSON-LD is compatible with all the JSON parsers and generators, and it introduces global identifiers through URIs as in RDF to disambiguate keys shared among different JSON documents, the ability to annotate strings with their language, and facility to express one or more directed graphs. The JSON-LD format is recommended by W3C and RDA to represent RDF data, both for the advantages stated above of combining both human- and machine-readability and for the simplicity of use inherited from the JSON objects.

3.5 The Research Object

A recent approach for implementing FAIR data was introduced with the notion of Research Object (RO) [28]. The concept of Research Object emerged as a means to address the challenges of organizing, sharing, and reproducing while preserving scientific research artifacts.

In 2013 a working group from the W3C community called Research Objects for Scholarly Communications (ROSC)[9] was established. The goal of the working group is to create a platform for scholars, librarians, publishers, archivists, and policymakers to discuss the requirements and expectations for defining a new form of scholarly communication which enables better reuse and reproduction of research results and knowledge [29]. ROSC aims to define the requirements, gather use cases, and propose best practices and guidance for facilitating the publishing and sharing of research objects. Research Objects are intended as the combination of all the research assets, including data used and generated, methods for producing the data, together with people and organizations involved in the study.

Following the core principles of identity, aggregation, and annotation, the Research Object offers several benefits for the scientific community. Firstly, it enables reproducibility by packaging the research artifacts along with their metadata, code, and workflows. This helps for the repeatability of the experi-

[9]`https://www.w3.org/community/rosc/`

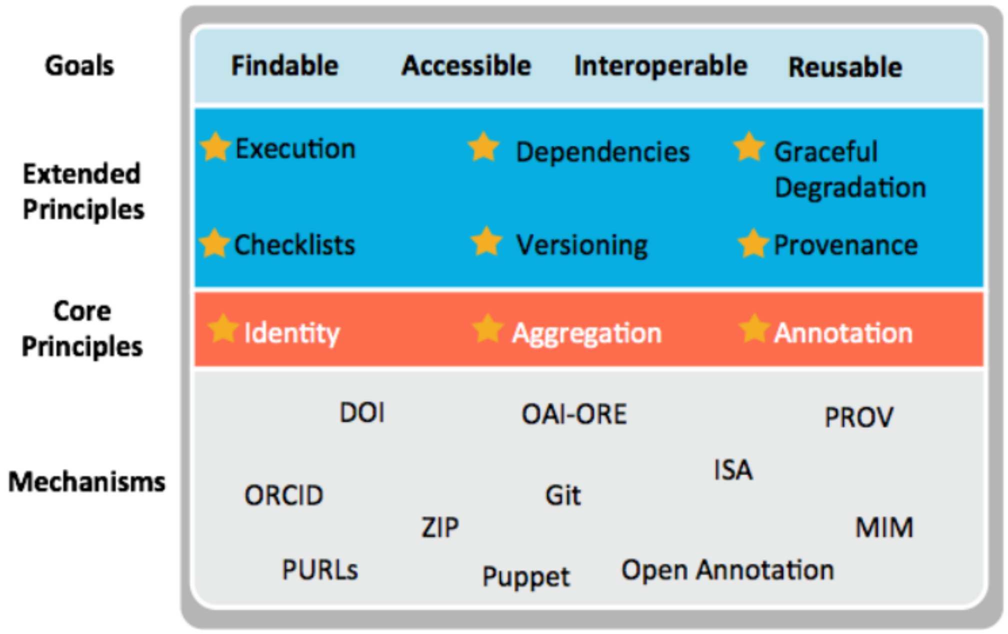


Figure 3.3: Research Objects goals, principles, and methods taken from `https://www.researchobject.org/overview/`

ments and the permanent storing with their context. Secondly, Research Objects provide a unified structure to organize and describe the research components, allowing for easy understanding and access to the data and methodologies. Thirdly, they facilitate the sharing and referencing by bundling the documentation together with the resources. This contributes also to the longevity and preservation of the outputs, by making them accessible and understandable over time. Other fundamental principles that research objects should exhibit include reusability, traceability, attribution, metadata, lifecycle, management, and security. Figure 3.3 showcases the main goals, principles, and methods adopted by the initiatives involved in the Research Objects field.

The major problem encountered with Research Objects was that the implementation required a large technology stack and wasn't easily usable by end-user. That's why a community of researchers, developers, and publishers introduced and developed the Research Object Crate (RO-Crate), to facilitate the generation of multiple types of FAIR research artefacts.

3.5.1 RO-Crate

RO-Crate [32] is a lightweight and adaptable packaging format for creating Research Objects, that answers the question of how to package a directory with all its material. RO-Crate provides a human- and machine-readable resource that aims to be simple and easy to understand for developers, showcasing a guide regarding best practices. RO-Crate was introduced as a combination of the Research Object with the method of Data Crate[10]. Data Crate defines a standard way of packaging file-based research data for reuse and distribution. It is based on the already existing Bagit[11] packaging specification and has the goal of maximizing the utility and comprehension of data, enabling the discovery of data, and automating the ingest and storing into repositories.

In [32] the RO-Crate community defines the norms that should be followed for the development of RO-Crate and the packaging of FAIR data. These include simplicity, developer friendliness, focus on examples and best practices instead of rigorous specifications, and reuse just enough web standards.

The RO-Crate format relies on JSON-LD with `schema.org`[12] annotations, providing a means for data persistence and ensuring long-term accessibility. Schema.org is a collaborative, community activity with a mission to create, maintain, and promote schemas for structured data on the Internet, on web pages, in email messages, and other online resources. It provides a vocabulary to enhance the research objects with structure metadata and contextual information, covering entities, their relations, and their actions. This structured data markup enhances the presentation of search results and enables various applications to extract and utilize information from web pages in a more organized and consistent manner. RO-Crate follows the Linked Data and FAIR principles to describe the resources, using global identifiers and relationships to link them. The JSON-LD metadata uses linked data to describe data resources, as well as contextual entities like people, organizations, software, and equipment as a series of interconnected JSON-LD objects.

Its structure is composed of a Root Data Entity, which is the directory containing all the material and the RO-Crate Metadata File (`ro-crate-metadata.json`).

[10]`https://github.com/UTS-eResearch/datacrate`
[11]`https://dbpedia.org/page/BagIt`
[12]`https://schema.org/`

This is the main file describing the content and metadata concerning the structure of the directory and the user who created it. The minimal information required by the metadata file are `name`, `description`, `datePublished`, and `license`. Other metadata can be added depending on the goal of the specific RO-Crate.

3.5 displays an example of the RO-Crate metadata file used in the extension developed for the *Ranking Citation tool*. It contains the root directory as well as the output files, which include the data gathered, the authors, and the snapshot of the page.

```
{
    "@context": "https://w3id.org/ro/crate/1.1/context",
    "@graph":    [
            {
                "@id": "./",
                "@type": "Dataset",
                "datePublished": currentYear,
                "hasPart": [
                        { "@id": "output-data.jsonld" },
                        { "@id": "ranking-snapshot-page1.png" }
                    ],
                "author": { "@id": user_id }
            },
            {
                "@id": "ro-crate-metadata.json",
                "@type": "CreativeWork",
                "about": { "@id": "./" },
                "conformsTo": { "@id":"https://w3id.org/ro/crate/1.1"
    }
            },
            {
                "@id": "output-data.json",
                "@type": "File",
                "author": { "@id": user_id },
                "encodingFormat": "application/json",
                "name": "result data"
            },
            {
                "@id": ranking-snapshot-page1.png,
                "@type": {"File", "ImageObject"},
                "author": { "@id": user_id },
                "encodingFormat": "image/png",
                "name": "screenshot"
            },
            {
                "@id": user_id,
                "@type": "Person",
                "name": ZENODO_USER,
                "affiliation": AFFILIATION,
            }
    ]
}
```

Code 3.5: RO-Crate definition

The structure of the JSON object is specified by the RO-Crate, which produces an RDF graph included in a single *'@graph'* array, containing all the entities, contained inside curly brackets ('{ }') in a flat list. Files, folders, people, workflows, images, videos, and other data and metadata related to the research are considered entities, and each one of them is registered with its own corresponding path with respect to the root folder ('./'). Each entity represents the corresponding RDF triples with a subject, property, and object, which can be either literal string values or JSON lists. The `@id` tag is mandatory for every entity, it contains the name or local identifier for the resource. The `@type` is used to describe the type of resources represented by the entity. For example, an image should be assigned the types "File" as well as "ImageObject", whilst the author is stored with the type "Person". The other terms seen in the `@graph` array are taken from the Schema.org vocabulary, such as "name" and "author".

The *'@context'* array is applied to map to Schema.org terms, already defined. More specifically, the `@context` element maps the terms used for the graph entities to their corresponding URIs that define their semantics, and among the most used contexts, we have Schema.org and SKOS. In this way, it is possible to use shorter keys instead of full URIs, making the text more readable and organized.

Some examples of applications of RO-Crate for data, datasets and workflows are: WorkflowHub[13] a FAIR workflow registry that through the use of *Workflow RO-Crates* allows to package an executable workflow with all necessary documentation; Life-Monitor[14] a testing and monitoring service for computa-

[13]https://about.workflowhub.eu/
[14]https://crs4.github.io/life_monitor/

tional workflows; Language Data Commons of Australia Program (LDaCAP)[15] is a project using RO-Crate as an interchange and archive format for language data; and ROHub[16] a platform developed to support the creation, sharing, and management of scientific work and operational processes via Research Objects.

[15]https://www.ldaca.edu.au/

[16]https://www.rohub.org/

4 Methodology

.

4.1 Citation Generation and Persistence

The process of creating the citation involves several phases that include data gathering, creation of the output resources, publication on Zenodo, and the construction of the textual citation.

This chapter will delve into each of these steps in detail.

4.1.1 Data gathering

The first phase in the generation of the citation involves gathering data from the web ranking, which is a fundamental function of the tool. The process begins when the user interacts with the tool while on the search page or result page of one of the supported systems, which is displaying the results obtained from a search query prompted by the user.

The tool uses web-scraping techniques and analyzes the Document Object Model (DOM) of the webpage to capture the necessary data. The DOM is a programming interface for HTML and XML documents, defining the document structure and how it can be managed. Using the DOM it is possible to access and manipulate all the elements of the HTML including IDs, classes, tags, attributes, and styles as well as the elements themselves. The main methods used for this purpose are `getElementById()`, `getElementsByClassName()`, `getElementsByName()`, and `getElementsByTagName()`, which are respectively returning an element with the given id and all the elements with the given class name, name value, and tag name. The first two methods are crucial and largely adopted for the data gathering phase of the tool being presented.

The RDF Graph's basic structure is defined to organize and represent data, the graph is then populated with the collected data. The graph's structure incorporates classes, data properties, and object properties (see figure 4.1).

The model for the *Ranking Citation Ontology* was created with the use of a tool called Protégé[1]. Protégé is an open-source ontology editor that provides a graphical interface for easily creating and editing ontologies stored with one of the supported formats. This software was developed by the Standford University School of Medicine and initially released in 1999, and it offers both desktop and web-based (called WebProtégé) versions. Apart from the Ontology creation and graphical interface, other features provided by Protégé include customization and extension of the tool with plugins, inferencing and reasoning capabilities to detect implicit relationships and logical inconsistencies in the ontology, visualization tools for the ontology structure, and integration with external data sources, databases, and web services. Additionally, it supports OWL (Web Ontology Language) and RDF languages and allows to export the created ontology and to import of external ontologies using standard ontology formats such as Turtle, RDF/XML, and JSON-LD syntax.

