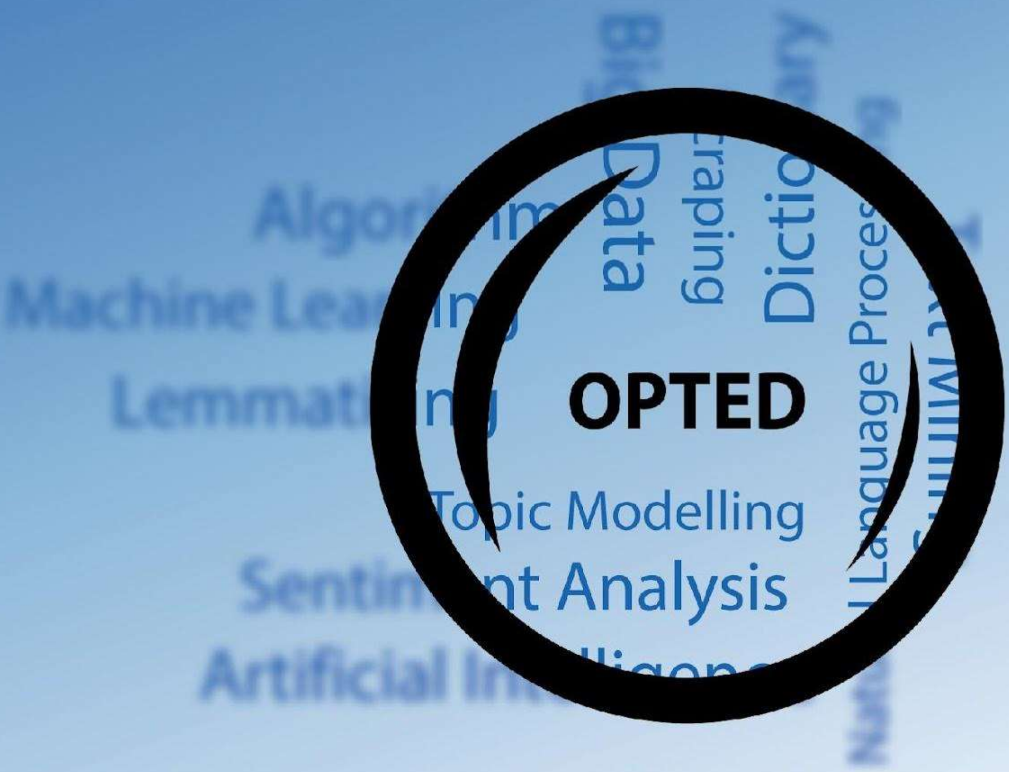
# OPTED

## Tool Collection

Paul Balluff, Marvin Stecker, Fabienne Lind,
Hajo G. Boomgaarden & Annie Waldherr



**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

# Tool collection

**Deliverable 3.3**

**Authors: Paul Balluff, Marvin Stecker, Fabienne Lind, Hajo G. Boomgaarden & Annie Waldherr[1]**

[1] University of Vienna

**Due date:** September 2022

## Executive Summary

We present a comprehensive collection of tools that are used for gathering, processing, and analysing media text data. We employ a wide and inclusive understanding of text analysis tools and define a set of properties for categorising tools. The collection is stored in the knowledge graph we introduced in D3.1 and is therefore linked with the other inventories (e.g., news sources) created by our work package. The tool collection contains 305 entries, is continuously reviewed and updated, and is available at *Meteor* (https://meteor.opted.eu/).

## 1 Introduction

The development and application of computational methods and especially automated content analysis methods is becoming an established branch in the social sciences. Communication science and political science departments have founded research groups specialised in computational social science or even founded new chairs or professorships (Geise & Waldherr, 2021), new journals dedicated to the topic were launched (van Atteveldt et al., 2019a), and interest groups of important conferences have become institutionalised as "permanent divisions" (Theocharis & Jungherr, 2021; Geise & Waldherr, 2021).

Computational methods for text analysis are implemented with software and digital resources. Often these resources are adopted from software that has already been made available by other related disciplines (e.g., computer science, computational linguistics). In other cases, the tools are specifically developed by and for the various branches of computational social sciences (Boumans & Trilling, 2015, p.17).

