

**HARNESSING WEB ARCHIVES
TO TACKLE SELECTED SOCIETAL CHALLENGES**

by

Johannes Kiesel

Dissertation to obtain the academic degree of
Dr. rer. nat.

Faculty of Media
Bauhaus-Universität Weimar
Germany

Advisor: Prof. Dr. Benno Stein
Reviewer: Prof. Dr. Arjen P. de Vries
Date of oral exam: June 3, 2022

—TO ALL WHO TRUSTED ME.

WHAT YOU HAVE DISCOVERED IS A RECIPE NOT FOR MEMORY, BUT FOR REMINDER.
AND IT IS NO TRUE WISDOM THAT YOU OFFER YOUR DISCIPLES, BUT ONLY THE SEM-
BLANCE OF WISDOM, FOR BY TELLING THEM OF MANY THINGS WITHOUT TEACHING
THEM YOU WILL MAKE THEM SEEM TO KNOW MUCH WHILE FOR THE MOST PART THEY
KNOW NOTHING.

—SOCRATES ON WRITING, AS PER WRITINGS BY PLATO, ~370 BCE
OXFORD WORLD'S CLASSICS: PLATO: PHAEDRUS

Abstract

HARNESSING WEB ARCHIVES TO TACKLE SELECTED SOCIETAL CHALLENGES

With the growing importance of the World Wide Web, the major challenges our society faces are also increasingly affecting the digital areas of our lives. Some of the associated problems can be addressed by computer science, and some of these specifically by data-driven research. To do so, however, requires to solve open issues related to archive quality and the large volume and variety of the data contained.

This dissertation contributes data, algorithms, and concepts towards leveraging the big data and temporal provenance capabilities of web archives to tackle societal challenges. We selected three such challenges that highlight the central issues of archive quality, data volume, and data variety, respectively: (1) For the preservation of digital culture, this thesis investigates and improves the automatic quality assurance of the web page archiving process, as well as the further processing of the resulting archive data for automatic analysis. (2) For the critical assessment of information, this thesis examines large datasets of Wikipedia and news articles and presents new methods for automatically determining quality and bias. (3) For digital security and privacy, this thesis exploits the variety of content on the web to quantify the security of mnemonic passwords and analyzes the privacy-aware re-finding of the various seen content through private web archives.

Abstract (in German)

HARNESSING WEB ARCHIVES TO TACKLE SELECTED SOCIETAL CHALLENGES

Mit der wachsenden Bedeutung des World Wide Webs betreffen die großen Herausforderungen unserer Gesellschaft zunehmend auch die digitalen Bereiche unseres Lebens. Einige der zugehörigen Probleme können durch die Informatik, und einige von diesen speziell durch datengetriebene Forschung, angegangen werden. Dazu müssen jedoch offene Fragen im Zusammenhang mit der Qualität der Archive und der großen Menge und Vielfalt der enthaltenen Daten gelöst werden.

Diese Dissertation trägt mit Daten, Algorithmen und Konzepten dazu bei, die große Datenmenge und temporale Protokollierung von Web-Archiven zu nutzen, um gesellschaftliche Herausforderungen zu bewältigen. Wir haben drei solcher Herausforderungen ausgewählt, die die zentralen Probleme der Archivqualität, des Datenvolumens und der Datenvielfalt hervorheben: (1) Für die Bewahrung der digitalen Kultur untersucht und verbessert diese Arbeit die automatische Qualitätsbestimmung einer Webseiten-Archivierung, sowie die weitere Aufbereitung der dabei entstehenden Archivdaten für automatische Auswertungen. (2) Für die kritische Bewertung von Information untersucht diese Arbeit große Datensätze an Wikipedia- und Nachrichtenartikeln und stellt neue Verfahren zur Bestimmung der Qualität und Einseitigkeit/Parteilichkeit vor. (3) Für die digitale Sicherheit und den Datenschutz nutzt diese Arbeit die Vielfalt der Inhalte im Internet, um die Sicherheit von mnemonischen Passwörtern zu quantifizieren, und analysiert das datenschutzbewusste Wiederauffinden der verschiedenen gesehenen Inhalte mit Hilfe von privaten Web-Archiven.

Ehrenwörtliche Erklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Teile der Arbeit, die bereits Gegenstand von Prüfungsarbeiten waren, sind ebenfalls unmissverständlich gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt. Ich versichere, dass ich nach bestem Wissen die reine Wahrheit gesagt und nichts verschwiegen habe.

