Open Access Evidence in Unpaywall

Open Access Evidence in Unpaywall

Unpaywall has become a primary source to find open access full-texts. We investigated more than 42 million scholarly works contained in Unpaywall that were published between 2008 - 2018 with Google BigQuery and R. Our data analysis revealed various open access location types and large overlaps between them, raising important questions about how to reponsibly re-use Unpaywall data in bibliometric research and open access monitoring.

Najko Jahn https://twitter.com/najkoja (State and University Library Göttingen)https://www.sub.uni-goettingen.de/ , Anne Hobert (State and University Library Göttingen)https://www.sub.uni-goettingen.de/
2019-04-17

Unpaywall, developed and maintained by the team of Impactstory, is a non-profit service that finds open access copies of scholarly literature. Providing DOIs, Unpaywall's REST API not only returns open access full-text links, but also helpful metadata about the open access status of a publication indexed in Crossref. While the API is useful for a limited amount of scholarly works, Unpaywall also provides database snapshots for large-scale analysis, which many bibliometric databases and open access monitoring activities re-use.

In this blog post, we show how we made use of the Unpaywall data dump together Google BigQuery, a cloud-based service that allows fast analysis of large datasets, and how we interfaced BigQuery for our analysis with R. We wanted to know the extent of open access status information in Unpaywall, particularly, how this information can be utilized for bibliometric research. In our case, we intend to match evidence from Unpaywall with the Web of Science in-house database from the German Competence Center for Bibliometrics to determine factors influencing open access publication activities among German Universities as part of our BMBF-funded research project OAUNI.

Setting up Google Big Query

Unpaywall Overview

To investigate the overall number and proportion of journal articles provided using Unpaywall, we firstly connect to Google BigQuery with using the DBI interface and bigrquery.

```
#' connect to google bg where we imported the jsonl Unpaywall dump
library(DBI)
library(bigrquery)
con <- dbConnect(
  bigrquery::bigquery(),
  project = "api-project-764811344545",
  dataset = "oadoi_full"
)
```

Our BigQuery project has two table, one containing all records between 2008 - 2012, and another with publications since 2019. When connecting, Google will ask you to login via your web browser or to supply an private access token.

```
upw_08_12 <- tbl(con, "feb_19_mongo_export_2008_2012_full_all_genres")
upw_13_19 <- tbl(con, "feb_19_mongo_export_2013_Feb2019_full_all_genres")
```

The bigrquery package allows querying BigQuery tables using SQL or the dplyr syntax. The latter is very convienent when you just started to learn SQL, but feel more experienced in the tidyverse. Here's an example where we obtain the first ten records from 2018, restricting our search to journal articles, the most common publication genre in Unpaywall.

```
upw_13_19 %>%
  filter(year == 2018, genre == "journal-article") %>%
  head()
#> # Source:   lazy query [?? x 13]
#> # Database: BigQueryConnection
#>    year genre updated             published_date journal_is_in_d...
#>   <int> <chr> <dttm>              <date>         <lgl>
#> 1  2018 jour... 2019-01-20 22:58:42 2018-02-01     TRUE
#> 2  2018 jour... 2018-06-16 16:34:02 2018-05-01     FALSE
#> 3  2018 jour... 2019-02-11 20:30:47 2018-03-06     FALSE
#> 4  2018 jour... 2018-12-17 21:46:48 2018-04-28     FALSE
#> 5  2018 jour... 2018-09-15 08:27:25 2018-04-01     TRUE
#> 6  2018 jour... 2018-12-04 21:06:18 2018-12-03     TRUE
#> # ... with 8 more variables: journal_is_oa <lgl>, journal_issns <chr>,
#> #   oa_locations <list>, doi <chr>, is_oa <lgl>, publisher <chr>,
#> #   journal_name <chr>, data_standard <int>
```

Notice that our schema follow the Unpaywall data format. The column

`oa_locations` is a list-column that contain comprehensive metadata about the OA sources found by Unpaywall.