The first step in the generation of the model for the ranking citation task is the creation of the classes with Protégé. The classes considered for modeling the citation are the following:

- **Search Query**: specifies the query performed by the user and executed on a specific System. The data collected for the search query comprises the query text, the language in which it is written (actually taken from the navigator language), and most of the search filters applied for that specific query, each system manages the filters for the research in a different way.
- **System**: encapsulates the description of the service adopted for web research. It is limited to the currently supported systems: Google Scholar, Google Search, Bing, Scopus, and Twitter. The data collected for this type of entity is limited to the name of the system and its base URL.
- **Ranking Snapshot**: represents the snapshot of the ranking, encapsulating the properties of date and time at which the citation is captured, and the number of pages captured.

[1]`https://protege.stanford.edu/`

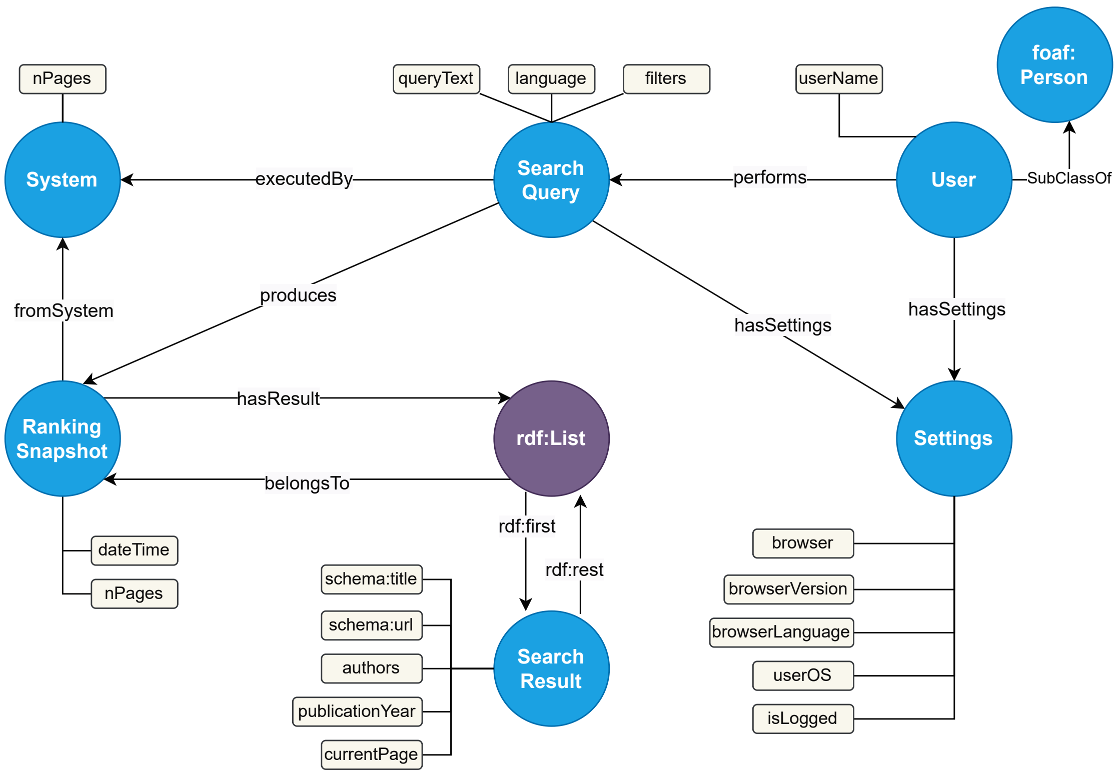


Figure 4.1: A graphical representation of the *Ranking Citation Ontology*

- **Settings**: specifies the general settings of the browser in use. It collects information about the browser's version and language, the user's Operating System, and if the user is logged in.
- **User**: indicates other relevant information about the user consisting of the username and its ORCID.
- **Search Result**: models each result that appears during the research. The properties captured are the result's URL, title, the authors that contributed to it, the publication year or period, and the result page in which it was found.

In addition to the above-mentioned classes there is the `Person` class. This class was taken from the FOAF ontology and is connected with the relationship `subClassOf` with the User class, indicating that the user of the system is an actual person.

In the context of ranking citations, it is essential to keep track of the order in which the results appear. Representing ordered data with RDF language is a non-trivial task but there are a variety of methods dealing with that, including Collections and Containers. RDF Collections are defined linked lists consisting in the chaining of nodes of type `rdf:List`. They are defined by three key

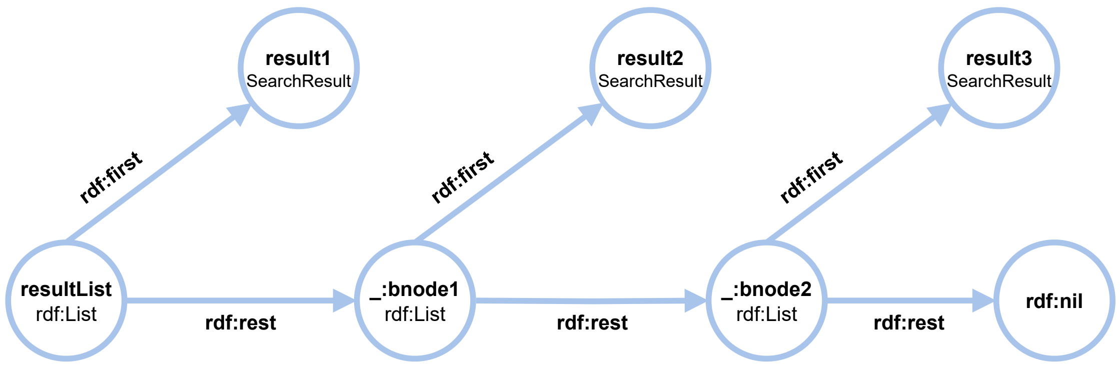


Figure 4.2: A graphical representation of the ordering of the results using RDF Collections

properties: `rdf:first` pointing to the actual item in the list, `rdf:rest` pointing to the next node of the collection, and `rdf:nil` indicating the end of the list. Every node of type `rdf:List` has a `rdf:first` object and is connected to the next node(list) with the `rdf:rest` relationship, until the last node in the ordering which points to the `rdf:nil` node. In the case of the Ranking Citation tool, the items in the list are the instances of the `SearchResult` class. A first instance of the class `rdf:List` is created and it is the one connected to the `RankingSnapshot` class. All the other interconnected nodes of type List are blank nodes, which are only used for ordering reasons and don't have a global scope, and as already explained the last blank node points to the null node representing the ending of the ordering. Figure 4.2 shows how the RDF Collection mechanism is implemented in the developed tool.

Together with the description of the classes above there are listed also the properties relative to the classes themselves. These properties are called Data Properties and are used to enrich the content of each concept with attributes and values relevant to it. The relationships interconnecting classes to data properties or other classes are named Object Properties. The Object and Data Properties used for modeling the *Ranking Citation Ontology* are described in detail with their characteristics in the tables 4.1 and 4.2 respectively. The domain of these properties indicates the class or union of classes to which an instance using the given property is belonging. In the case of Object Properties, the range defines the classes for values that properties can assume. Instead for the Data properties, the type is used to state the specific data type and values that can be assigned to the property.

Object property	**domain**	**range**
belongs to	List	Ranking Snapshot
executed by	Search Query	System
first	List	Search Result
from system	Ranking Snapshot	System
has result	Ranking Snapshot	List
has settings	Search Query or User	Settings
performs	User	Search Query
produces	Search Query	Ranking Snapshot
rest	Search Result	List

Table 4.1: Object Properties

4.1.2 Creation of the output resources

Once the data is captured, it is organized and stored in an RDF graph represented as a JavaScript Object, which is then parsed into a JSON-LD Object. For this purpose, a Javascript array is created and all the data is pushed into it with an automated process that cycles through the ranking results and repeated for all the selected pages. This array is then parsed into JSON-LD with the `Json.stringify()` function. More details about this on the next chapter. The JSON-LD object includes a *@context* element that maps unique long URIs to local attribute names, which are then used in the *@graph* array. The attribute names used are reported in table 4.3, and they enable the integration of well-known ontologies and vocabularies, such as RDF Schema, schema.org, and the FOAF vocabulary among others.

Moreover, the context contains the namespace *'rco'*, which is used to indicate the base URL[2] of the Ontology model: the *Ranking Citation Ontology* model. The URL directs to the web page containing the documentation which provides a clear understanding of all its components, which were described in chapter 4.1.1.

The documentation for the ontology model is generated with Live OWL Documentation Environment (LODE)[3], an online tool that automatically extracts classes, object properties, and data properties along with named individuals, annotation properties, general axioms, and namespace declarations, and renders

[2]https://rankingcitation.dei.unipd.it/ontology/
[3]https://essepuntato.it/lode/

Data property	**domain**	**type**
authors	Search Result	[string]
browser	Settings	string
browser language	Settings	string
browser version	Settings	string
current page	Search Result	int
date time	Ranking Snapshot	string
filters	Search Query	[string]
is logged	Settings	boolean
language	Search Query	string
n pages	Ranking Snapshot	int
name	System	string
publication year	Search Result	gYear
query text	Search Query	string
title	Search Result	string
url	Search Result	any uri
user name	User	string
user OS	Settings	string

Table 4.2: Data Properties

namespace	**url**
foaf	http://xmlns.com/foaf/0.1/
owl	http://www.w3.org/2002/07/owl
rco	https://rankingcitation.dei.unipd.it/ontology/
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns
rdfs	http://www.w3.org/2000/01/rdf-schema
schema	https://schema.org/

Table 4.3: JSON context

them as an ordered list in a human-readable HTML page that can be copied and used freely. To create the documentation with LODE the user simply needs to put the link of the ontology, which has to be publicly available, and after selecting a few optional parameters it is possible to let the service proceed in the automatic extraction and creation of the documentation. By integrating an ontology generated with Protégé with the documentation obtained from LODE it is possible to obtain an optimal human-readable resource that enhances the comprehension of the model, and that can be easily integrated into any web page with few adjustments.

One additional feature provided by the extension is the generation of one or multiple screenshots of the web pages containing the results. This has the objective of creating human-readable resources for the citation object offering a visual representation of the web page at the time of capturing which includes additional resources like images and videos that are not collected with the data for the results but can be helpful for providing context. To protect user privacy some of the user's data displayed by the systems are removed from the generated image. For example, when users are logged into the Google Search system, their current location is displayed at the bottom of the page. The tool contrasts this behavior by simply removing that specific element from the DOM of the web page.

The final resource that is generated by the tool is the metadata for the RO-Crate, which describes the structure of the deposit. It specifies all the names and paths of the screenshots and output data, attributing them to the author, who is stored along with its ORCID. The `ro-crate-metadata.json` file which can be seen in 3.5 is a typical example of the output file describing the organization of the Research Object. More in detail, the `@context` is linking the metadata to the RO-Crate definition on the web at the URL `https://w3id.org/ro/crate/1.1/context`. The `@graph` array is composed of different entities, the first of them has id "./" and indicates the root directory of the object and is enhanced with other metadata such as the date of publication (`"datePublished"`), which is computed during the execution of the code, and the `"hasPart"` attribute indicating the ids of the output files. The second entity refers to the metadata file itself and is linked to its specification on the web. The following entities are the file containing the data collected in the RDF graph and the screenshot. Both of these are enriched with the attributes relative to the author, name, and encoding format namely `"application/json"` and `"image/png"`. The final entity is the user, who is the creator of the citation and is identified through its ORCID, name, and affiliation.