There is no shortage of places to stumble upon new or already established tools, but it is difficult for researchers to keep an overview of them. For example, the *Comprehensive R Archive Network* (CRAN) lists over 18,000 packages[1] and the *Python Package Index* (PyPI) even more with around 350,000[2]. Therefore, tool presentation slots at scientific conferences and tool workshops have become established formats. Journals prepare special issues on tool demonstrations (Bleicher et al, 2021; Strippel et al., 2022) and Twitter is a popular social media platform to receive tool recommendations for specific operations. There are also other structured curation efforts[3] such as the list provided by Hepp et al., (2021, p. 4), or the CRAN Task view for Natural Language Processing[4].

Currently available databases for text analysis tools are a) often specialised to certain tasks such as data collection, b) include only a limited number of tools, c) do not allow for community contributions, or d) are designed for other target groups. A comprehensive tool database that includes the most relevant tools for text analysis and that offers search filters customised for the search requests of social scientists is missing so far. We argue that such an infrastructure can support the work with text analysis in multiple ways: First, a tool database will help potential researchers (i.e., end-users) to find available tools and to keep track of new developments. Second, a tool database can also help creators of tools, because it can increase their visibility and stimulate citations. Moreover, tool developers can make more informed decisions on whether an off-the shelf tool can be used as it is, whether it can serve as a baseline, or whether a completely new tool development is necessary (van Atteveldt et al, 2019b). Third, a research infrastructure can collect the work of many different research teams at one platform and thus improve software sustainability (Kuchinke et al., 2016). Ideally, it helps to "move to a culture of sharing and reusing tools" (van Atteveldt et al. 2018, p.88) and to "stimulate the maintenance and documentation of tools" (van Atteveldt et al. 2018, p.88).

[1] https://cran.r-project.org/web/packages/

[2] https://pypi.org/

[3] There are numerous lists available online, to name a few more: https://corpus.tools, https://tapor.ca, https://github.com/Leibniz-HBI/Social-Media-Observatory/wiki

[4] https://cran.r-project.org/web/views/NaturalLanguageProcessing.html

To promote sharing and documentation practices in the best possible way, we have built a tool database for computational social science and political communication. We integrated the tool collection in *Meteor* (https://meteor.opted.eu/) where users can not only search and browse the database, but also contribute to it. In this deliverable we present its scope, the properties documented for each tool, and a summary of the tools included so far.

## 2 Scope of Collection

There is a great variety of available tools: some are highly specialised and can only perform one operation, others are a one-stop solution and offer a great number of features. Researchers querying our tool collection will get a holistic overview of available tools and can make informed decisions on choosing the "right" tool for the task. Tools come in various shapes and formats such as software packages, digital resources, or online services. We decided to employ a wide and inclusive understanding of tools and expanded the original scope of the deliverable. Instead of limiting the tools to specialised operations mainly useful for journalistic texts (WP3), the now presented collection includes tools that are likewise relevant for text types covered in WP2, WP4, WP5 and beyond, which is arguably of greater use for a large-scale research infrastructure like OPTED

More precisely, the collection contains tools that are used for **gathering, processing, analysing, and visualising text data**.

- **Gathering text data** goes beyond querying digital archives. Researchers of journalistic mass mediated texts collect them from a variety of sources: printed material, social media, websites, and so forth. All these collection methods require different specifications for the used software. For instance, mining data from social media platforms is often only possible with proper API access, which in turn requires the tool to have some server request routines in place and also often comes with a data storage solution. Digitising printed materials requires a completely different set of procedures such as optical character recognition and considerations such as the supported languages become critical.
- **Text processing** includes a diverse set of operations and range from converting text from one data format to another, translation, removing features from the text (e.g., punctuation, URLs), over keyword extraction, to dependency parsing.
- **Data analysis** is also an encompassing term, which includes applying rule-based methods to a corpus (e.g., dictionary analysis), or employing topic models.
- **Visualisation** of text data such as word clouds do not only help to present the findings, but also to interpret them. For example, plots of word-co-occurrence networks assist researchers to make sense of the vast amount of data.