To start with, we want to retrieve the number and proportion of journal articles with open access fulltext between 2008 and 2018 using the most basic Unpaywall variable `is_oa`, a logical value, which is `TRUE` when at leats one open access fulltext was found. `collect()` loads the data into a local tibble.

```
library(tidyverse)
oa_08_12 <- upw_08_12 %>%
  # query and aggregate with dpylr
  filter(genre == "journal-article") %>%
  group_by(year, is_oa) %>%
  summarise(n = n()) %>%
  # load the data into a local tibble
  collect()
oa_13_18 <- upw_13_19 %>%
  # query and aggregate with dpylr
  filter(genre == "journal-article", year < 2019) %>%
  group_by(year, is_oa) %>%
  summarise(n = n()) %>%
  # load the data into a local tibble
  collect()
my_df <- bind_rows(oa_08_12, oa_13_18) %>%
  # calculate proportion per year
  ungroup() %>%
  mutate(year = as.Date(as.character(year), format = "%Y")) %>%
  group_by(year, is_oa) %>%
  summarise(n = sum(n)) %>%
  mutate(prop = n / sum(n))
my_df
#> # A tibble: 22 x 4
#> # Groups:   year [11]
#>    year       is_oa       n  prop
#>    <date>     <lgl>   <int> <dbl>
#>  1 2008-04-17 FALSE 1454745 0.711
#>  2 2008-04-17 TRUE   590250 0.289
#>  3 2009-04-17 FALSE 1562765 0.701
#>  4 2009-04-17 TRUE   665951 0.299
#>  5 2010-04-17 FALSE 1724760 0.697
#>  6 2010-04-17 TRUE   749139 0.303
#>  7 2011-04-17 FALSE 1585092 0.654
#>  8 2011-04-17 TRUE   838014 0.346
#>  9 2012-04-17 FALSE 1636130 0.630
#> 10 2012-04-17 TRUE   962036 0.370
#> # ... with 12 more rows
```

In total, 31,159,960 journal articles published between 2008 - 2018 were included in the Unpaywall. For 11,633,886 articles, Unpwaywall was able to link a DOI to at least one freely available full-text (37 %).

Next, let's create a graph that presents the prevalence of open access to journal articles over year. We turn our ggplot object into an interactive plotly chart, a javascript library, using `ggplotly()`.

```
library(scales)
plot_a <- my_df %>%
  # prepare label that we want to present as tooltip
  mutate(`Proportion in %` = round(prop * 100, 2)) %>%
  ggplot(aes(year, n, label = `Proportion in %`)) +
  geom_area(aes(fill = is_oa, group = is_oa),  alpha = 0.8) +
  labs(x = "Year published", y = "Journal Articles",
       title = "Open Access to Journal Articles") +
  scale_fill_manual("Is OA?",
                    values = c("#b3b3b3a0", "#56B4E9")) +
  scale_x_date(date_labels = "%y") +
  scale_y_continuous(labels = scales::number_format(big.mark = " ")) +
  theme_minimal(base_family = "Roboto") +
  theme(plot.margin = margin(30, 30, 30, 30)) +
  theme(panel.grid.minor = element_blank()) +
  theme(axis.ticks = element_blank()) +
  theme(panel.grid.major.x = element_blank()) +
  theme(panel.border = element_blank())
# turn ggplot object into interactive plotly chart
plotly::ggplotly(plot_a, tooltip = c("label", "y"))
```

Figure 1: Open Access to Journal Articles according to Unpaywall data. Blue areas represent journal articles with at least one freely available full-text, the gray represent toll-access articles.

While a general growth of journal articles as well open access provision can be observed, there is a considerable decline in the number of journal articles published in 2018, presumbably because of an indexing lag between Crossref and Unpaywall. The decline in open access full-text availability was even clearer in comparison between 2017 and 2018, suggesting that some open access content is provided after an embargo period.

Unpaywall OA location types

Using Unpaywall's OA location types allows a more detailled analysis of open access provision. In the following, we explore the variable `host_type`, showing whether Unpaywall found the open access full-text on a publisher website or in a repository. We furthermore want to include articles from fully open access journals that are indexed in Directory of Open Access Journals (DOAJ) as

indicated by the `journal_is_in_doaj` variable. A a start, we only examine the best open access location per DOI, `is_best`, defined by Unpaywall's algorithm that prioritizes publisher-hosted content.