Before the publication of the resources, it is necessary to create the files from the structures defined above. Indeed until this step, all the output files are simply stored as JavaScript arrays/objects in the code, except for the image files. Storing the `ro-crate-metadata.json` and `output-data.jsonld` files requires first parsing them as JSON objects with the function `JSON.stringify()` which converts Javascript values into JSON strings. The parsed files are then saved as Blob objects and subsequently encapsulated into JavaScript *File* objects. This step is essential to avoid the need to download the resources in the user's local

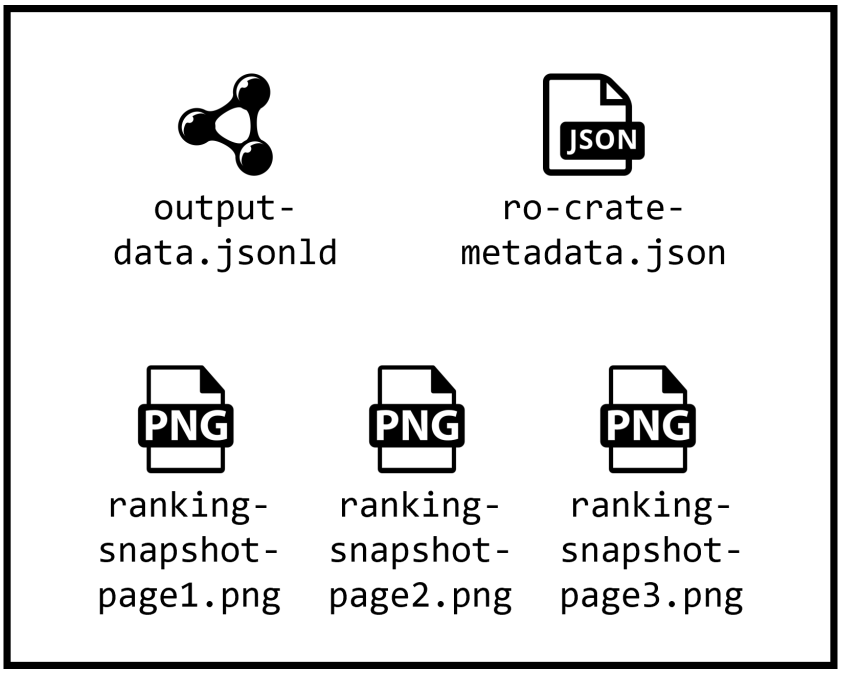

output-data.jsonld
ro-crate-metadata.json
JSON
PNG
ranking-snapshot-page1.png
ranking-snapshot-page2.png
ranking-snapshot-page3.png

Figure 4.3: Graphical representation of an example of the output files contained in the deposit

storage, and this allows to process the files directly in the extension's code to proceed with the publication.

4.1.3 Publication on Zenodo

Once all the output files and resources are generated, the next step is to publish them on Zenodo.

Zenodo [36] is a multi-disciplinary open-access repository developed for the Open Access Infrastructure for Research in Europe (OpenAIRE) program and maintained by CERN. It stores datasets, documents, and other research material, providing a platform for researchers to share, preserve, and cite their scientific outputs while promoting the use of Open Data. Zenodo allows researchers to create their own folders, called deposits, assigning a unique DOI to each of them. These deposits, which can be cited thanks to their DOI, allow for permanent storage of data and are enhanced with a versioning feature to keep track of the changes, which is applied also for reproducibility purposes.

The tool prepares the deposit's metadata, which includes the title, notes, description, keywords, and authors. The title is customized based on the query text and the system used for capturing the web ranking. In addition to this, the description contains details about the Zenodo User that is publishing, including its affiliation, the time and date of the publication, and the number of pages captured, together with the notes that specify the creators of the tool being used. The list of keywords is used to facilitate the search of the deposit created with the tool and includes some predefined keywords, such as the name of the tool (*Unipd Ranking Citation Tool*), and others that can be added by the user. Next, with the use of Zenodo API, the deposit is created, and all the files are uploaded and finally published with their metadata, making them permanent.

Thanks to the DOI assignment, versioning management, metadata support, and principles of open access to data, Zenodo represents an optimal choice for the task of citing web rankings, providing persistence and reproducibility capabilities, in a way that aligns with best practices for data sharing and scholarly communication.

Managing citations and testing with Zenodo Sandbox

As stated previously, once the deposit with the data is published on Zenodo, the published resource is assigned its unique DOI and becomes permanent, making the removal of the deposit virtually impossible. In this context, it is important to consider potential errors related to connection issues, server issues, or user actions while utilizing the extension, and that could prevent the correct publication of the resource. To overcome this challenge, two solutions are implemented.

Before the publication in Zenodo, the tool incorporates a mechanism to generate temporary citations associated with the uploaded deposit, which is not yet published and therefore lacks a DOI. While this temporary citation cannot be used as a formal textual citation, it serves as an intermediate step. Users can interact with the temporary citation, making corrections or additions. Once satisfied they can proceed to publish the deposit and generate the proper citation. Further chapter 5.3.5 describes more in detail the process for generating these temporary citations and their differences from the permanent citations, which can actually be used.

To further facilitate the evaluation and refinement of the extension, users are provided with the option to choose between using Zenodo or its trial version called Zenodo Sandbox. Zenodo Sandbox [37] is a clone of Zenodo, specifically designed to test applications. The testing environment is useful for practicing with the Zenodo interface and API protocols, and to understand the deposition and management capabilities of the platform, before working with the actual research data. Everything on the Sanbox works in the same way as in the standard Zenodo, except that once a resource is published the DOI returned in the response's metadata is a fake one. This means that the user is able to experiment with the tool using Sandbox and generate the citation and the deposit, but the generated citation will not be permanent and therefore not usable for referencing purposes. Preserving a resource on Zenodo Sandbox is not a safe strategy since the sandbox environment can be cleaned at any time and the stored data could be deleted.

The two features specified above enable researchers to effectively manage the publication process, providing flexibility for revisions and a controlled testing environment.

4.1.4 Construction of the textual citation

Upon successful publication on Zenodo, the researcher receives a response back, which contains relevant information for constructing the citation. The response serves as the foundation for constructing the textual citation. It includes the deposit's DOI, a list of the creators/authors, the title of the deposit, and the publication date/period.

The styles and formats used for the text of data citations are a crucial matter in scholarly communication. There are multiple data citation formats to choose from but it is important that the citation includes at least the following information:

- creator(s) and contributor(s)
- date of publication
- title of dataset
- publisher
- identifier (DOI) and/or URL of the resource

The choice of the style may depend on the discipline of the research, examples of the most used formats are displayed in table 4.4. The format for the presented tool was chosen as the most suitable for the web ranking citation task, keeping only the essential information but also the most relevant. Examples of it are shown in the next chapters.

Citation format	**Example**
APA (6th edition)	Smith, T.W., Marsden, P.V., & Hout, M. (2011). General social survey, 1972-2010 cumulative file (ICPSR31521-v1) [data file and codebook]. Chicago, IL: National Opinion Research Center [producer]. Ann Arbor, MI: Inter-university Consortium for Political and Social Research [distributor]. doi: 10.3886/ICPSR31521.v1
MLA (7th edition)	Smith, Tom W., Peter V. Marsden, and Michael Hout. General Social Survey, 1972-2010 Cumulative File. ICPSR31521-v1. Chicago, IL: National Opinion Research Center [producer]. Ann Arbor, MI: Inter-university Consortium for Political and Social Research [distributor], 2011. Web. 23 Jan 2012. doi:10.3886/ICPSR31521.v1
Chicago (16th edition)	Smith, Tom W., Peter V. Marsden, and Michael Hout. 2011. General Social Survey, 1972-2010 Cumulative File. ICPSR31521-v1. Chicago, IL: National Opinion Research Center. Distributed by Ann Arbor, MI: Inter-university Consortium for Political and Social Research. doi:10.3886/ICPSR31521.v1
IEEE	T. W. Smith, P. V. Marsden, and M. Hout, "General Social Survey, 1972-2010 Cumulative File. ICPSR31521-v1. Chicago, IL: National Opinion Research Center. Distributed by Ann Arbor, MI: Inter-university Consortium for Political and Social Research," doi: 10.3886/ICPSR31521.v1.

Table 4.4: Data citation formats

For each displayed citation, the tool provides functionality to copy and remove the citation, although, once published, it cannot be truly removed from the upload destination but only from the extension. The citation built upon these elements ensures that the researcher's work is properly attributed, cited, and referenced.

5
System Design and Development

The model proposed in the preceding chapters has been implemented as a Chrome extension/plugin, to enhance accessibility and facilitate its usage for researchers. The extension is currently available on the dedicated web page `https://rankingcitation.dei.unipd.it/`, which provides detailed instructions on the installation process and how to effectively utilize it.

This chapter will delve into the specific aspects of the extensions' architecture, following the workflow of the tool.

5.1 Overview on Chrome extensions

Google Chrome extensions are programs installed in the Chrome web browser that enhance and extend its functionalities and features. Built upon the same web technologies adopted for the development of web applications, mainly HTML, CSS, and JavaScript languages. Chrome extensions have access to all the JavaScript API available on the web, in addition to Chrome APIs specifically built for extensions. Among them, the chrome.scripting API is used to execute scripts injected in the web pages, the chrome.storage API to store, retrieve and track changes to the user data, and the chrome.tabs API that enables interaction with the Chrome tabs system for creating, modifying, and rearranging tabs. Many of the Chrome APIs use Asynchronous methods that return a Promise once an operation is finished.

The basic structure of each Chrome extension includes the following files:

- The **manifest** (`manifest.json`) file serves as the backbone of the Chrome extension. It is the only required file and it is located in the root directory. Contains the configuration details, such as the extension's name and version, declares the permissions, and identifies the content and background scripts.
- The **service worker** or **background script** handles and listens for browser events, that can run also when the user is not interacting with them. It is supposed to act as a central hub for the extension and is able to make API requests and communicate with the other components.
- **Content scripts** are injected into the web pages, enabling them to edit and manipulate the page's content and behavior. While content scripts have limited access to the Chrome APIs, they can communicate with the service workers.
- The **popup** provides a User Interface (UI) to interact with the extension and access its features. Additional UI components, such as **options pages** and context menus, can extend the functionality of the extension.

Typically, a Chrome extension is developed to accomplish a specific task, it can be composed of multiple modules but it should serve a single purpose.

Developing the proposed tool as a Chrome extension offers the advantage of quick accessibility through its popup, eliminating the need to develop and connect to an external web page.

5.1.1 Message Passing

In the process of the generation of the citations using the Unipd Ranking Citation tool, a significant amount of information and data are exchanged between the scripts injected in the web pages and the background and popup scripts. The communication relies on the message-passing mechanism, provided by Chrome extensions, which enables seamless interaction and data sharing among the different components of the extension.

Message passing consists of a two-sided communication between the extension and content scripts, where each side can listen or send messages to the other end, and respond on the same channel. Two models of communication can be established, simple one-time requests and long-lived requests, with the latter allowing the exchange of multiple messages within the same context.

