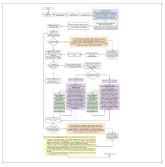

Version 3

May 03, 2021

Investigating Invalid DOIs in COCI - Protocol V.3

Sara Coppini[1], Nooshin Shahidzadeh[1], Alessia Cioffi[1], Arianna Moretti[1]

[1]University of Bologna

Sara Coppini: Bachelor's degree in Philosophy, currently studying Digital Humanities and Digital Knowledge (Master Degree) at the University of Bologna.

Nooshin Shahidzadeh: Bachelor's degree in Software Engeneering, currently studying Digital Humanities and Digital Knowledge (Master Degree) at the University of Bologna.

Alessia Cioffi: Bachelor's degree in Classical Literature, currently studying Digital Humanities and Digital Knowledge (Master Degree) at the University of Bologna.

Arianna Moretti: Bachelor's degree in Anthropology, Religions, Oriental Civilizations, currently studying Digital Humanities and Digital Knowledge (Master Degree) at the University of Bologna.

1 *Works for me* dx.doi.org/10.17504/protocols.io.buqhnvt6

Open Science 2020/2021

Sara Coppini

Citation: Sara Coppini, Nooshin Shahidzadeh, Alessia Cioffi, Arianna Moretti (05/03/2021). Investigating Invalid DOIs in COCI - Protocol. https://dx.doi.org/10.17504/protocols.io.buqhnvt6

ABSTRACT

A preliminary note

This protocol illustrates the workflow adopted within a scholarly research that operates within the OpenCitations environment, which is an independent infrastructure organization for open scholarship dedicated to the publication of open bibliographic and citation data by the use of Semantic Web (Linked Data) technologies. COCI is the OpenCitations Index of Crossref open DOI-to-DOI citations.

Purpose

The primary purpose of this research is to find the publishers responsible for the missing citations in COCI (the OpenCitations Index of Crossref open DOI-to-DOI citations) by sending incorrect metadata to Crossref, the publishers to whom such invalid citations point to, and the number of previously invalid citations which are currently valid. Further, the additional aim of the present research is to provide material for future and deeper research on the same subject matter. In particular, we focus on keeping track of the main trends of evolution on the validity of citational data and on providing data to facilitate publishers' identification.

Study design/methodology

In order to study the invalid citations, we use an already generated CSV file which is available online, containing - for each citation - the valid citing DOI and the invalid cited DOI. First of all, through a DOI API request we check whether the validity state of the cited DOI has changed, and then we use DOI's prefix for a CROSSREF API request, in order to identify the responsible and referenced publishers.

Findings

For each individual publisher, we retrieve the number of incorrect given citations metadata sent, and the number of invalid citations received, to which we decided to add the information about the number of addressed and received citations validated with the software we developed, and its list of prefixes. We also extract the total number of invalid citations that have since been corrected. Any further material provided as result of this research, such as the lists of validated and still invalid citations, is meant to incentivize future improvements of the studies on this field.

Originality/value

The results of this research may point us to publishers who generally send out incorrect citation metadata and, inversely, those who generally receive invalid citations. These findings can first of all raise awareness of the accuracy of certain publishing houses in managing their metadata (or lack thereof). Moreover, finding these trends and showcasing the labor of the corrections may lead to increasingly valid citations if the proper measures are taken.

Research limitations/implications

Based on the available data for COCI, there may be a slight bias in our sample, causing some publishers to be incorrectly represented.

Minimal bibliography on related projects

- Heibi, I., Peroni, S. & Shotton, D. Software review: COCI, the OpenCitations Index of Crossref open DOI-to-DOI citations.Scientometrics 121, 1213–1228 (2019). https://doi.org/10.1007/s11192-019-03217-6.
- Silvio Peroni, David Shotton (2020). OpenCitations, an infrastructure organization for open scholarship. Quantitative Science Studies, 1(1): 428-444. https://doi.org/10.1162/qss_a_00023.
- Peroni, S. (2021). Citations to invalid DOI-identified entities obtained from processing DOI-to-DOI citations to add in COCI (1.0). Zenodo. https://doi.org/10.5281/ZENODO.4625300.

Related Diagrams

In order to improve readability and reusability of our protocol, we provide here both the reduced and the extended version of the flow diagram representing the steps of the procedure.

