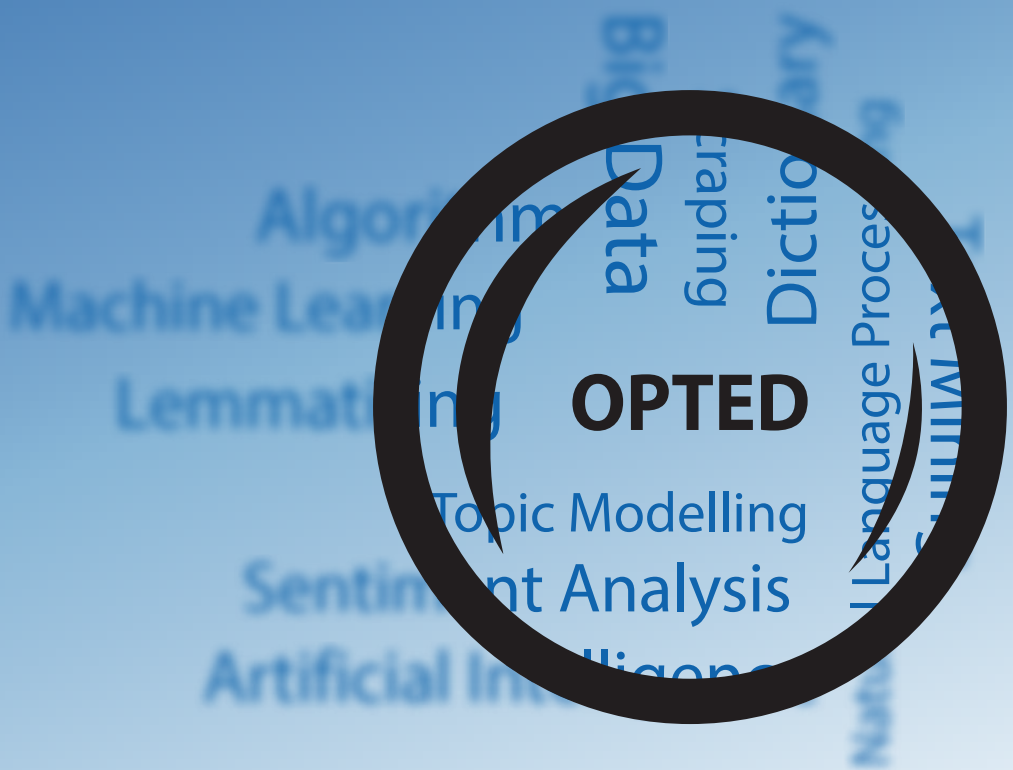
OPTED

AmCAT 4: a toolkit for corpora storage, sharing and pre-processing

Deliverable 7.4

Wouter van Atteveldt, Johannes B. Gruber, and Kasper Welbers

Vrije Universiteit Amsterdam



D7.4: AmCAT 4: a toolkit for corpora storage, sharing and pre-processing

Disclaimer

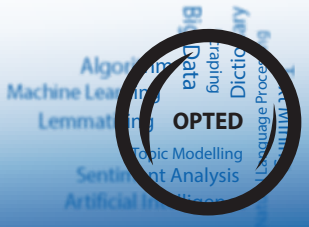
This project has received funding from the European Union's Horizon 2020 research & innovation programme under grant agreement No 951832. The document reflects only the authors' views. The European Union is not liable for any use that may be made of the information contained herein.

Dissemination level

Public

Type

Report



OPTED
Observatory for Political Texts in European Democracies:
A European research infrastructure