The collection is confined to tools that are used for analysing **text** data and are **readily available**. Consequently, it does not include tools that are mainly used for other types of data (e.g., numerical data), but are sometimes applied to the results of a content analysis (e.g., linear regression, t-test). Furthermore, we also exclude tools which are still in development and have no working release yet (e.g., beta or preview versions are excluded).

As a result, ***Meteor* covers text analysis tools for a variety of text analysis tasks for a diverse group of text types**.

## 3 Tool Properties

The tool collection is stored in a knowledge graph that allows for complex and interlinked data structures, and also enables browsing and querying the data flexibly. Analog to D3.1 we define a list of properties to systematically describe the tools. They mainly consist of meta information, but also include details on the **operations** the tool can perform and the **concepts** that it can measure.

Most meta information is optional and the only required fields for every tool are **name**, **authors**, **URL**, **platform**, and the performed **operation**. These five are the minimum to identify and describe a tool. A variety of properties are included to describe a tool's usability (e.g., programming languages, graphical user interface, supported input formats). We also added properties that make it more convenient for scholars to find tools in various standardised and open software repositories. This also includes GitHub since it is becoming more and more popular among researchers. The complete list of properties is described in Table 1.

**Table 1:** Properties of tools

| Property | Description |
|---|---|
| **Name** | Name of the tool. |
| **Authors** | Authors of the tool. |
| **URL** | Link to the tool, such as a project website or software repository. |
| **Platform** | Operating systems that the tool supports: Windows, Linux, and MacOS. |
| **Used For** | Operation(s) the tool can perform (growing and curated list of operations). |
| Published | Year of publication. |
| Last Updated | Most recent date of update from one version to another. |
| Version | Current version as entered in the tool collection. |
| Last Activity | Last time of activity in repository (e.g., GitHub, PyPI, or CRAN). |
| Description | Description as provided by the authors. |
| Concepts | Concepts that the tool can measure (growing and curated list of concepts). |
| DOI | DOI to the tool or its major publication. |
| arXiv | ID in www.arxiv.org. |
| CRAN | Package name in the Comprehensive R Archive Network (CRAN). |
| PyPi | Package name in Python Package Index (PyPi). |
| GitHub | Repository on www.github.com. |
| Programming Languages | Programming languages that are used for the tool or can directly interface with it (list of 37 most common programming languages, see Appendix). |
| Open Source | Source code of the tool is open to the public (yes / no / NA). |
| License | License used for distributing the software. |
| User Access | Requirements to use the tool (Free / Registration / Upon Request / Purchase). |
| Graphical User Interface | Tool provides some form of Graphical User Interface (yes / no). |
| Channels | Tool is optimised for specific Channels as in D3.1 (Print, Website, Facebook, Twitter, Instagram, Telegram, Transcripts, VKontakte). |
| Language Independent | Tool is independent of languages (yes / no). |
| Languages | Languages the tool supports (list of languages as specified by ISO 639-2). |
| Input File Format | Accepted input file formats (growing and curated list of file formats). |
| Output File Format | File formats the tool can output (see above). |
| Author Validated | Authors of the tool reported some form of validation for the tool (yes / no / NA). |

| | |
|---|---|
| Validation Corpus | Corpus that was used for validating the tool (link to inventory entry for corpora). |
| Materials | Links to additional materials that help using the tool (e.g., tutorials, documentation, scientific publications). |
| Defunct | Indicate if the tool is no longer maintained or ended its lifecycle (yes / no). |