InvestigatingMissingCitationsInCociReduced.pdf

InvestigatingMissingCitationsInCoci.pdf

Citation: Sara Coppini, Nooshin Shahidzadeh, Alessia Cioffi, Arianna Moretti (05/03/2021). Investigating Invalid DOIs in COCI - Protocol. https://dx.doi.org/10.17504/protocols.io.buqhnvt6

DOI

dx.doi.org/10.17504/protocols.io.buqhnvt6

PROTOCOL CITATION

Sara Coppini, Nooshin Shahidzadeh, Alessia Cioffi, Arianna Moretti 2021. Investigating Invalid DOIs in COCI - Protocol. **protocols.io**
https://dx.doi.org/10.17504/protocols.io.buqhnvt6
Version created by Nooshin Shahidzadeh

KEYWORDS

data science, COCI, citations, Crossref, Open Science, OpenCitations, publishers, DOI, Digital Object Identifier, Python, publisher

CREATED

May 03, 2021

LAST MODIFIED

May 03, 2021

OWNERSHIP HISTORY

May 03, 2021 Sara Coppini

PROTOCOL INTEGER ID

49641

GUIDELINES

In order to properly follow the steps of the present protocol, you will need to install Python 3.6 - or any other subversion of Python 3. Needed packages: requests ~=2.25.
The graphic visualizations have been made with the javascript libraries:
chart.js version 2.5.0 https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.5.0/Chart.min.js;
d3.js version 4, https://d3js.org/d3.v4.min.js.

BEFORE STARTING

Before starting, make sure you are going to work with the required Python version, or a compatible one, and that you have requests ~=2.25 installed.

Presenting the Input Material

1 Given the academic background of the project, we got our input material from Peroni, S. (2021). Citations to invalid DOI-identified entities obtained from processing DOI-to-DOI citations to add in COCI (1.0). Zenodo. https://doi.org/10.5281/ZENODO.4625300. As explained in the description of the resource, this dataset contains a two-column CSV file, where the first column ("Valid_citing_DOI") contains the DOI of a citing entity retrieved in Crossref, while the second column ("Invalid_cited_DOI") contains the invalid DOI of a cited entity identified by looking at the field "reference" in the JSON document returned by querying the Crossref API with the citing DOI.
These citations to invalid DOIs have been retrieved while processing Crossref data for adding open citations in COCI, but they have not been added in COCI since they point to a non-resolvable cited document.

Reading the CSV Data

2 First we read the CSV of the invalid DOIs, containing all the invalid cited DOIs in one column and their valid citing counterparts in another, using xsv_reader.

Citations to invalid DOI-identified entities

2.1 Then, we create:
1. an empty dictionary (i.e.: publisher_data), which is going to be filled with dictionaries representing each publisher encountered,
2. two empty lists (i.e.: correct_dois_data and incorrect_dois_data), which are going to store citational data in form of lists.

2.2 After that, in order to retrieve data gathered in the previous runs of the code, we run the function extract_row_number(publisher_data), which reads the cache files we create in the following steps and fills the aforementioned publisher_data dictionary and also returns:
1. start_index, an integer which shows from which row in the CSV we should restart the process,
2. prefix_to_name_dict, a dictionary which keeps known prefix to publisher name matchings, in order to avoid making extraneous API calls when encountering an already seen prefix.

Processing Each Line in the CSV File and Extracting the Needed Information

3 In this step, we check if we have processed 100 rows since the last write to cache. If we have, we write to the cache files once again.

3.1 We initialize a counter's value to 0 (i.e.: i = 0), so to keep the count of the already processed rows.

3.2 If the counter is lesser than 100, we skip to the API request step. In the opposite case, when the counter reaches 100 (a sample value that can be changed if wanted), we call the write_to_csv(publisher_data, prefix_to_name_dict, correct_dois_data, incorrect_dois_data) function, whose aim is that of writing everything we have processed so far to cache files in order to avoid loss of data in case the running of the code is for some reason interrupted. The cache files are:
1. correct_dois.csv, which includes every citation row with a cited DOI we found to have been validated since the creation of our source, the invalid_dois.csv file,

Valid_citing_doi	Valid_cited_doi
10.1007/s11771-020-4410-2	10.13745/j.esf.2016.02.011
10.1007/s40617-018-00299-1	10.1901/jaba.2012.45-657
10.1145/3190617	10.1145/2838344.2856460

1. incorrect dois.csv, which includes every citation row with a cited DOI we found to still be invalid,

Valid_citing_doi	Invalid_cited_doi
10.14778/1920841.1920954	10.5555/646836.708343
10.5406/ethnomusicology.59.2.0202	10.2307/20184517
10.1161/01.cir.63.6.1391	10.1161/01.cir.63.6.1391

- publiser_data.csv, which includes a row for each publisher, containing information on:

1. its name,
2. the number of still invalid citations it is responsible for,
3. the number of now valid citations it is responsible for,
4. the number of still invalid citations its publications have received,

Citation: Sara Coppini, Nooshin Shahidzadeh, Alessia Cioffi, Arianna Moretti (05/03/2021). Investigating Invalid DOIs in COCI - Protocol. https://dx.doi.org/10.17504/protocols.io.buqhnvt6

5. the number of now valid citations its publications have received.

name	responsible_for_v	responsible_for_i	receiving_v	receiving_i
VLDB Endowment	15	377	120	0
Test accounts	0	0	0	17267
University of Illinois Press	0	187	0	6

- prefix_name.json, which includes the aforementioned prefix to publisher name matchings.

```
{
    "10.14778": "VLDB Endowment",
    "10.5555": "Test accounts",
    "10.5406": "University of Illinois Press",
    ...
}
```

3.3 We empty both correct_dois_data and incorrect_dois_data by assigning to them two empty lists. We then increment the value of the variable start_index by i (i.e.: the value of the counter), and we re-initialize the value of the counter i to 0.