Simple one-time requests are the primary methods used by the tool, allowing for the transmission of a single message to another script and receiving at most one response back. This is achieved using the `runtime.sendMessage()` and `tabs.sendMessage()` methods. The difference between the two is given by the fact that sending messages to the content script requires knowing the tab where the script is injected. On the other hand, listening to messages can be done with the `runtime.onMessage` event listener. The responses are handled using Promises.

Messages exchanged between the components are JSON Objects of any type. In the context of the Unipd Ranking Citation tool, they are composed of a message/command property that states the intended action, and a payload property that carries the necessary data. This structure for the messages enhances code organization and facilitates error checking, ensuring a robust and manageable communication system.

5.1.2 Asynchronous methods and Promises

The development of Chrome extensions involves the implementation of numerous event listeners and API calls, many of which rely on asynchronous methods.

Asynchronous programming is a JavaScript technique that enables the program to execute time-consuming tasks without blocking the execution of other tasks. Instead of waiting for a task to finish, asynchronous methods enable the code to execute tasks concurrently. Asynchronous programming serves as a replacement for synchronous programming, preventing the browser from becoming unresponsive during long-running operations.

Promises are the foundation for managing asynchronous operations enabling the handling and tracking of their results. Promises are objects returned by asynchronous functions and represent the current state of the operation. They can assume one of three states: Pending, the initial state before the Promise succeeds or fails; Resolved, indicates that a Promise is completed; and Rejected, indicates a failed Promise. Promises allow for chaining operations and handling success or error cases using the `.then()` and `.catch()` methods.

In the context of Chrome extensions, asynchronous methods and Promises are essential for performing background tasks such as handling page events like page loading, tabs creation, and receiving or uploading data from and to external sources.

5.2 The Chrome Extension CLI Framework

By default, Chrome extensions cannot import external JavaScript libraries and Node modules. To address this limitation, there are different solutions and workarounds, the one integrated into this tool is the Chrome Extension Command Line Interface (CLI)[1], a command-line development tool that simplifies the setup, development, and deployment process.

The framework includes the Webpack, a module bundler that is able to generate a single JavaScript file containing all the packages included through the `require()` function, solving the previously mentioned problems. Additionally, The Chrome Extension CLI provides a project template to start developing the extension, defining its structure with essential folders and source files. The "src" folder contains the background, content, and popup scripts, along with the stylesheets for the HTML pages. The "public" folder includes the user-accessible files, such as the HTML files for the popup and the code relative to the options page. It also stores the icons and images' directory and the manifest file for the configuration. Additionally, it provides access to Node and Node Package Manager (NPM) for efficient dependency management.

The Chrome Extension CLI framework allows for a quick and simple development process, offering an automatic reload feature, that ensures that any code change is immediately reflected in the extension. A "build" folder is continuously updated encapsulating all the files that constitute the compiled and packaged version of the extension, ready for deployment.

5.3 Workflow and Architecture

In this section, we will delve deep into the architecture of the Unipd Ranking Citation tool, outlining the various steps involved in the process of generating the citation. Figure 5.1 presents a detailed image of the architecture, illustrating the key steps.

[1]`https://github.com/dutiyesh/chrome-extension-cli`

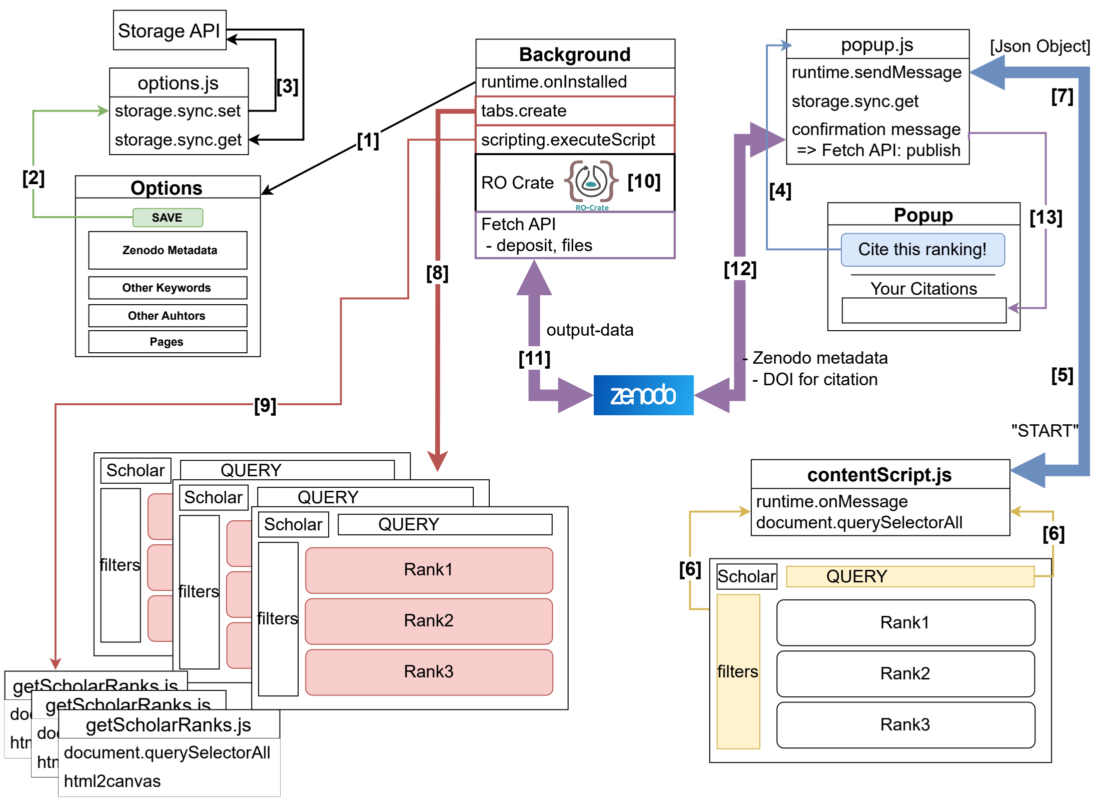
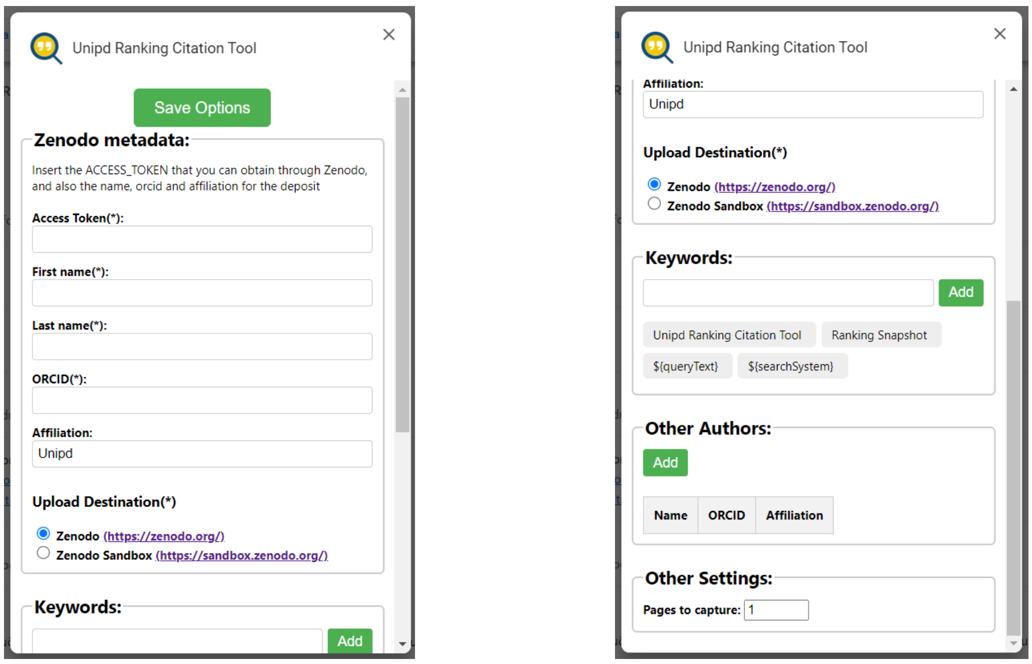


Figure 5.1: The *Unipd Ranking Citation Tool* architecture diagram

5.3.1 Options Menu and Setup

The first phase begins with the installation of the tool on the Chrome web browser. The background script activates the `onInstalled` listener (step [1]), a runtime listener that is fired when the extension is installed, updated, or when the Chrome browser is updated. This listener triggers the `openOptionsPage()` function, a standard function of the extension that directs the user to the options page specified in the manifest file (step[2]).

The options page is utilized as a configuration page for the user's settings and preferences. Figure 5.2 displays all the settings accessible from the options page. The Zenodo metadata section allows users to configure the account details for Zenodo or Zenodo Sandbox, including the username, ORCID, affiliation, and personal access token. The keywords section allows the user to include personalized keywords in the deposit, in addition to the predefined ones. The Other Authors section enables users to specify additional authors or collaborators for attribution of the research, every new author requires the name and ORCID for



Zenodo metadata:

Insert the ACCESS_TOKEN that you can obtain through Zenodo, and also the name, orcid and affiliation for the deposit

Access Token(*):

First name(*):

Last name(*):

ORCID(*):

Affiliation: Unipd

Upload Destination(*)

[✓ Zenodo (https://zenodo.org/)]
Zenodo Sandbox (https://sandbox.zenodo.org/)

Keywords:

Unipd Ranking Citation Tool | Ranking Snapshot | ${queryText} | ${searchSystem}

Other Authors:

Name	ORCID	Affiliation

Other Settings:

Pages to capture: 1

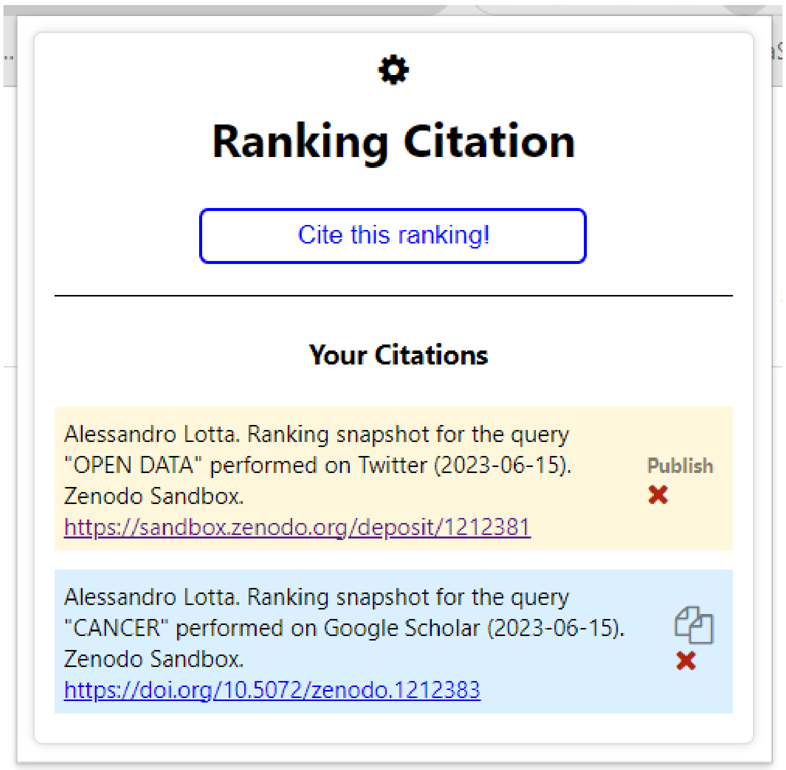
(a) (b)

Figure 5.2: *Unipd Ranking Citation Tool*: Options page.

proper publication. Additionally, the options page presents a section where it is possible to select the desired number of pages to consider for capturing rankings during the research process.