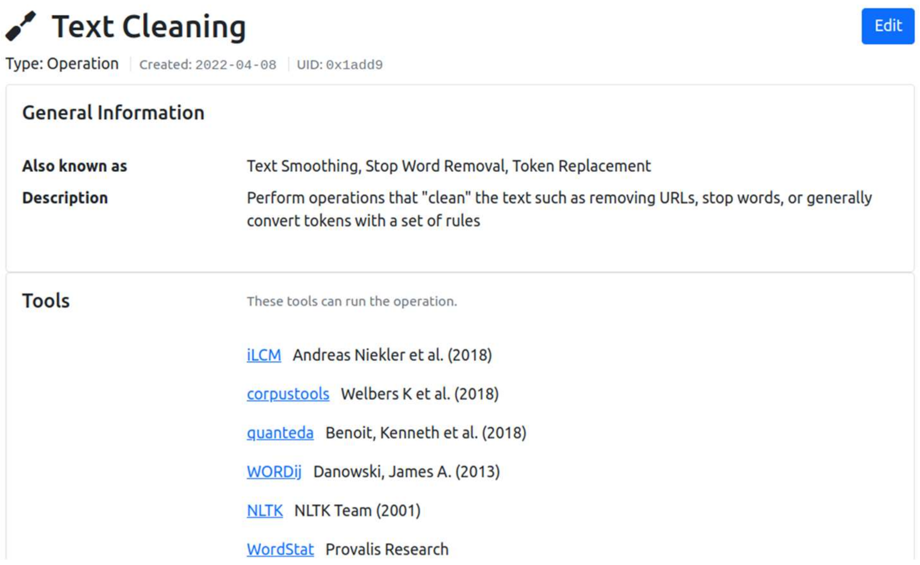
Researchers often develop software for specific applications and present their tools in scholarly publications. For such cases we provide data fields to link to the corresponding publication (e.g., via a DOI), which makes it more convenient to cite the corresponding publication as well.

The **list of operations and concepts are dynamically added to the knowledge graph**. Instead of starting with a finite list of possible operations tools could perform, we are using a bottom-up approach. When a tool is added to the collection that can perform a "new" operation (i.e., not yet part of the knowledge graph) the operation is added as well. In the course of an internal review process the operations are then verified by experts. The verification includes adding a short description to the operation and also adding synonyms to the operation. The reviewer also checks whether the operation is already listed in *Meteor*. This prevents duplicated operations or very similar kinds of operations from appearing in the collection. This process also includes subsuming very similar or closely related operations in one label. For example, the operations to remove stop words and to replace numbers with words ("100" to "one hundred") are summarised as the operation "text cleaning" (see Figure 1). Another advantage of this approach is that the operations can continuously be evaluated and improved, based on feedback and input by the researcher community.

The same logic applies to **concepts** a tool can measure, where we aim to balance between granularity of concepts and not having too many concepts in the collection. We understand concepts as theoretical constructs that researchers aim to measure or infer from texts. We acknowledge that the specific definitions for some concepts may differ among scholars to a great extent. For example, "sentiment" is a construct that is frequently measured in text analysis, but it also has various definitions (Hase, 2021). In such cases, the expert reviewers add alternative names to concepts as well as a description text that briefly notes the heterogeneity of the concept (see Figure 2).

Finally, **file formats** are also added with a bottom-up approach, since there is an abundance of file extensions, which in turn are used by several programs with different implementations (e.g., the extension .sav is used by SPSS as well as a range of other programmes as binary data storage). Therefore, we can ensure the file extensions are disambiguated by the reviewers, and also do not add an unnecessary long list of file formats and extensions to the inventory.

**Figure 1:** Example of an operation



## Text Cleaning

Edit

Type: Operation | Created: 2022-04-08 | UID: 0x1add9

### General Information

**Also known as** Text Smoothing, Stop Word Removal, Token Replacement

**Description** Perform operations that "clean" the text such as removing URLs, stop words, or generally convert tokens with a set of rules

### Tools

These tools can run the operation.

iLCM Andreas Niekler et al. (2018)

corpustools Welbers K et al. (2018)

