# Generic Web Content Extraction with Open-Source Software

Adrien Barbaresi



Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019)


# Generic Web Content Extraction with Open-Source Software

**Adrien Barbaresi**
Center for Digital Lexicography for the German Language (ZDL)
Berlin-Brandenburg Academy of Sciences (BBAW)
Jägerstraße 22/23 D-10117 Berlin
barbaresi@bbaw.de

## Abstract

Web corpus construction involves numerous design decisions. The software packages presented here can help facilitate collection and enhance corpus quality.

## 1 Problem description

Large "offline" web corpora are now standard throughout disciplines among the research community. Corpus construction notably involves "crawling, downloading, 'cleaning' and de-duplicating the data, then linguistically annotating it and loading it into a corpus query tool." (Kilgarriff, 2007) As such, this process involves a significant number of design decisions and turning points in data processing. Depending on the purpose of data collection, a substantial filtering and quality assessment may also be needed. While some large-scale algorithms can be expected to smooth out irregularities, uses requiring a low margin of error as well as close reading approaches imply constant refinements and improvements in the constitution of the dataset and its processing, for example in the context of an aggregated lexical information platform (Geyken et al., 2017).

Recently, approaches using the CommonCrawl[1] have flourished as they allow for faster download and processing by skipping (or more precisely outsourcing) the crawling phase. Barring the fact that finding one's "own" way through the Web can be preferable, it is clear that such data should not be used without some filtering. Corresponding to the potential lack of metadata is a lack of information regarding the content, whose adequacy, focus and quality are the object of a post hoc evaluation (Baroni et al., 2009). Because of the vastly increasing variety of corpora, text types and use cases, it becomes more and more difficult to assess the usefulness and appropriateness of certain web texts for given research objectives. Most notably, an essential operation in corpus construction consists in retaining the desired content while discarding the rest, a polyonymous task referring to peculiar subtasks or to the whole, most notably web scraping, boilerplate removal, web page cleaning, or web content extraction (Lejeune and Zhu, 2018).

Consequently, a significant challenge lies in the ability to extract and pre-process web data to meet scientific expectations with respect to corpus quality (Barbaresi, 2019b). In the following, two libraries grounding on previous efforts (Barbaresi, 2016) are presented which can help enhancing the quality of webcorpora. They are both relying on Python, currently one of the most used programming languages, within and outside of academia.[2]

## 2 htmldate: finding the publishing date

The *htmldate* library (Barbaresi, 2019a) can find both the original and the updated publication dates of web pages. It involves a rule-based examination of the semantic structure of HTML documents, using a combination of tree traversal, common structural patterns, text-based heuristics and robust date extraction. First, it uses the markup in the document header, where common patterns are used to identify relevant elements (e.g. *link* and *meta* elements) including common standards and idiosyncracies of content management systems. Second, it looks for cues within the HTML code as the whole document is searched for structural markers: *abbr/time* elements and a series of attributes (e.g. *postmetadata*). Finally, a series of heuristics is run on text and markup. The library currently focuses on texts written in English and German, it is used in production and is documented online.[3]

[1] https://commoncrawl.org

[2] Python Software Foundation, http://www.python.org https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019

[3] https://github.com/adbar/htmldate

## 3 trafilatura: targeting the main content

The second software component focuses on the main content, which is usually the part displayed centrally, without the left or right bars, the header or the footer, but including potential titles and comments. Distinguishing between whole page and essential parts can help to alleviate many quality problems related to web texts. While this is particularly useful for de-duplication, other tasks related to content extraction also benefit from a cleaner text base. In the concrete case of linguistic and lexicographic research, it allows for content checks on the only portion of the document that really counts.

Although most corresponding Python modules are not actively maintained, the following alternatives perform similar tasks: *dragnet*[4] features combined and machine-learning approaches, but requires many dependencies as well as extensive tuning; *python-readability*[5] cleans the page and preserves some markup but is mostly geared towards news texts; *html2text*[6] converts HTML pages to Markup language and thus keeps the structure, but it doesn't focus on main text extraction. Another issue resides in the lack of output formats corresponding to corpus linguists' needs for document storage and processing, e.g. XML formats such as TEI/XML following the recommendations of the Text Encoding Initiative.[7]

The *trafilatura* library (Barbaresi, 2019c) scrapes the main text of web pages while preserving some structure, which is equivalent to boilerplate removal, DOM-based content extraction, main content identification, and HTML text cleaning. The extraction focuses on original text and can help with the noise consisting of recurring elements (headers and footers, ads, links/blogroll, etc.). It has to be precise enough not to miss texts or discard valid documents, it also has to be reasonably fast, as it is expected to run in production on millions of documents. The processing result can be in plain text or XML format. In the latter case, basic formatting elements are preserved such as text formatting (bold, italic, etc.) and page structure (paragraphs, titles, lists), which can be used for further processing.

This is work in progress[8], currently experimental features include the extraction of comments (separated from the rest), duplicate detection at sentence, paragraph and document level using a least recently used (LRU) cache, TEI/XML output, and language detection on the extracted content.

[4]https://github.com/dragnet-org/dragnet
[5]https://github.com/buriy/python-readability
[6]https://github.com/Alir3z4/html2text
[7]https://tei-c.org
[8]https://github.com/adbar/trafilatura

## 4 Conclusions

This ongoing work constitutes a step towards the ability to extract and pre-process web texts in order to make them available in clearly definable and coherent collections. In both software components presented here, all the operations needed from web page download to HTML parsing are handled, including scraping and textual analysis. URLs, HTML files or parsed HTML trees are given as input and the libraries output strings in the desired format. They can be used on common operating systems, by themselves, within Python, or on the command-line. Their versatility allows for work on different languages and corpus types as well as for inclusion in various processing chains.

## References


Adrien Barbaresi. 2016. Efficient construction of metadata-enhanced web corpora. In *Proceedings of the 10th Web as Corpus Workshop, Annual meeting of the ACL 2016*, pages 7–16. Association for Computational Linguistics.

Adrien Barbaresi. 2019a. htmldate. https://doi.org/10.5281/zenodo.3459599.

Adrien Barbaresi. 2019b. The Vast and the Focused: On the need for thematic web and blog corpora. In *Proceedings of the CMLC-7 workshop*, pages 29–32.

Adrien Barbaresi. 2019c. trafilatura. https://doi.org/10.5281/zenodo.3460969.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky Wide Web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Alexander Geyken, Adrien Barbaresi, Jörg Didakowski, Bryan Jurish, Frank Wiegand, and Lothar Lemnitzer. 2017. Die Korpusplattform des "Digitalen Wörterbuchs der deutschen Sprache" (DWDS). *Zeitschrift für germanistische Linguistik*, 45(2):327–344.

Adam Kilgarriff. 2007. Googleology is bad science. *Computational Linguistics*, 33(1):147–151.

Gaël Lejeune and Lichao Zhu. 2018. A New Proposal for Evaluating Web Page Cleaning Tools. *Computación y Sistemas*, 22(4).