Once the required input fields are filled, users can save the settings by clicking on the dedicated Save button at the top of the page. This action triggers the `saveOptions()` callback function defined in the script, which utilizes the Chrome Storage API (step[3]) to save the data. The Chrome Storage API is divided into four storage areas: local, sync, session, and managed. The local storage stores data locally with a limit of 5MB, and the data is cleared when the extension is removed. The sync storage instead syncs the data into the Chrome browser where the user is logged in if syncing is enabled, otherwise, it works the same as local storage. The Unipd Ranking Citation tool primarily uses sync storage to facilitate usage across different devices. To save the data, the tool Storage API provides the `storage.sync.set()` method, while the `storage.sync.get()` method retrieves the data from the saved keys. This function is used to fill the input fields every time the options page is displayed, with the information defined previously.

The correct execution of the saving process is indicated through a message that disappears after a short time.


Ranking Citation

Cite this ranking!

Your Citations

Alessandro Lotta. Ranking snapshot for the query "OPEN DATA" performed on Twitter (2023-06-15). Zenodo Sandbox. https://sandbox.zenodo.org/deposit/1212381

Publish

Alessandro Lotta. Ranking snapshot for the query "CANCER" performed on Google Scholar (2023-06-15). Zenodo Sandbox. https://doi.org/10.5072/zenodo.1212383

Figure 5.3: The *Unipd Ranking Citation Tool* popup's page

5.3.2 The extension's popup

Once the configuration settings are filled with the required fields, the user is able to access the popup page, which provides a user interface to interact with the extension, accessible by clicking on the extension's icon. The popup content, shown in Figure 5.3, displays the title 'Ranking Citation' along with the options button allowing to access the previously described options page.

Below that, the popup page is divided into two sections. The first section includes the button for capturing the citation, which is available only if the current page matches one of the supported search systems (Google Scholar, Google Search, Bing, Scopus, Twitter). This check is performed using the `match()` method, comparing the Uniform Resource Locator (URL) with predefined regular expressions. If the page is supported, the users can proceed to capture the citation by clicking on the button. Instead, if the page is not supported, a red message is displayed without the button, indicating that the citation cannot be generated on the current page.

The second section of the popup, with the title 'Your Citations', contains a list of all the available citations, including both temporary and proper citations. Temporary citations are displayed as yellow cards, indicating that they are uploaded but not yet published. These citations also include a publish button that once clicked allows the user to definitively publish them. On the other hand, proper citations are represented as blue cards, meaning that they are ready to use, and include a button to copy the text of the citation in the navigator clipboard. Additionally, some control systems are implemented and displayed in the popup's capture button. More in detail, when the user clicks on the Capture button, the button text changes from '*Capture this citation!*' to '*Capturing the citation...*' to provide visual feedback that the process is in progress. Additionally, the button becomes non-clickable to prevent multiple simultaneous capture requests. If any error occurs during the process, the popup is able to capture it and display an error message informing the user about the encountered issue.

The errors captured include upload and publication errors on the upload platform chosen, and the errors coming from background and content script. If any of those occur, it is sufficient to reload the page and try again, otherwise for more complex errors some code corrections may be needed.

5.3.3 Capturing and Processing Data

Clicking the button displayed on the popup starts the process of capturing the data (step[4]).

First, a message is sent from the popup script to the content script, which is injected into the currently visualized page, using the `tabs.sendMessage()` method. This method requires the id of the tab where the content script is injected, which is determined by querying the active pages using the `tabs.query()` method. On the other side, the content script checks for any message containing the '*START*' keyword (step [5]), using the `runtime.onMessage` event listener. In this stage, the content script captures the data from the web page (step[6]), following the process specified in chapter 4.1.1, with the exception of collecting the ranking results.

The specific techniques for acquiring information may vary depending on the search system in use, which will be discussed in later chapters. Once the data is successfully captured, the content script sends a response message back to the popup, containing the RDF graph stored as a JSON Object (step[7]). The popup script directly forwards the response's data to the background script using a one-time request including it in the payload along with the command keyword *'CREATE RO'*.

After receiving the message, the background script executes the `getPagesRanks()` function. This function first opens the new pages where the ranking results will be captured, using the `tabs.create()` method (step[8]). The number of new pages created depends on the value specified in the options page.

In this step it is important to note two aspects: one is that the user is able to specify filters during the research, affecting the rankings of the results. These filters are transported to the newly opened pages, ensuring consistency. The second aspect to take into account is that a new script is injected into each of the newly opened pages (step[9]). This is achieved with the `scripting.executeScript()` method, which requires, as a parameter, the tab where each script needs to be injected. Each injected script has access to the DOM content and accomplishes the task of capturing the remaining data, consisting of the actual ranking of results.

Once the injected scripts finish collecting the ranking data, they each send a message to the background script containing the collected data in the payload. The background script waits until all the scripts accomplish their task before proceeding with the next steps in the citation generation process.

5.3.4 Upload Phase and Data preparation

The next stage is the upload phase, initiated when calling the `uploadData()` function. In this step, the information provided on the options page is retrieved using the `storage.sync.get()` method. Providing the predefined keys it is possible to access the personal access token, name of the Zenodo user, affiliation, ORCID, keywords, preferred upload destination, additional authors, and the number of pages to capture.

The next step (step[10]) involves creating the Research Object Crate (RO-Crate) following the process described in section 4.1.2. The RO-Crate defines the structure of the output files and provides attribution to the authors specified in the options. The gathered data is then converted into JSON, with the `JSON.stringify()` function, stored into a Blob object, and then converted into a JavaScript File variable.

The output files are sent to the server using the JavaScript Fetch API, which provides access and manipulation of parts of the communication protocol. The `fetch()` function is used to connect to the Zenodo REST API, which allows the uploading and publishing of research outputs, retrieving published records, and downloading/uploading of files. The first fetch call generates the deposit and returns the ID for the deposit. Subsequent calls are made to upload the 'output-data.jsonld' file, containing the collected data for the RDF graph, and the 'ro-crate-metadata.json' file (step[11]). The Fetch API is Promise-based and resolves in a Response object representing the response to the request that is sent. The `fetch()` promise only rejects in case of network errors.

The tool handles various errors that can occur (see table 5.1), including HTTP errors, and sends them to the popup script using a one-time request with the keyword '*ERROR*'. The popup script then displays them through alerts and by changing the capture button accordingly.

If the upload of the first two resources has been successful, the background script sends a message to the injected pages to initiate the process of capturing screenshots for each page. Each injected page includes a 'runtime.onMessage' listener that waits for the keyword '*ADD SCREENSHOT*' along with the page number. Upon receiving this message, the html2canvas method, from the homonym library, is invoked.

The `html2canvas` method[2], combined with the `HTMLCanvasElement.toBlob()` method is able to render the body element of the HTML page as a canvas image. This image is then converted into a Blob Object and further transformed into a JavaScript File variable. The file is then uploaded using the fetch() call, with the deposit ID and access token passed as the payload from the background script.

Error	**Name**
400	Bad Request
401	Unauthorized
403	Forbidden
404	Not Found
405	Method Not Allowed
409	Conflict
415	Unsupported Media Type
429	Too Many Requests
500	Internal Server Error

Table 5.1: Uploading Errors

[2]`https://html2canvas.hertzen.com/`

5.3.5 Publication and Generation of the Citation

The final phase starts once all the screenshots have been successfully uploaded to the deposit in Zenodo or in Zenodo Sandbox. The popup script is notified with a simple one-time request containing as payload the deposit ID, access token, and upload destination, necessary for the publication. To ensure the user's intention to proceed with the publication, a confirmation message is displayed on the popup page.

If the user agrees to publish, a fetch call is made to post the deposit and create the permanent citation (step[12]). The text of the citation is generated using the information contained in the response of the fetch call, including the deposit DOI, creators, title, and publication date. Additionally, it is specified the upload destination and the link to the DOI of the resources. The `updateCitation()` function is called to generate the blue card for the permanent citation. This card also contains a button to remove the citation from the extension and a button to copy the text of the citation into the clipboard. The card is then inserted in the 'Your Citations' section, which displays the citations captured previously (step[13]).

If the user chooses not to publish the deposit, the popup script generates the temporary citation. This citation appears as a yellow card with a remove button and a button that calls the `publishTempCitation()` function, which handles the publishing at a later moment. The text of the temporary citation is the same as the text of a proper citation, except for the link that contains the deposit id instead of the DOI for the resources.

The `storage.sync.set()` method is used to store the citations (both temporary and permanent) with their respective deposit ID. Then the `storage.sync.get()` method allows retrieving each saved citation and displaying that on the popup. The `storage.sync.remove(key)` method, triggered with the remove button, is used to delete the citation with the specified key. This action doesn't actually delete the deposit from the upload destination, but only the citation reference from the extension.

5.4 Considerations on the supported platform

During the development of the *Unipd Ranking Citation tool*, the systems chosen to support the citation of web rankings were Google Scholar, Google Search, Scopus, Bing, and Twitter. These platforms were selected for their popularity and to provide accessibility both for academic researchers, students, and professionals, but also for general users seeking citation-related information.

A key aspect in the tool's development was the reliance on data extracted from the DOM components of each platform. As web pages' structure frequently changes, it is essential to maintain and update the tool to ensure compatibility with the evolving platforms.

To adapt the tool to each supported system, specific code sections were implemented to handle the different filters available on each platform (see table 5.2). The filters allow users to customize the search criteria to find specific results. However, the considered systems are provided with different levels of filtering. For instance, Google Scholar and Scopus provide more advanced filtering options, while Google Search and Bing have a more limited selection.

System	**Filters captured**
Google Scholar	include patents since/until year sort by relevance/date result type (any/review articles)
Google Search and Bing	from past hour/day/week/month/year since/until date
Scopus	open/limited/closed access from/to year exclude/only publication year
Twitter	display top/latest/people/images/videos exclude/only replies and links since/until date

Table 5.2: Available filters for each system

Furthermore, each supported platform requires specific code to capture the ranking results. The background script is able to check the name of the system in use and select the correct scripts to inject into the page (selecting one from `getScholarRanks.js`, `getGoogleRanks.js`, `getScopusRanks.js`, `getBingRanks.js`, or `getTwitterRanks.js`). While the structure of the scripts remains consistent, the methods for extracting elements from the DOM differ depending on the platform. For instance, the script for Twitter doesn't allow capturing multiple pages since Twitter's feed can only be viewed by scrolling the page, meaning that the number of results captured is limited compared to the other systems.