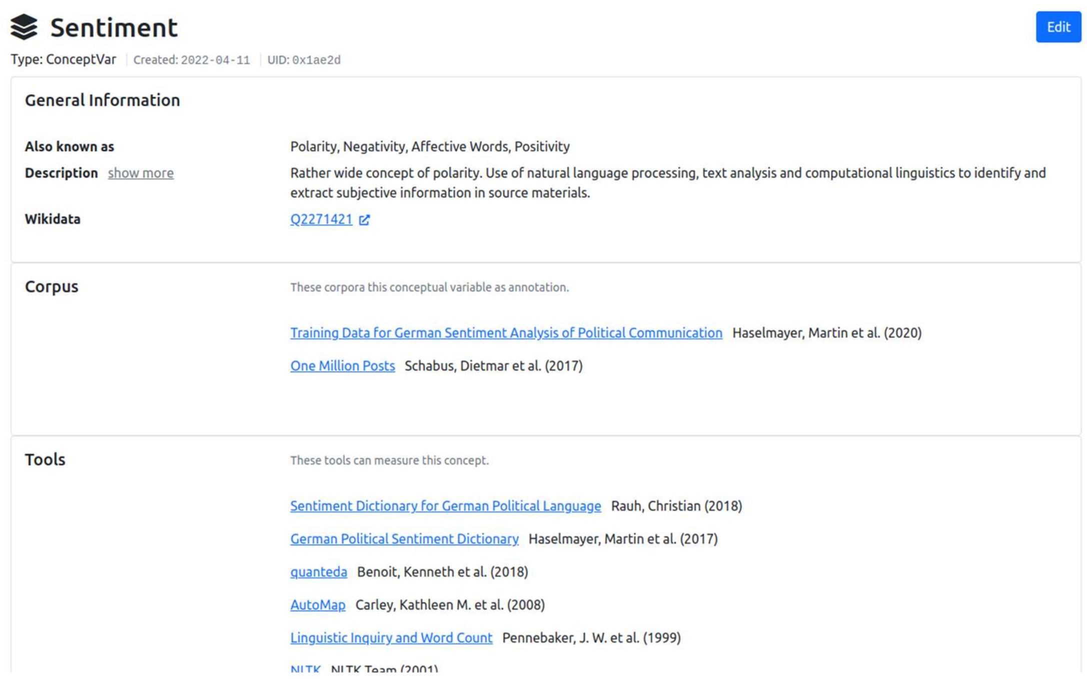
quanteda Benoit, Kenneth et al. (2018)

WORDij Danowski, James A. (2013)

NLTK NLTK Team (2001)

WordStat Provalis Research

**Figure 2:** Example of a concept variable



Sentiment

Edit

Type: ConceptVar | Created: 2022-04-11 | UID: 0x1ae2d

General Information

|  |  |
|---|---|
| **Also known as** | Polarity, Negativity, Affective Words, Positivity |
| **Description** show more | Rather wide concept of polarity. Use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials. |
| **Wikidata** | Q2271421 |

Corpus

These corpora this conceptual variable as annotation.

Training Data for German Sentiment Analysis of Political Communication Haselmayer, Martin et al. (2020)

One Million Posts Schabus, Dietmar et al. (2017)

Tools

These tools can measure this concept.

Sentiment Dictionary for German Political Language Rauh, Christian (2018)

German Political Sentiment Dictionary Haselmayer, Martin et al. (2017)

quanteda Benoit, Kenneth et al. (2018)

AutoMap Carley, Kathleen M. et al. (2008)

Linguistic Inquiry and Word Count Pennebaker, J. W. et al. (1999)

NLTK NLTK Team (2001)

## 4 Integration with D3.1

The tool collection is stored in the same knowledge graph as the collection of news sources, organisations, and data archives[5]. This allows **linking the entries based on properties** such as channels or languages. Users can query the knowledge graph based on various criteria or use full-text search to find specific tools. The system also makes it easier for researchers to add their tools to the collection later, or also to facilitate the taxonomy of operations and concepts.

A simple form allows adding a new tool and it comes with some convenience features. The user can automatically retrieve meta-information via public APIs: DOI, arXiv, GitHub, CRAN, and PyPI. Depending on the API, information such as the author of the tool, links to documentation, licensing details, etc. are retrieved.

After a new tool is entered, it is held back from public view and needs to be reviewed by an internal expert first. The review process follows the same implementation of D3.1.

## 5 Data collection

In order to get a broad coverage of text analysis tools that are currently in use, we employed several strategies. We started with an internal survey of the **OPTED consortium** to receive an initial set of seven tools. This also served to fine tune the property list (see Table 1) and exclusion criteria.

Next, we reviewed all tools **presented in the "Computational Methods Tool Demonstration"** slots in the Computational Communication Methods division from the Annual Conferences of the International Communication Association 2018-2022 (ICA). We performed a Google Scholar and Google search for the presentation titles and authors, trying to find working papers, published articles or other information about the tool. In many cases, no further information about the presentation could be found or the tools presentation was more the presentation of a prototype or process protocol but not of actually completed tools that are made ready for users. This yielded a total of 11 tools.