Weimar, 11. Januar 2022



Johannes Kiesel

Contents

1	INTRODUCTION	1
1.1	Societal Challenges Discussed in this Thesis	3
1.2	The Data Analysis Pipeline in the Thesis	11
1.3	Thesis Structure	13
2	THE TECHNOLOGY OF WEB ARCHIVES	15
2.1	The Web Archive Ecosystem	15
2.2	Archiving for Reproducibility: The Webis Web Archiver	17
2.3	Summary	27
3	HARNESSING WEB ARCHIVES TO PRESERVE DIGITAL CULTURE	29
3.1	Assessment of Web Page Reproduction Quality	30
3.2	Analysis of Unexpected Content in Web Archives	48
3.3	Segmentation of the Pages in Web Archives	56
3.4	Summary	86
4	HARNESSING WEB ARCHIVES TO CRITICALLY ASSESS INFORMATION	87
4.1	Identification of Vandalism on Wikipedia	88
4.2	Identification of Hyperpartisan News	107
4.3	Summary	133
5	HARNESSING WEB ARCHIVES FOR ONLINE SECURITY AND PRIVACY	135
5.1	Security Estimate for Mnemonic Passwords	136
5.2	WASP for Privacy-Aware Information Re-Finding	165
5.3	Summary	179
6	CONCLUSION	181
6.1	Main Contributions and Implications	181
6.2	Open Problems and Future Work	184
	REFERENCES	186

1

Introduction

The World Wide Web is, without doubt, the most extensive public sphere of today. The number of people being able to participate in this sphere is astonishing: Nearly 60% of the world's population are Internet users.¹ The rate at which people collectively contribute to this sphere is staggering, as Table 1.1 illustrates. Moreover, as part of mutually reinforcing dynamics, the World Wide Web is also deeply embedded in most societies. For example, online newspapers—as online versions of classical public sphere media—are located in at least 80% of earth's states.² In its mere three decades of existence, the web indeed became worldwide, yet still grows in importance.

By virtue of its importance, the World Wide Web also pertains to some of today's societal challenges. The societies we live in continuously face challenges in many aspects of life, for example with respect to public health, environmental protection, or security. For some of such challenges, the World Wide Web's data is essential for facing it. A prime example is the challenge of reflection, which is to better understand society's past. Clearly, the web provides key data to investigate on a society's past in huge amounts. Other relevant challenges are, amongst others, supporting and safeguarding people in their information behavior—much of which is online these days—or securing the web as a space of freedom while supporting democratic principles. For these challenges, the web provides a unique opportunity to study the current status of the respective problems, potentially unveiling both causes and solutions.

¹Statista. Global digital population as of January 2021. <https://perma.cc/PY4A-S5FY>

²Estimated from the number of states from Wikipedia, <https://perma.cc/4K4A-L8DG>, and the number of origins in the GDELT news archives from 2017, <https://perma.cc/2LQ9-SPUD>; These states account for more than 99% of earth's population, <https://perma.cc/PB9A-78C9>

TABLE 1.1: Examples of average content creation and modification on the World Wide Web per minute (one significant digit) reported for specific years.

Web content	Each minute	In year
Tweets on Twitter ³	500,000	2019
Stories on Instagram ⁴	300,000	2020
Photos on Facebook ⁴	100,000	2020
Posts on Tumblr ³	90,000	2019
Video minutes uploaded to YouTube ⁴	30,000	2020
Comments on Reddit ⁵	4,000	2020
News articles in some online newspaper ⁶	300	2020
Edits to Wikipedia ⁷	100	2021

Though the World Wide Web allows for large-scale analyses of the present, the web’s ephemerality necessitates the harnessing of web archives to analyze the past and make analyses replicable. According to the Internet Archive, a typical web page is expected to last about 92 days before changing, moving, or going offline.⁸ With this rapid rate of change, analyses of the past or of developments over time require the foresighted capturing of web snapshots. Moreover, without pairing web analyses with a snapshot, it is impossible to replicate or extend analyses later on. However, web analysis technologies are becoming more sophisticated, employing the rendering of the web page (as a human would see it) and interactions with it instead of the bare HTML code. To make such sophisticated analyses replicable, it is crucial to allow for re-rendering the web page after the fact. In general, two approaches exist for the necessary preservation: (1) the “classical” web archives that preserve all communication between the browser and the web for replay and (2) the preservation in a page-specific format (e.g., a database and source code dump). The latter is rarely possible but provides the highest preservation fidelity.

Altogether, this thesis contributes data, algorithms, and concepts towards leveraging the big data and provenance capabilities of web archives to tackle societal challenges. Specifically, we address open issues related to archive quality and the large volume and variety of the data contained. Archive quality is prominent for preserving digital culture, for which we contribute corpora, methods, and concepts for quality assurance and pro-

³DOMO, Data Never Sleeps 7.0, July 2019, <https://perma.cc/5T9N-GWHX>

⁴DOMO, Data Never Sleeps 8.0, August 2020, <https://perma.cc/7E2X-ZYD6>

⁵Reddit Blog, Reddit’s 2020 Year in Review, <https://perma.cc/RD46-V5WP>

⁶Based on the GDELT API, all articles in 2020, <https://perma.cc/DXK7-3DD7>

⁷Wikipedia, Statistics, August 2021 (1.9 edits every second) <https://perma.cc/5XZ9-BB99>

⁸Archive-It. About Archive-It (Video). <https://perma.cc/F56G-9E7T>

cessing. The huge volume of web data enables large-scale analyses, which are especially needed for critically assessing information, and for which we contribute corpora and methods to assess Wikipedia edits and news articles. The huge variety of web data both demands and enables new approaches, e.g., for digital security and privacy, for which we contribute corpora and methods to address the variety in privacy-aware re-finding and exploit the variety for password security estimation.