Instead of dplyr, we are now querying BigQuery with SQL. We built and tested the SQL query in the BigQuery web UI. Here are our queries:

```
host_type_08_12_query <- "SELECT year, host_type, journal_is_in_doaj, COUNT(DISTINCT(doi)) A
host_type_13_18_query <- "SELECT year, host_type, journal_is_in_doaj, COUNT(DISTINCT(doi)) A
```

Let's call BigQuery:

```
host_type_08_12_query_df <- dbGetQuery(con,host_type_08_12_query)
host_type_13_18_query_df <- dbGetQuery(con,host_type_13_18_query)
host_type_df <- bind_rows(host_type_08_12_query_df,host_type_13_18_query_df) %>%
  mutate(host = case_when(journal_is_in_doaj == TRUE ~ "DOAJ-listed Journal",
                          host_type == "publisher" ~ "Other Journals",
                          host_type == "repository" ~ "Repositories only")) %>%
  mutate(year = as.Date(as.character(year), format = "%Y"))
host_type_df
#> # A tibble: 34 x 5
#>    year       host_type  journal_is_in_do... number_of_artic... host
#>    <date>     <chr>      <lgl>                          <int> <chr>
#>  1 2012-04-17 publisher  FALSE                         554081 Other Jou...
#>  2 2012-04-17 publisher  TRUE                          219293 DOAJ-list...
#>  3 2008-04-17 repository FALSE                         149967 Repositor...
#>  4 2008-04-17 publisher  TRUE                           81731 DOAJ-list...
#>  5 2010-04-17 publisher  FALSE                         436375 Other Jou...
#>  6 2011-04-17 repository FALSE                         178323 Repositor...
#>  7 2011-04-17 publisher  TRUE                          170917 DOAJ-list...
#>  8 2009-04-17 repository FALSE                         165372 Repositor...
#>  9 2012-04-17 repository FALSE                         188662 Repositor...
#> 10 2010-04-17 publisher  TRUE                          139345 DOAJ-list...
#> # ... with 24 more rows
```

As to explore our data, we follow Claus Wilke excellent book "Fundamentals of Data Visualization" and visualise our proportions separately as parts of the total. Again, our final ggplot object will be transformed to an interactive plotly chart.

```
# calculate all oa articles per year
all_articles <- host_type_df %>%
  ungroup() %>%
  group_by(year) %>%
  summarise(number_of_articles = sum(number_of_articles))

plot_b <-
  ggplot(host_type_df, aes(x = year, y = number_of_articles)) +
  geom_bar(
```

```
    data = all_articles,
    aes(fill = "All OA Articles"),
    color = "transparent",
    stat = "identity"
  ) +
  geom_bar(aes(fill = "by Host"), color = "transparent", stat = "identity") +
  facet_wrap( ~ host, nrow = 1) +
  scale_fill_manual(values = c("#b3b3b3a0", "#56B4E9"), name = "") +
  labs(x = "Year", y = "OA Articles (Total)") +
  theme(legend.position = "top",
        legend.justification = "right") +
  scale_x_date(date_labels = "%y") +
  scale_y_continuous(labels = scales::number_format(big.mark = " ")) +
  theme_minimal(base_family = "Roboto") +
  theme(plot.margin = margin(30, 30, 30, 30)) +
  theme(panel.grid.minor = element_blank()) +
  theme(axis.ticks = element_blank()) +
  theme(panel.grid.major.x = element_blank()) +
  theme(panel.border = element_blank())
# turn ggplot object into interactive plotly chart
plotly::ggplotly(plot_b, tooltip = c("y"))
```

Figure 2: Open Access to journal articles by open access hosting location. The colored bars represent the number of open access articles per host ("DOAJ-listed Journal", "Other Journals", "Repositories only"), the gray bar the total number of journal articles indexed in Crossref where Unpaywall was able to identify at least one open access full-text.

The figure shows that most publisher-provided open access links were obtained from journals that were not indexed in the DOAJ (6,531,822 articles, representing 56 % of all journal articles with openly available full-text identified by Unpaywall), raising important questions about the other ways publisher made articles openly available.

Currently, we are discussing the following explanations:

1. a journal did not meet DOAJ indexing criteria, but is indexed by Crossref. Examples include articles from so-called "predatory publishers" like OMICS or collections of procedia published as journals(e.g. Elsevier's Procedia journals.
2. Toll-access journals made specific articles openly available immediately upon publication ("hybrid open access").
3. The category "Other Journals" accounts for delayed open access journals where full issues were made openly available after an embargo period. Example includes the publisher Elsevier, which lists 118 journals that provide free access after a certain period of time, ranging between six and

48 months. The sharp decline in 2018 suggests that the proportion of delayed open access provided by publishers should not be underestimated.

Unpaywall OA evidence types

Discussion and Conclusion

Reuse