[5] The development is open source and available here: https://github.com/opted-eu/wp3inventory

Finally, we revisited the **dataset used in D6.1** (Living Hub for Textual Research in a Multilingual World) which systematically reviewed 854 scientific publications (see: Baden et al, 2021). We selected a subset of 406 articles which were marked to have employed some form of computer assisted text analysis. Four researchers manually reviewed the subset and extracted the tools which were used to perform the text analysis. If the publication contained supplementary materials (e.g., technical report, raw data files, replication scripts) they were downloaded and inspected as well. Among the 406 reviewed publications, 310 documented which tool they used, leaving 96 publications where the authors merely described the employed method, but not the software for performing it. As a result, we collected a **total of 292 tools**.

## 6 Summary of Collection

The following section presents an initial overview of the tools included in *Meteor*. Table 2 shows that R and Python are the most popular programming languages for tools included in *Meteor*. This is perhaps unsurprising, as it mirrors the abundance of teaching materials or workshops available for these two languages targeted toward social scientists. These languages lend themselves towards modular packages that can be combined for different workflows, while the three following (Java and C/C++) are more associated with stand-alone, multi-purpose software or high-performance computing, mainly in computer science disciplines, respectively.

**Table 2:** Programming languages of tools in Meteor, by frequency

| | Programming Language | Tools included | |
|---|---|---|---|
| 1 | R | 78 | 25 % |
| 2 | Python | 76 | 24 % |
| 3 | Java | 31 | 10 % |
| 4 | C | 20 | 6 % |
| 5 | C++ | 15 | 4 % |
| 6 | JavaScript | 10 | 3 % |
| 7 | PHP | 7 | 2 % |
| 8 | Ruby | 7 | 2 % |
| 9 | C# | 6 | 1 % |
| 10 | Perl | 5 | 1 % |

Regarding the operations for which the tools can be used, dictionaries for text classification are the most popular (Table 3). This might be explained by their flexibility because of their independence of specific programmes and by the fact that substantial research knowledge, but not programming knowledge, is required to create them. This is followed by libraries for data acquisition from APIs provided by companies. The high number is perhaps not a surprise, given the wide variety and life cycles of social media platforms in particular, and volatility of company policies on the other hand that require regular software updates to comply with API changes. It appears that developers do not always keep up with these changes, because 15 tools in our collection that can gather data from APIs are actually defunct, lowering the effective number of such tools to 27.

**Table 3:** Operations provided by tools in Meteor, by frequency

| Operation | | Tools capable of |
|---|---|---|
| 1 | Dictionary Resource | 66 |
| 2 | API Access | 42 |
| 3 | Document Scoring | 38 |
| 4 | POS-Tagging | 31 |
| 5 | Word Frequencies & Corpus Statistics | 29 |
| 6 | Supervised Machine Learning | 26 |
| 7 | Tokenization | 26 |
| 8 | Text Cleaning | 24 |
| 9 | Network Analysis | 23 |
| 10 | Data Scraping | 22 |
| 11 | Topic Modelling | 22 |
| 12 | Named Entity Recognition | 21 |
| 13 | Document Classification | 14 |
| 14 | Document Similarity Scoring | 14 |
| 15 | Word Embedding | 14 |
| 16 | Dictionary Analysis | 13 |
| 17 | Lemmatization | 13 |
| 18 | Data Visualization | 12 |
| 19 | Dependency Parsing | 12 |
| 20 | Stemming | 12 |

Over 75% (n = 230) tools included in *Meteor* are freely accessible to researchers or only require registration before they can be used without charge. Paid services make up only a small selection of the available tools (n = 38), while another 19 tools are available upon request from their authors. This positions *Meteor* as an accessible resource for both academics as well as more generally interested users because most listed programmes can be explored without any financial hurdle. Together with the training materials provided later in OPTED (e.g., Deliverable D3.4), this ensures equitable access to software tools.

Of the freely accessible tools, more than 77% (n = 179) are open source. This fulfils the previously stated goal that *Meteor* might contribute to a vibrant ecosystem of application development by highlighting tool diversity as well as their source code. It encourages developers to contribute to respective packages.