AmCAT 4: a toolkit for corpora storage, sharing and pre-processing

Deliverable 7.4

Authors: Wouter van Atteveldt, Johannes B. Gruber, and Kasper Welbers

Vrije Universiteit Amsterdam

Due date: 30. September 2023

Contents

1 Statement of need

Sharing a collection of text data with other researchers comes with different needs than sharing, for example, a survey data set. Researchers who want to use a corpus for secondary analysis might want to explore it first to see if it contains relevant data; they might then want to filter said data set using metadata filters, keywords or search queries; text data nowadays often come with limitations for re-distribution connected to copyright or privacy concerns, making alternative distribution avenues (such as only sharing metadata or pre-processed versions of text) necessary. Current infrastructure for sharing data, such as the Dataverse, were not built with these needs in mind, as they generally do not apply to the much smaller data sets that, for example, survey researchers use. In the current iteration of the *Amsterdam Content Analysis Toolkit* (AmCAT), we focused on addressing these needs of the text-as-data community. In a time when access to data becomes more difficult due to APIs of social media platforms being shut down and media outlets starting to defend themselves more rigorously against scraping, the importance of sharing corpora for secondary analysis has suddenly and drastically surged. With the AmCAT infrastructure software we developed as part of the EU-funded H2020 project OPTED (**O**bservatory for **P**olitical **T**exts in **E**uropean **D**emocracies), we hope to encourage the research community to embrace more widespread sharing and secondary analysis of text data.

2 Summary

The Amsterdam Content Analysis Toolkit (AmCAT) has been in development in various guises since about 2001 as a text research / content analysis platform. The core of the project has always been a database of documents, combined with a graphical user interface and an API to facilitate targeted text data extraction for casual and power users alike. In the current fourth iteration of the toolkit, a particular focus was put on sharing data online, while access is being managed with a flexible set of user roles. Within the OPTED project, we identified the needs for ourselves and the text-as-data community, developed AmCAT to address these needs and based the framework on a modular design and open source tools. Figure 1 shows the parts of the toolkit, which consists of a data layer that contains the storage, optional back-end applications that can be used to annotate or pre-process data, and the user facing frontend, which allows access via a graphical user interface or API clients for `R` and `Python`. The modular design means that parts of the toolkit may be included in existing or custom infrastructure.[1]

2.1 Web Interface and API

Instead of just showing how many documents are contained in a corpus, AmCAT makes it possible to easily explore data before committing to it. Figure 2 shows the ParlEE data set (Sylvester, Greene, and Ebing 2022), which consists of several gigabytes of csv files and can be downloaded from the Harvard Dataverse. In AmCAT, it can be explored by, for example, filtering for speeches in the Austrian parliament that feature a term connected to

[1] For example, the dashboard at https://parliaments.opted.eu/dashboard was developed on top of the AmCAT storage and query modules, but with an entirely differenet custom interface.

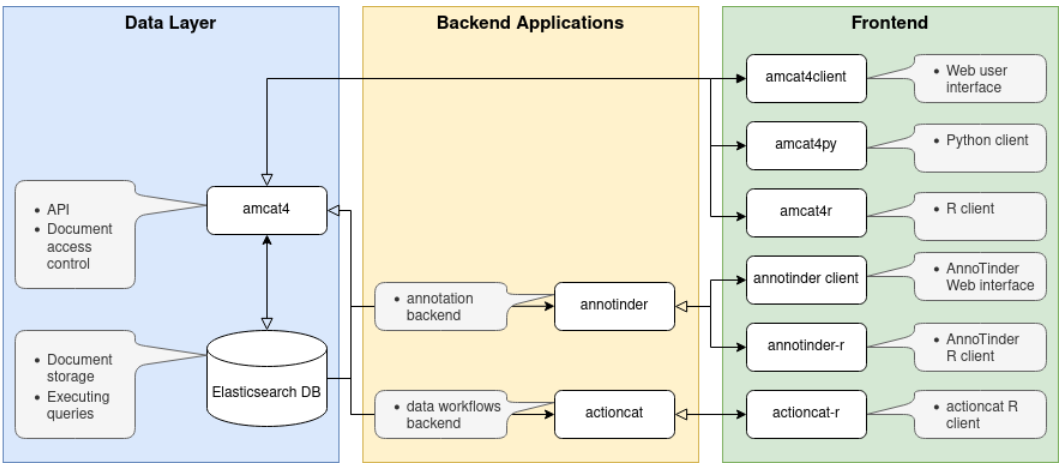


Figure 1: The AmCAT4 design overview.

migration. The example shows that there is a stark increase in mentions of the terms in 2015, which is probably interesting researchers focused on the migration debate. Queries are made in *Elasticsearch*'s "mini-language" for query strings, which is widely used and well documented.[2]