6

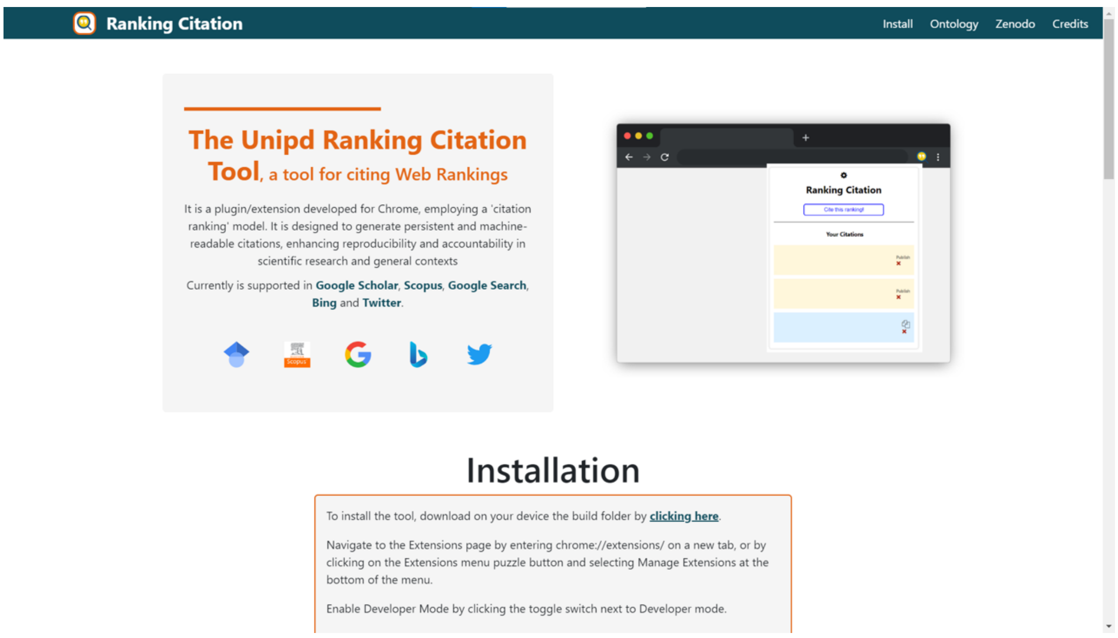
Use Case and Applications

The following section presents a practical example of the application of the *Unipd Ranking Citation Tool*, providing a step-by-step explanation of the process along with screenshots for reference.

To begin, the user must install the extension on the Google Chrome browser. The page located at the URL
`https://rankingcitation.dei.unipd.it/` is a dedicated web page built with HTML, CSS, and JavaScript, to present the tool (Figure 6.1). The page offers a description of the main features, a link to the Ontology documentation, and a link to the Zenodo landing page. Moreover, the page contains an installation guide and a brief "Get Started" guide that explains how to use the extension.

The installation of the tool is accomplished by following the provided link, which directs to a shared Google Drive folder[1], which provides only visualization (and download) access. The folder contains the build version of the extension that can be downloaded on the local machine in the compressed zip format. After extracting the folder, it is possible to navigate to the Chrome Extension page that can be reached by either entering the URL `chrome://extensions` on a new tab or by clicking on the Extensions menu puzzle button and selecting the "Manage Extensions" option at the bottom of the menu. The user must enable the developer mode, on the top right of the page, and click on the "Load unpacked extension" that opens a window to select the extension directory from

[1]`https://drive.google.com/drive/folders/1UV2INVtBfrkVs6UnNgkcQZ7cgYCNhIUe`



The Unipd Ranking Citation Tool, a tool for citing Web Rankings

It is a plugin/extension developed for Chrome, employing a 'citation ranking' model. It is designed to generate persistent and machine-readable citations, enhancing reproducibility and accountability in scientific research and general contexts

Currently is supported in **Google Scholar**, **Scopus**, **Google Search**, **Bing** and **Twitter**.

Installation

To install the tool, download on your device the build folder by clicking here.

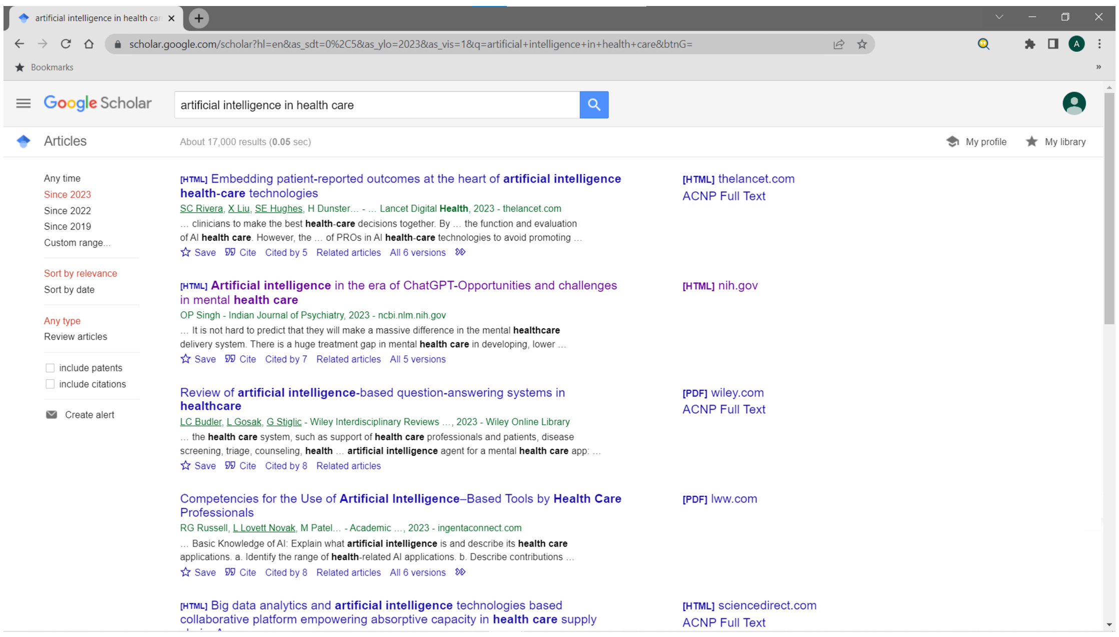
Navigate to the Extensions page by entering chrome://extensions/ on a new tab, or by clicking on the Extensions menu puzzle button and selecting Manage Extensions at the bottom of the menu.

Enable Developer Mode by clicking the toggle switch next to Developer mode.

Figure 6.1: Image of the web page for the *Unipd Ranking Citation Tool*, available at `https://rankingcitation.dei.unipd.it/`

the previously downloaded folder. It is important to note that the correct extension directory is the one containing the `manifest.json` file. At this point, the extension should be correctly installed.

As soon as the installation is completed, the extension's options page is presented, as shown in Figure 5.2. As explained in chapter 5, all the information required on this page must be provided to use the extension. The access token can be obtained from one of the two upload destinations, Zenodo or Zenodo Sandbox. The two platforms do not share the same information, since Sandbox is only a trial version, meaning that a user who wants to publish on both of them, will need two create two accounts. To produce the access token for uploading and publishing the deposits, the user needs to navigate to the "Applications" page of the personal account and from there, generate a new token with `deposit:actions` and `deposit:write` scopes. The alphanumeric string that is generated has to be copied to the options page. The First and Last names fields, together with the ORCID field should be filled with the information on the user that owns the account in the upload destination chosen. The ORCID is necessary for proper attribution of the deposit in Zenodo, and if the user does not have one, it must be created to interact with the extension. Below the fields for inserting the affiliation and choosing the upload destination, there's


artificial intelligence in health car
scholar.google.com/scholar?hl=en&as_sdt=0%2C5&as_ylo=2023&as_vis=1&q=artificial+intelligence+in+health+care&btnG=
Bookmarks
Google Scholar
Articles
My profile
My library
Save
Cite
Related articles

artificial intelligence in health care

About 17,000 results (0.05 sec)

Any time
Since 2023
Since 2022
Since 2019
Custom range...

Sort by relevance
Sort by date

Any type
Review articles

include patents
include citations

Create alert

[HTML] Embedding patient-reported outcomes at the heart of artificial intelligence health-care technologies — [HTML] thelancet.com, ACNP Full Text
SC Rivera, X Liu, SE Hughes, H Dunster... - ... Lancet Digital Health, 2023 - thelancet.com
... clinicians to make the best health-care decisions together. By ... the function and evaluation of AI health care. However, the ... of PROs in AI health-care technologies to avoid promoting ...
Save Cite Cited by 5 Related articles All 6 versions

[HTML] Artificial intelligence in the era of ChatGPT-Opportunities and challenges in mental health care — [HTML] nih.gov
OP Singh - Indian Journal of Psychiatry, 2023 - ncbi.nlm.nih.gov
... It is not hard to predict that they will make a massive difference in the mental healthcare delivery system. There is a huge treatment gap in mental health care in developing, lower ...
Save Cite Cited by 7 Related articles All 5 versions

Review of artificial intelligence-based question-answering systems in healthcare — [PDF] wiley.com, ACNP Full Text
LC Budler, L Gosak, G Stiglic - Wiley Interdisciplinary Reviews ..., 2023 - Wiley Online Library
... the health care system, such as support of health care professionals and patients, disease screening, triage, counseling, health ... artificial intelligence agent for a mental health care app: ...
Save Cite Cited by 8 Related articles

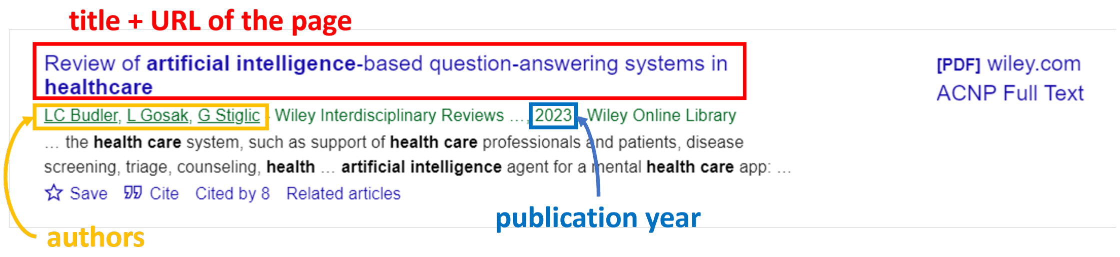
Competencies for the Use of Artificial Intelligence–Based Tools by Health Care Professionals — [PDF] lww.com
RG Russell, L Lovett Novak, M Patel... - Academic ..., 2023 - ingentaconnect.com
... Basic Knowledge of AI: Explain what artificial intelligence is and describe its health care applications. a. Identify the range of health-related AI applications. b. Describe contributions ...
Save Cite Cited by 8 Related articles All 6 versions

[HTML] Big data analytics and artificial intelligence technologies based collaborative platform empowering absorptive capacity in health care supply — [HTML] sciencedirect.com, ACNP Full Text

Figure 6.2: Screenshot of the ranking results obtained from an example of a search query on the Google Scholar systems

the keywords section. The options menu enables users to add and subsequently remove keywords in addition to the default ones. Keywords can be added to help in the traceability of the cited resources stored in Zenodo and enhance the discoverability of related research. In the "Other Authors" section the users can add collaborators. By clicking on the 'add' button, a new form is revealed to input the name, ORCID, and affiliation of the new author, who can then be inserted into the list of collaborators displayed underneath. The final field is for specifying the number of pages to capture, with a default value of 1. The input field allows selecting a range from 1 to 10 pages. Adding more pages is useful to integrate additional information but increases the size of deposits, slowing the capturing process and possibly creating confusion in the produced output.