Despite the data collection criteria, which due to the consideration of research papers published a few years ago, might have turned up outdated software, over 90% of the tools included in *Meteor* are actively developed or usable in 2022. *Meteor* is therefore not a software archive, but a living inventory of relevant and current tools. Of the tools that are outdated and no longer functioning, more than half are designed to access APIs, perhaps challenged by their strong dependence on external company policies.

In terms of accessibility, nearly 40% (n = 121) of tools included are independent of languages and therefore suitable for research purposes in any European language. Yet, the greater share, specifically dictionaries, only support certain languages. They make up more than a third of all language dependent tools (n = 64). Table 4 shows the pronounced inequalities in availability of computational tools according to language. This is one of the challenges tackled by WP6, with recommendations and standards for multilingual text analysis being developed to improve the research landscape. *Meteor* aims to provide a spotlight also for tools in less represented languages, to encourage more research (e.g., the construction of dictionaries in more languages) and to diversify the methodological toolkit.

## 7 Conclusion

This deliverable introduced the tool database as a central component of *Meteor*. The size and scope of the current state of the collection is a manifestation of the significance that software tools have gained within the field of computational approaches to political communication over the last years. Not only relevant for journalistic texts but useful for various text types studied by social scientists or data journalists, the tool database will help its users to find, compare, and assess the most suited tools for specific tasks. Viewed from another perspective, with *Meteor* tool developers gain a fast overview about related tools in the field and benefit from a new platform to disseminate their tools to users. Overall, the tool database will foster the sustainability of text analysis software (Kuchinke et al., 206; Strippel, 2021).

Looking ahead and inspecting the development and use of tools in social science more generally, a new database is only one important aspect. Additional efforts are necessary to foster the continued development, improvement and evaluation of high-quality tools. *Meteor* and OPTED can contribute further to an environment that supports the sustainability of research tools. Encouraging interdisciplinary standards in giving credit for the use of tools (Howison & Bullard, 2016) or the building of communities for the sustainability of software (Katz, 2018) are just two examples.

**Table 4:** Languages supported by language-specific tools in *Meteor*, by frequency

| Language | | Number of tools | | Language | | Number of tools | |
|---|---|---|---|---|---|---|---|
| 1 | English | 96 | 52 % | 16 | Hebrew | 15 | 8 % |
| 2 | German | 41 | 22 % | 17 | Indonesian | 15 | 8 % |
| 3 | French | 29 | 15 % | 18 | Polish | 15 | 8 % |
| 4 | Italian | 27 | 14 % | 19 | Finnish | 14 | 7 % |
| 5 | Spanish | 25 | 13 % | 20 | Korean | 14 | 7 % |
| 6 | Dutch | 25 | 13 % | 21 | Turkish | 14 | 7 % |
| 7 | Portuguese | 22 | 11 % | 22 | Czech | 13 | 7 % |
| 8 | Russian | 22 | 11 % | 23 | Hindi | 13 | 7 % |
| 9 | Swedish | 18 | 9 % | 24 | Lithuanian | 13 | 7 % |
| 10 | Arabic | 17 | 9 % | 25 | Norwegian | 13 | 7 % |
| 11 | Danish | 17 | 9 % | 26 | Catalan | 12 | 6 % |
| 12 | Chinese (Simplified) | 17 | 9 % | 27 | Romanian | 12 | 6 % |
| 13 | Hungarian | 16 | 8 % | 28 | Slovenian | 12 | 6 % |
| 14 | Japanese | 16 | 8 % | 29 | Estonian | 11 | 5 % |
| 15 | Greek | 15 | 8 % | 30 | Croatian | 11 | 5 % |

## References

Baden, C., Pipal, C., Schoonvelde, M., van der Velden, M. A., (2021). Three Gaps in Computational Text Analysis Methods for Social Sciences: A Research Agenda. *Communication Methods and Measures 16*(1), 1-18. DOI: 10.1080/19312458.2021.2015574

Boumans, J. W., & Trilling, D. (2016). Taking stock of the toolkit: An overview of relevant automated content analysis approaches and techniques for digital journalism scholars. *Digital Journalism*, *4*(1), 8-23.