Besides the web interface, we offer a fully featured REST API and API wrapper packages for `R` and `Python`. We also include OpenAPI specifications in the default installation, which makes it easy to develop other wrappers. The API has the same search, upload, download, and data modification capabilities as the web interface. Below, we reproduce the search from Figure 2 inside `R`:

[2]https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-query-string-query.html#query-string-query-notes

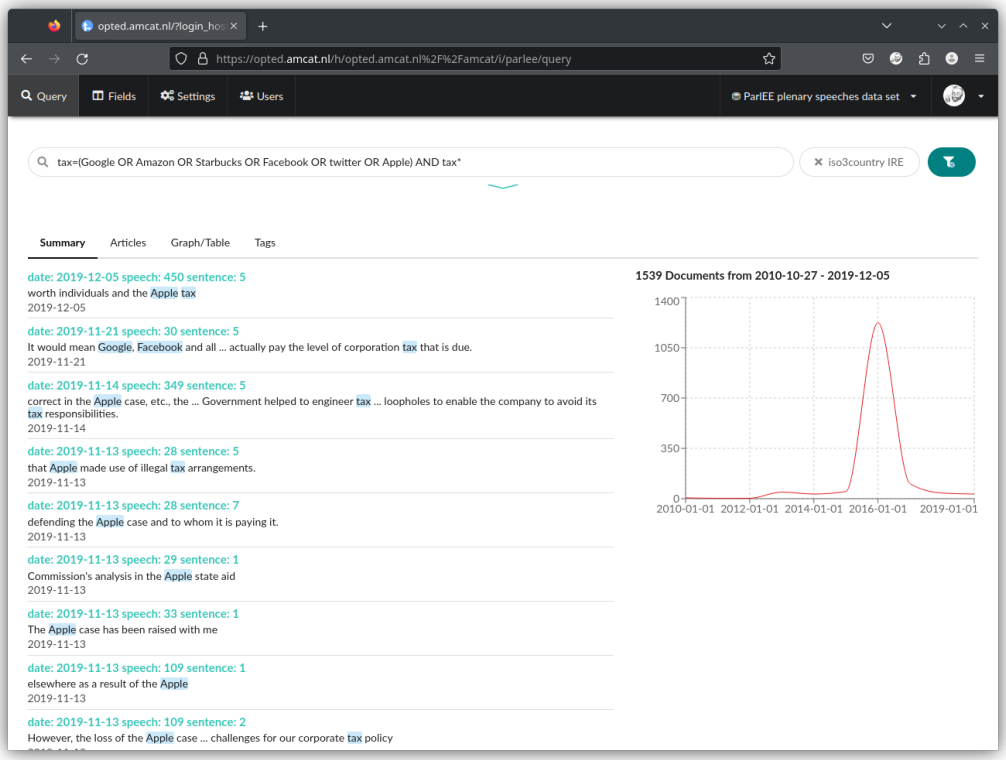
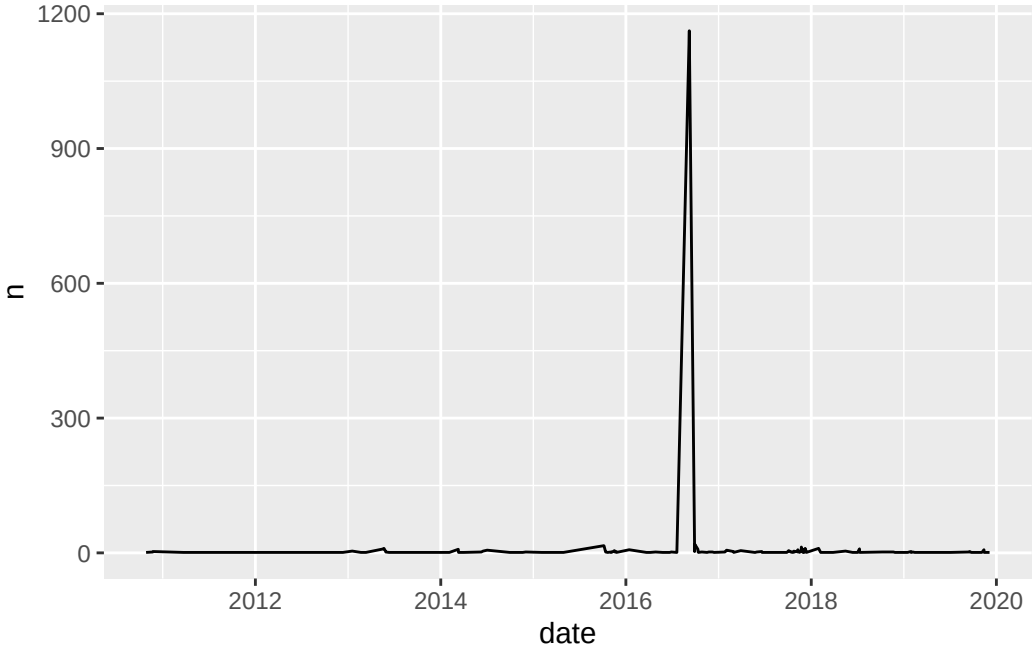