4 In order to verify whether the validity state of the previously invalid cited DOI of a citation has changed over the time, we make an API request to the doi.org service.
First of all, we assign to a URL variable a value composed of a base URL for the API request (i.e.: "https://doi.org/api/handles/") and the Digital Object Identifier of the cited element of the citational data, retrieved by considering the second element of each row of the input file.
We use the composed url to try to make the effective API request considering the obtained response status code.
In the case the request fails, a connection error is raised and the running of the application stops.

Step 4 includes a Step case.
Now Valid
Still Invalid

step case

Now Valid

In this case we extract the publisher data and add the row to the valid citations cache.

5 In the case the response is positive (i.e.: we get a status code 200), we append the list representing the citational data to correct_dois_data list, and we call the function extract_publisher_valid(row, publisher_data, prefix_to_name_dict) in order to manage the identification of both the publishers of the citing and the cited DOIs. With this function, we first check whether the prefix of the cited DOI exists in our prefix to publisher name matching dictionary. If it does not, we call the extract_publishers(prefix, prefix_to_name_dict) function to find the name of the publisher by sending an API request to the Crossref REST API for prefixes, https://api.crossref.org/prefixes/, (putting the publisher's name as "unidentified" in case Crossref does not recognize the prefix) and then, if the publisher name is not already a key in the publisher_data dictionary, we create a new item in the publisher_data dictionary with key set to the name we found and the respective field based on the publisher (i.e. "responsible_for_v" or "receiving_v") set to 1. otherwise , based on the function, we add 1 to the respective field of the publisher item with that name as key in the publisher_data dictionary (i.e. "responsible_for_v" or "receiving_v").

Creating the Output JSON File

6 In this step, we read all the cache files and create a final output file, deleting the caches in the process.

Citation: Sara Coppini, Nooshin Shahidzadeh, Alessia Cioffi, Arianna Moretti (05/03/2021). Investigating Invalid DOIs in COCI - Protocol. https://dx.doi.org/10.17504/protocols.io.buqhnvt6

6.1 In order to store the data retrieved after the last cache files update (i.e.: the data related to the csv lines processed after the last re-initialization of the i counter to 0), the write_to_csv(publisher_data, prefix_to_name_dict, correct_dois_data, incorrect_dois_data) function is called once again, and the two lists correct_dois_data and incorrect_dois_data are newly emptied.
At this point, the retrieved data has to be used to compile the output JSON file.

6.2 We read all cache files and put them together in the following way:

```
{
    "citations": {
        "valid": [
            {
                "Valid_citing_doi": "10.1007/s11771-020-4410-2",
                "Valid_cited_doi": "10.13745/j.esf.2016.02.011"
            },
            ...
         ]
        "invalid": [
            {
                "Valid_citing_doi": "10.14778/1920841.1920954",
                "Invalid_cited_doi": "10.5555/646836.708343"
            },
            ...
        ]
    },
    "publishers": [
        {
            "name": "VLDB Endowment",
            "responsible_for_v": 15,
            "responsible_for_i": 377,
            "receiving_v": 120,
            "receiving_i": 0,
            "prefix_list": [
                "10.14778"
            ]
        },
        ...
    ],
    "total_num_of_valid_citations": 12219
}
```

The output file is a json file which includes the following fields:

- citations, a dictionary with the two fields "valid" and "invalid", each of which includes a list of dictionaries for all the citation data in correct_dois.csv and incorrect_dois.csv respectively,
- publishers, a list of dictionaries, each of which contains all information gathered about a publisher extracted from publisher_data.csv and prefix_name.json,
- total_num_of_valid_citations, an integer showing the number of processed citation data that we found to have been validated after the initial source file creation.

Creating Visualizations

7 As an extra step, from the collected data we have extrapolated and restructured the relevant information for each research question. We have chosen to represent the data concerning the first two questions with stacked bar charts and tables, and those concerning research question number three with a donut chart. We used Javascript, JQuery,

Citation: Sara Coppini, Nooshin Shahidzadeh, Alessia Cioffi, Arianna Moretti (05/03/2021). Investigating Invalid DOIs in COCI - Protocol. https://dx.doi.org/10.17504/protocols.io.buqhnvt6

d3.js and ChartJS.

Citation: Sara Coppini, Nooshin Shahidzadeh, Alessia Cioffi, Arianna Moretti (05/03/2021). Investigating Invalid DOIs in COCI - Protocol. https://dx.doi.org/10.17504/protocols.io.buqhnvt6