Geise, S. & Waldherr, A. (2021). Computational communication science: Lessons from working group sessions with experts of an emerging research field. In U. Engel, A. Quan-Haase, S. X. Liu, Su & L. Lyberg (Eds.), *Handbook of computational social science. Volume 1: Theory, case studies and ethics* (pp. 66-82). Routledge. DOI: 10.4324/9781003024583

Hase, V. (2021). Sentiment/tone (Automated Content Analysis). *DOCA - Database of Variables for Content Analysis*. DOI: 10.34778/1d

Hepp, A., Hohmann, F., Belli, A., Boczek, K., Haim, M., Heft, A., Jünger, J., Jürgens, P., Koenen, E., von Nordheim, G., Rinsdorf, Lars, Rothenberger, L., Schatto-Eckrodt, Tim &Unkel, J. (2021). Forschungssoftware in der Kommunikations- und Medienwissenschaft: Stand, Herausforderungen und Perspektiven. DGPuK Positionspapier. https://www.dgpuk.de/sites/default/files/DGPuK%20Positionspapier%20-%20Forschungssoftware%20in%20der%20Kommunikations-%20und%20Medienwissenschaft_0.pdf

Howison, J., & Bullard, J. (2016). Software in the scientific literature: Problems with seeing, finding, and using software mentioned in the biology literature. *Journal of the Association for Information Science and Technology*, *67*(9), 2137-2155.

Katz, D. D. (2016). Sustainable software for science. Paper presented at agstuhl Perspectives Workshop 16252 "Engineering Academic Software".

Kuchinke, W., Ohmann, C., Stenzhorn, H., Anguista, A., Sfakianakis, S., Graf, N., & Demotes, J. (2016). Ensuring sustainability of software tools and services by cooperation with a research infrastructure. *Personalized Medicine*, *13*(1), 43-55.

Bleicher, J. K., Hasebrink, U., Herzog, A., Hölig, S., Kettemann, M. C., Lampert, C., Loosen, W., Schmidt, J.-H., Schröder, H.-D., Schulz W., Wagner, H.-U., Wiedemann, G. (2021): Call for Papers für neue Rubrik Software-Rezensionen [Call for Papers for the new section software reviews]. *Medien & Kommunikationswissenschaft*, 4.

Strippel, C. (2021). Forschungsinfrastrukturen für die Kommunikations-und Medienforschung im deutschsprachigen Raum. Initiativen, Bedarfe und Perspektiven. *M&K Medien & Kommunikationswissenschaft*, *69*(1), 136-157.

Strippel, C., Breuer, J., Fürst, S., Koenen, E., Prandner, D., & Schwarzenegger, C. (2022). Call for Papers - Special Issue "Data, Archives and Tools: Infrastructures and Resources for Communication and Media Research" - Deadline: 15 August 2022. https://www.springer.com/journal/11616/updates/20566430

van Atteveldt, W., Margolin, D., Shen, C., Trilling, D., & Weber, R. (2019a). A roadmap for computational communication research. *Computational Communication Research*, *1*(1), 1-11.

van Atteveldt, W. Strycharz, J., Trilling, D., Welbers, K. (2019b). Computational Communication Science. Toward Open Computational Communication Science: A Practical Road Map for Reusable Data and Code. *International Journal of Communication*, 13, 3935–3954.

van Atteveldt, W., & Peng, T. Q. (2018). When communication meets computation: Opportunities, challenges, and pitfalls in computational communication science. *Communication Methods and Measures*, *12*(2-3), 81-92.

## Appendix

### List of Programming Languages

- APL
- Assembly
- Bash/Shell
- C
- C#
- C++
- COBOL
- Clojure
- Crystal
- Dart
- Delphi
- Elixir
- Erlang
- F#
- Go
- Groovy
- Haskell
- Java
- JavaScript
- Julia
- Kotlin
- LISP
- Matlab
- Node.js
- Objective-C
- PHP
- Perl
- PowerShell
- Python
- R
- Ruby
- Rust
- SQL
- Scala
- Swift
- Stata Script (Ado)
- TypeScript
- VBA