Figure 2: ParlEE data set on AmCAT web interface.

```r
library(tidyverse)
library(amcat4r)
# publicly available server
amcat_login("https://opted.amcat.nl/amcat")
query_documents(
  "parlee", # name of the index
  queries = "(Google OR Amazon OR Starbucks OR Facebook OR twitter
   OR Apple) AND tax*",
  # include all fields
  fields = NULL,
  # only search in Austrian corpus
  filters   = list(iso3country = "IRE"),
  per_page = 10000,
  page = NULL, # retrieve all pages
  max_pages = Inf
) |>
  # visualisation using ggplot2
  count(date) |>
  ggplot(aes(x = date, y = n)) +
  geom_line()
```



The API makes AmCAT a valuable asset for power users and enables them to include data in their reproduction materials. API commands can also be used to in a script for larger-than-memory data, as chunks of the data can be processed one by one instead of loading the whole data into memory first.

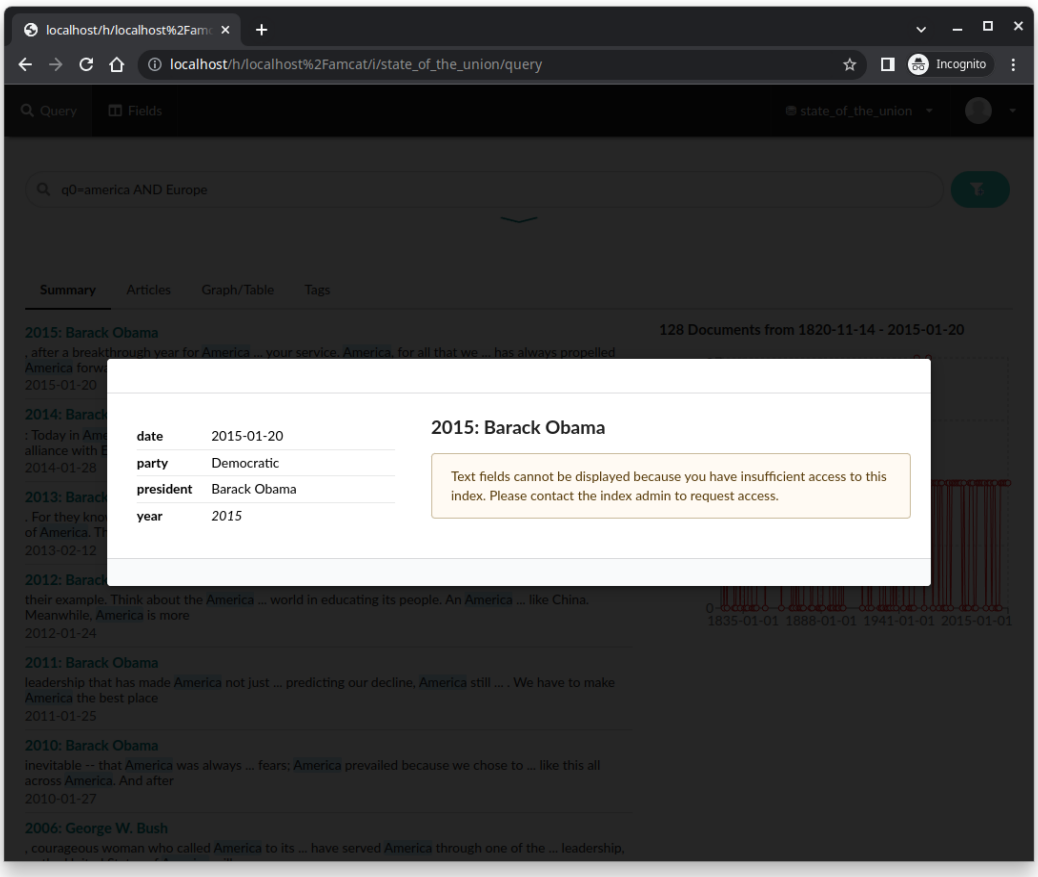


Figure 3: 'Metareader' users can query, but not read text.

2.2 Flexible Access control

An important feature of AmCAT is that it allows for fine-grained access control and *non-consumptive* research – analyses that use computational methods, while the researcher can not consume (i.e., read) the text. This makes AmCAT the only easily available open source tool (that we are aware of), which can enable researchers to share a corpus in a way that users can query and visualize the data without being granted access to the underlying text – which circumvents most copyright and ethical restrictions. We accomplish this with a set of instance and index roles that allow the admin users to grant or revoke access of indices (i.e., text collections), fields (columns) and the AmCAT instance. What makes this special is that users can be granted access to only to metadata of a text data set. For the queries an searches a user can perform, this makes little difference. Only when they try to read a text, they are presented with the message in Figure 3.

This setup allows for *non-consumptive* research similar to the functions of *Google Books*, where researchers can check if a corpus contains relevant data before working towards getting access. We extend these capabilities by providing a framework to package arbitrary workflows that can then be run on an AmCAT instance.