At this stage, the user is able to interact with the features provided by the extension and perform a query search on one of the supported systems. For demonstration purposes, Google Scholar is chosen due to its advanced filters and since it is able to provide us with a comprehensive overview of the process and results. The upload destination selected is Zenodo Sandbox since it's a demonstration platform. An example query is "Artificial intelligence in health care" which yields a substantial number of search results that are frequently



Review of **artificial intelligence**-based question-answering systems in **healthcare**

LC Budler, L Gosak, G Stiglic - Wiley Interdisciplinary Reviews ..., 2023 - Wiley Online Library

... the **health care** system, such as support of **health care** professionals and patients, disease screening, triage, counseling, **health** ... **artificial intelligence** agent for a mental **health care** app: ...

☆ Save 99 Cite Cited by 8 Related articles

[PDF] wiley.com
ACNP Full Text

Figure 6.3: Representation of the information captured from the third-ranked result of the example.

updated (Figure 6.2).

To refine the results the user can select the filters on the left-side panel of the page. As stated previously, the Google Scholar platform implements filters for custom ranges of time, for result sorting and type, and to include patents and citations. In this example, we limit the results to the ones published since 2023, sort by relevance, and exclude both patents and citations.

Once the user is satisfied with the displayed results and has chosen the number of pages to capture in the extension's options, they can proceed to cite the current web ranking. By clicking on the extension's icon (a blue magnifying lens with white citation marks on a yellow background), the popup page is activated, displaying the "Cite this ranking!" button and the "Your citations" section with previously captured citations, both permanent (blue cards) and temporary (yellow cards). Clicking the blue button starts the process of capturing the data and generating the citation. The button turns into a non-clickable element, changing its color to orange and the text to "Capturing the citation..." to indicate that the task is running in the background. The steps described in chapter 5.3.3 are executed at this stage.

Figure 6.3 shows an example of what data is captured in one of the results. The data captured for each result comprise the title, which additionally encapsulates the link to the web address where the resource can be viewed, the authors involved and the publication years are the ones displayed on the second row of the result. The information for the current page number is computed from the Scholar page URL and is not obtainable from the result itself.

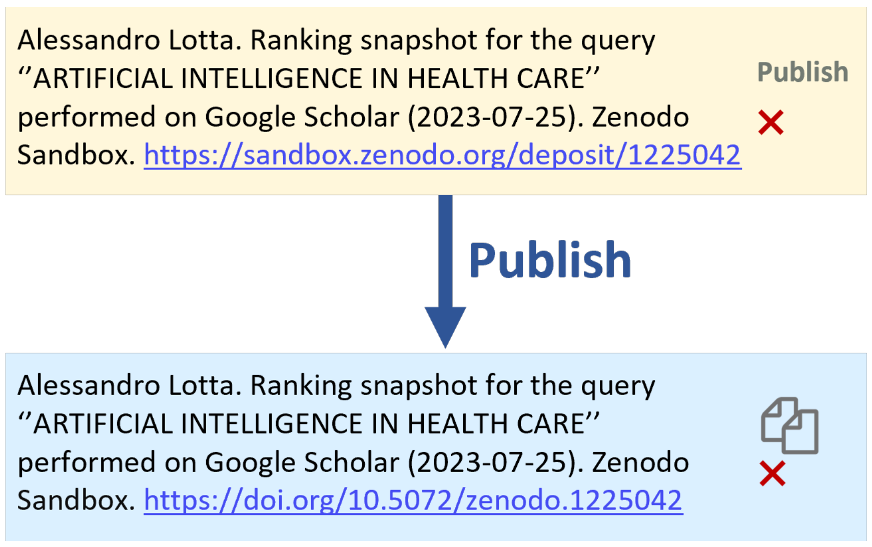
Since the result considered is the third one in the ranking, it is connected to the second through a blank node of type rdf:List, as shown in 6.1. The values of the "@id" elements correspond to the URLs of the second and third results in the ranking respectively.

```
{
      "@id": "_:bnode1",
      "rdf:first": {
        "@id": "https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10187878
   /"
      },
      "rdf:rest": {
        "@id": "_:bnode2"
      }
},
{
      "@id": "_:bnode2",
      "rdf:first": {
        "@id": "https://wires.onlinelibrary.wiley.com/doi/abs
   /10.1002/widm.1487"
      },
      "rdf:rest": {
        "@id": "_:bnode3"
      }
},
```

Code 6.1: Ordering of the ranking given by the blank nodes

After a small amount of time elapses to gather the data and capture the screenshots of the pages, the extension's popup will display the confirmation message, if no error occurred during the process. This message indicates that the deposit and output files have been successfully uploaded and are ready to be published. For demonstration purposes, the direct publication in Zenodo Sandbox is not confirmed, so a temporary citation is generated (yellow card in Figure 6.4). The temporary citation presents a first version of the textual citation containing the authors' names, search query, system, date of capture, and upload destination as in the final citation. However, it does not include the DOI of the deposit since it is generated only after publication.

At this point, the user can check the deposit just created by navigating to Zenodo and accessing the personal "Upload" section, where all stored files can be viewed and edited. If the user is satisfied with the content, the next step is to publish the deposit by clicking the dedicated button on the right side of the citation card. Once the publication process is complete (which will take a few seconds), the temporary citation becomes a permanent citation (blue card in Figure 6.4), and the DOI of the resource is added to the text, incorporating a link that directs to the deposit's page (Figure 6.5).

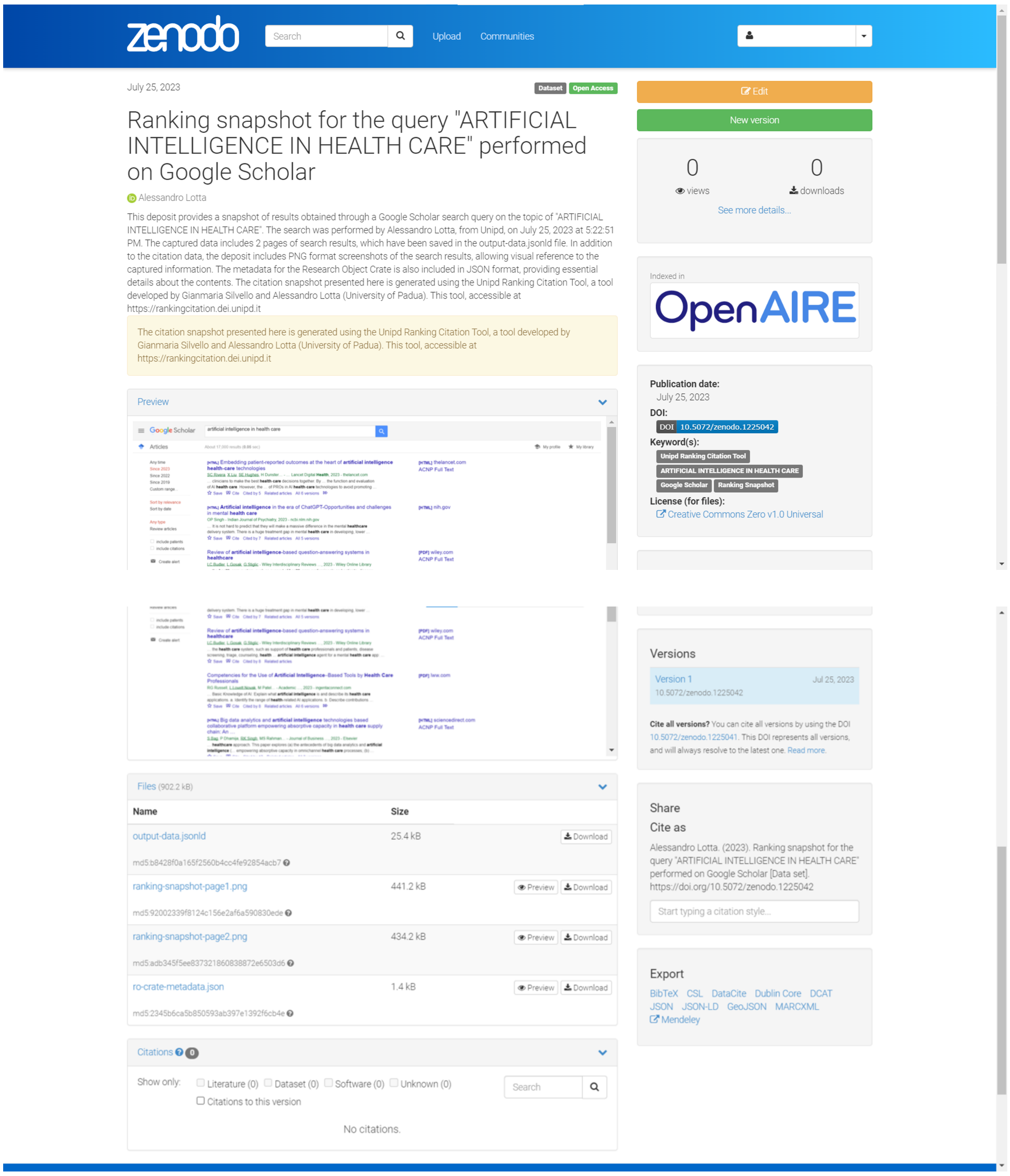

Alessandro Lotta. Ranking snapshot for the query ''ARTIFICIAL INTELLIGENCE IN HEALTH CARE'' performed on Google Scholar (2023-07-25). Zenodo Sandbox. https://sandbox.zenodo.org/deposit/1225042

Alessandro Lotta. Ranking snapshot for the query ''ARTIFICIAL INTELLIGENCE IN HEALTH CARE'' performed on Google Scholar (2023-07-25). Zenodo Sandbox. https://doi.org/10.5072/zenodo.1225042

Figure 6.4: Representations of the temporary citation and permanent citation generated with the example

The entire process for capturing the web ranking is now complete, and the citation can be copied and utilized for referencing the output data for research purposes.

An example of effective use of this citation is to cite the ranking to compare the results obtained at different points in time, evaluating the evolution of AI techniques in the health care field.



July 25, 2023

Dataset | Open Access

Ranking snapshot for the query "ARTIFICIAL INTELLIGENCE IN HEALTH CARE" performed on Google Scholar

Alessandro Lotta

This deposit provides a snapshot of results obtained through a Google Scholar search query on the topic of "ARTIFICIAL INTELLIGENCE IN HEALTH CARE". The search was performed by Alessandro Lotta, from Unipd, on July 25, 2023 at 5:22:51 PM. The captured data includes 2 pages of search results, which have been saved in the output-data.jsonld file. In addition to the citation data, the deposit includes PNG format screenshots of the search results, allowing visual reference to the captured information. The metadata for the Research Object Crate is also included in JSON format, providing essential details about the contents. The citation snapshot presented here is generated using the Unipd Ranking Citation Tool, a tool developed by Gianmaria Silvello and Alessandro Lotta (University of Padua). This tool, accessible at https://rankingcitation.dei.unipd.it

The citation snapshot presented here is generated using the Unipd Ranking Citation Tool, a tool developed by Gianmaria Silvello and Alessandro Lotta (University of Padua). This tool, accessible at https://rankingcitation.dei.unipd.it

Preview

Files (902.2 kB)

Name	Size	
output-data.jsonld md5:b8428f0a165f2560b4cc4fe92854acb7	25.4 kB	Download
ranking-snapshot-page1.png md5:92002339f8124c156e2af6a590830ede	441.2 kB	Preview Download
ranking-snapshot-page2.png md5:adb345f5ee837321860838872e6503d6	434.2 kB	Preview Download
ro-crate-metadata.json md5:2345b6ca5b850593ab397e1392f6cb4e	1.4 kB	Preview Download

Citations 0

Show only: Literature (0) Dataset (0) Software (0) Unknown (0) Citations to this version

No citations.