Using `actioncat`, users can write a script in `R`, `Python` or another language, package it into a Docker container and send a Docker Compose file to the administrator of a server to check if it conforms with theor policies on data access. Doing so enables researchers to perform, for example, destructive pre-processing of text (i.e., where the original text can not be restored). In this way, a corpus can still be used for essentially every text-as-data analysis pipeline imaginable, without running risk of violating copyrights or the privacy of data owners. We provide two example workflows, one that uses `R` to turn text into a document-feature-matrix, one that uses `Python` to turn the text into document embeddings.

To authenticate users, we wrote our own authentication provider called `middlecat`. Unlike in previous iterations, this completely omits passwords in favor of connecting to identity providers like Github, ORCID, or Google or one-time login links sent via e-mail. The advantage is that we can follow a parsimonious approach to user privacy (besides email addresses, we keep no user data) while still making sure that the user is actually who they claim.

2.3 Installation and Documentation

All different parts of AmCAT are publicly available on GitHub:

- `amcat4`: https://github.com/ccs-amsterdam/amcat4
- `amcat4actioncat`: https://github.com/ccs-amsterdam/amcat4actioncat
- `amcat4client`: https://github.com/ccs-amsterdam/amcat4client
- `amcat4py`: https://github.com/ccs-amsterdam/amcat4py
- `amcat4r`: https://github.com/ccs-amsterdam/amcat4r
- `middlecat`: https://github.com/ccs-amsterdam/middlecat

The individual repositories contain instructions to install the packages. We also provide a more comprehensive manual at https://amcat.nl/book/, which covers installation, workflows, user management and so on.

We recommend to install AmCAT through the open-source application Docker. In our `amcat4docker` repository, we provide several Docker Compose files that make it possible to get a full AmCAT instance running in minutes with no other dependecies than Docker itself.

3 References

Sylvester, Christine, Zachary Greene, and Benedikt Ebing. 2022. “ParlEE plenary speeches data set: Annotated full-text of 21.6 million sentence-level plenary speeches of eight EU states.” Harvard Dataverse. https://doi.org/10.7910/DVN/ZY3RV7.