0 views 0 downloads

See more details...

Indexed in OpenAIRE

Publication date: July 25, 2023

DOI: DOI 10.5072/zenodo.1225042

Keyword(s): Unipd Ranking Citation Tool | ARTIFICIAL INTELLIGENCE IN HEALTH CARE | Google Scholar | Ranking Snapshot

License (for files): Creative Commons Zero v1.0 Universal

Versions

Version 1 Jul 25, 2023
10.5072/zenodo.1225042

Cite all versions? You can cite all versions by using the DOI 10.5072/zenodo.1225041. This DOI represents all versions, and will always resolve to the latest one. Read more.

Share

Cite as

Alessandro Lotta. (2023). Ranking snapshot for the query "ARTIFICIAL INTELLIGENCE IN HEALTH CARE" performed on Google Scholar [Data set]. https://doi.org/10.5072/zenodo.1225042

Export

BibTeX CSL DataCite Dublin Core DCAT JSON JSON-LD GeoJSON MARCXML Mendeley

Figure 6.5: Screenshots of the final page of the deposit published on Zenodo Sandbox. In the example, the number of captured pages is 2

7 Conclusions and Future Works

The work presented in this thesis is a contribution to the data citation practices and implementations for scholarly communication. The "Unipd Ranking Citation tool" targets the emerging field of reproducible information retrieval (IR) rankings, focusing on the rankings produced by various search systems, namely web rankings. The tool provides a way to gather the information from the web rankings making them persistent and accessible, and generating a citation for the collected data, which can be used as a reference in scientific publications and business reports. Citing reproducible web rankings has applications both in the research field ensuring credit to the authors and verifiability of the resources, and in the business field where it is used to justify decision-making processes and monitor changes in the companies' rankings and social media tendencies.

The tool is able to overcome most of the problems related to data citation, indeed it provides an intuitive Chrome extension that is able to automatically capture the necessary data and generate the citation requiring only a starting input command and setup configuration from the user. Concerns related to fixity and versioning of the cited resources are solved thanks to the integration of the Zenodo platform which ensures persistency and long-term accessibility to the published repositories, assigning persistent identifiers (with the DOI system) to each version of the deposit and to the deposit itself. This also makes the resources reproducible in case of any update or deletion of files inside the deposit. The "Unipd Ranking Citation tool" is adapting to different systems and platforms, it currently operates in five different systems namely Google

Scholar, Google Search, Scopus Elsevier, Bing, and Twitter. It also adheres to the Linked Open Data principles by using unique identifiers in the form of URIs. This enables interoperability and integration of the output resources with other resources available on the web.

The extension has been presented by first giving an in-depth explanation of its features and components that are integrated, including Research Objects, RDF graphs, the Zenodo API, and the JSON-LD language. Subsequently, there has been an in-depth explanation of the whole process for generating the citation and delving into the details concerning the architecture of the developed extension. Finally, to demonstrate the extension's functionalities an example of a search query performed on the Scholar system is given.

As for now, the tool operates correctly for most of the queries performed in the listed search systems, but future development could help in improving its stability to errors which sometimes are caused by certain types of data or the advent of specific conditions. These limitations are due to the fact that the tool relies on parsing the DOM of web pages which could change over time. For these reasons, the tool will need to be updated and its capabilities could also be expanded to support other platforms employed by the users.

The following steps are to adopt the tool in different fields and to acquire more data so that it can be compared and queried using the SPARQL language with the triples produced in the output files. The growing popularity of AI systems and chatbots could benefit from the integration of reproducible rankings. Among other functions they are provided, some of them such as Google's Bard have the capability to report the best results available on the web upon a request from the user. Reproducible rankings can help Bard to rank the results that it generates by providing it with a set of criteria that can be used to assess the quality of the results, making them more accurate and reliable. In addition, the citation of web rankings can be useful in the learning process of AI and chatbots, because they can help to learn about the factors that are important for ranking results.

References

[1] Micah Altman et al. “OUT OF CITE, OUT OF MIND: THE CURRENT STATE OF PRACTICE, POLICY, AND TECHNOLOGY FOR THE CITATION OF DATA”. In: *Data Science Journal* 12 (Dec. 2013), pp. 1–75.

[2] Anita Bandrowski et al. “The Resource Identification Initiative: a cultural shift in publishing”. In: *Brain and Behavior* 6 (Dec. 2015), n/a–n/a. DOI: 10.1002/brb3.417.

[3] Christine Borgman. “Data Citation as a Bibliometric Oxymoron”. In: Feb. 2016, pp. 93–116. ISBN: 9783110308464. DOI: 10.1515/9783110308464-008.

[4] Peter Buneman, Susan Davidson, and James Frew. “Why Data Citation Is a Computational Problem”. In: *Communications of the ACM* 59 (Sept. 2016), pp. 50–57. DOI: 10.1145/2893181.

[5] Peter Buneman and Gianmaria Silvello. “A Rule-Based Citation System for Structured and Evolving”. In: *IEEE Data Eng. Bull.* 33 (Jan. 2010), pp. 33–41.

[6] Helena Cousijn et al. “A Data Citation Roadmap for Scientific Publishers”. In: *bioRXiv* 5 (Nov. 2018). DOI: 10.1038/sdata.2018.259.

[7] Helena Cousijn et al. “Bringing Citations and Usage Metrics Together to Make Data Count”. In: *Data Science Journal* 18 (Mar. 2019). DOI: 10.5334/dsj-2019-009.

[8] Merce Crosas. “The Evolution of Data Citation: From Principles to Implementation”. In: *IASSIST Quarterly*. Vol. 37. May 2014, p. 62. DOI: 10.29173/iq504.

[9] Merce Crosas et al. “Automating Open Science for Big Data”. In: *The ANNALS of the American Academy of Political and Social Science* 659 (Apr. 2015), pp. 260–273. DOI: 10.1177/0002716215570847.

[10] *Data Citation - USGS*. URL: https://www.usgs.gov/data-management/data-citation.

[11] *Data Citation Awareness - ANDS*. URL: http://ands.org.au/guides/data-citation-awareness.html.

[12] Erika Fabris, Tobias Kuhn, and Gianmaria Silvello. "A Framework for Citing Nanopublications". In: Aug. 2019, pp. 70–83. ISBN: 978-3-030-30759-2. DOI: 10.1007/978-3-030-30760-8_6.

[13] Martin Fenner et al. "Code of practice for research data usage metrics release 1". In: (Feb. 2018). DOI: 10.7287/peerj.preprints.26505.

[14] Paul Groth, Andrew Gibson, and Johannes Velterop. "The Anatomy of a Nano-publication". In: *Information Services and Use* 30 (Sept. 2010). DOI: 10.3233/ISU-2010-0613.

[15] *Introducing JSON*. URL: https://www.json.org/json-en.html.

[16] *JSON-LD 1.1*. URL: https://www.w3.org/TR/json-ld/.

[17] John E Kratz and Carly Strasser. "Researcher perspectives on publication and peer review of data". In: *PloS one* 10.2 (2015), e0117619. DOI: 10.1371/journal.pone.0117619.

[18] "Data Citation Synthesis Group: Joint Declaration of Data Citation Principles". In: ed. by Maryann Martone. San Diego, CA: FORCE11, 2014. DOI: https://doi.org/10.25490/a97f-egyk.

[19] Brian Nosek et al. "Promoting an Open Research Culture". In: *Science (New York, N.Y.)* 348 (June 2015), pp. 1422–5. DOI: 10.1126/science.aab2374.

[20] *Ontology*. URL: https://en.wikipedia.org/wiki/Ontology.

[21] *Ontology definition 2007*. URL: http://web.dfc.unibo.it/buzzetti/IUcorso2007-08/mdidattici/ontology-definition-2007.htm.

[22] Mark Parsons, Ruth Duerr, and Matthew Jones. "The History and Future of Data Citation in Practice". In: vol. 18. Nov. 2019. DOI: 10.5334/dsj-2019-052.

[23] Stefan Pröll and Andreas Rauber. "Scalable data citation in dynamic, large databases: Model and reference implementation". In: *2013 IEEE International Conference on Big Data*. 2013, pp. 307–312. DOI: 10.1109/BigData.2013.6691588.

[24] A. Rauber and M. Parsons. *Data Citation Working Group Mtg @ P19*. URL: `https://www.rd-alliance.org/system/files/documents/220623_rda_p19_wgdc_slides.pdf`, slide 52. June 2022.

[25] Andreas Rauber et al. *Data Citation of Evolving Data: Recommendations of the Working Group on Data Citation (WGDC)*. Oct. 2015. DOI: `10.15497/RDA00016`. URL: `https://doi.org/10.15497/RDA00016`.

[26] Andreas Rauber et al. "Identification of Reproducible Subsets for Data Citation, Sharing and Re-Use". In: *Bulletin of the IEEE Technical Committe on Digital Libraries* 12.1 (May 2016). DOI: `10.5281/zenodo.4048304`. URL: `https://doi.org/10.5281/zenodo.4048304`.

[27] *RDF - Semantic Web Standards - W3C*. URL: `https://www.w3.org/RDF/`.

[28] *Research Object*. URL: `https://www.researchobject.org/`.

[29] *ROSC Community Group Charter*. URL: `https://www.w3.org/community/rosc/rosc-community-group-charter/`.

[30] Gianmaria Silvello. "A Methodology for Citing Linked Open Data Subsets". In: Sept. 2014. DOI: `10.13140/2.1.2806.0489`.

[31] Gianmaria Silvello. "Theory and Practice of Data Citation". In: *Journal of the Association for Information Science and Technology* 69 (Jan. 2018), pp. 6–20. DOI: `10.1002/asi.23917`.

[32] Stian Soiland-Reyes et al. "Packaging research artefacts with RO-Crate". In: *Data Science* 5 (Jan. 2022), pp. 1–42. DOI: `10.3233/DS-210053`.

[33] *Whole Tale project - About*. URL: `https://wholetale.readthedocs.io/en/stable/README.html`.

[34] Richard Wiggins et al. "Image File Formats: Past, Present, and Future1". In: *Radiographics: a review publication of the Radiological Society of North America, Inc* 21 (Nov. 2000), pp. 789–98. DOI: `10.1148/radiographics.21.3.g01ma25789`.

[35] Mark Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship". In: *Scientific Data* 3 (Mar. 2016). DOI: `10.1038/sdata.2016.18`.

[36] *Zenodo*. URL: `https://zenodo.org/`.

[37] *Zenodo Sandbox*. URL: `https://sandbox.zenodo.org/`.

Acknowledgments

First of all, I would like to thank Professor Gianmaria Silvello who was the Supervisor for my thesis. Thanks for supporting and helping me in completing the project, which was part of the Research Training activity for my Master's Degree, and also for letting me contribute to the research paper derived from this project.

I would also like to thank my parents, brother, and sister for supporting me in achieving my goals during these years of study at the University of Padova, especially my mother and father for financially sustaining my studies and for helping me in making important decisions.

Finally, I want to thank my friends for sharing this journey together and making these years feel funnier and more exciting.