In the following, Section 1.1 details the three societal challenges discussed in this thesis and presents our top-level research questions and approaches for each challenge. Section 1.2 then provides a cross-sectional overview of these approaches in terms of the generic data analysis pipeline and Section 1.3 an outline of this thesis.

1.1 Societal Challenges Discussed in this Thesis

This thesis uses the term “societal challenge” to refer to issues that concern most if not all members of a society, either now or in a likely future. The term is taken from the European Union’s Horizon 2020 program, which describes the concept for the program’s purpose as “major concerns shared by citizens in Europe and elsewhere.”⁹ Though “society” can refer to groups of all sizes,¹⁰ for this thesis, “societal” shall have the connotation of at least a nationwide extent like in the European program.

Besides the Horizon 2020 program, several organizations have listed societal challenges to track the progress in tackling them. At the largest scale, the United Nations have announced 17 Sustainable Development Goals,¹¹ which nations and unions around the world have adopted according to their respective situation as own societal challenges. These “goals” set the agenda for the United Nations and mostly concern structural challenges. The World Economic Forum’s list of (currently) 104 global issues¹² contains not just structural, but also cultural, sociological, and technological challenges. The list is a service for staying informed on the issues, which is why each issue is associated with a curated stream of news articles. The list thus augments the World Economic Forums’ Global Risk Report¹³ that provides an annual assessment of the immediacy to tackle certain challenges and of the impact

⁹European Commission. Horizon 2020, Societal Challenges. <https://perma.cc/PWB5-Z3VB>

¹⁰Merriam-Webster dictionary. Society. <https://perma.cc/QEU3-48ZW>

¹¹United Nations, Department of Economic and Social Affairs. Sustainable Development, The 17 Goals. <https://perma.cc/C8Q6-UR59>

¹²WEF, Strategic Intelligence. Global Issues. <https://perma.cc/ML7Q-DJBA>

¹³WEF. The Global Risks Report 2021. <https://perma.cc/CZG2-V9AC>

of the failure to do so. The German Gesellschaft für Informatik's list of five grand challenges¹⁴ is focused on challenges that have Computer Science at its core. The goal of this list is to attract students to the field by showcasing Computer Science's potential impact on the society. The challenges selected for this thesis and discussed below are all taken from these lists.

This thesis discusses approaches based on web archives for three of the societal challenges from the lists mentioned above that call for large amounts of public sphere data: the preservation of digital culture (Chapter 3), the critical assessment of information (Chapter 4), and online security and privacy (Chapter 5). The following sections introduce one challenge each. The main contributions are summarized in Section 6.1.

1.1.1 Harnessing Web Archives to Preserve Digital Culture

The challenge of preserving digital culture arises from the heterogeneity, ephemerality, and volume of digital content. Though this thesis focuses on web content, digital content has become a significant part for diverse cultures even before the web, demanding its preservation. Moreover, web content reflects offline culture, but can be collected at a much larger scale [182]. The importance of preserving culture of all kinds has been recognized in many places: "Both these intangible and "tangible" (art works and monuments) heritage contain the diversity of human experience, and a framework for both individual and community identity. They nurture resilience in the face of change or crisis. It is therefore vital to protect and sustain what has been handed down to us from previous generations,"¹⁵ especially "as societies and technologies change at a faster rate than ever before."¹⁵ Also the Horizon 2020 program highlights the importance of the "cultural and social diversity and of its past [to] inform the reflection about present problems and help to find solutions for shaping Europe's future."¹⁶ Moreover, also one of the Grand Challenges of Computer Science is on (the technical problems of) preserving digital culture.¹⁷

Web archives allow to preserve digital culture as it is found on the World Wide Web, but face several of the issues mentioned above. The ease of accessing web pages hides the difficulty of preserving them for a longer period

¹⁴Gesellschaft für Informatik. Die Grand Challenges der Informatik (The Grand Challenges of Computer Science). <https://perma.cc/Q9BR-H9EW>

¹⁵WEF, Strategic Intelligence. Arts and Culture, Heritage Protection and Cultural Sustainability. <https://perma.cc/C22F-3QQ9>

¹⁶European Commission. Horizon 2020, Europe in a changing world – Inclusive, innovative and reflective societies. <https://perma.cc/BE5S-QB2B>

¹⁷Gesellschaft für Informatik. Erhalt des digitalen Kulturerbes (preservation of digital cultural heritage). <https://perma.cc/F7AR-EYWJ>

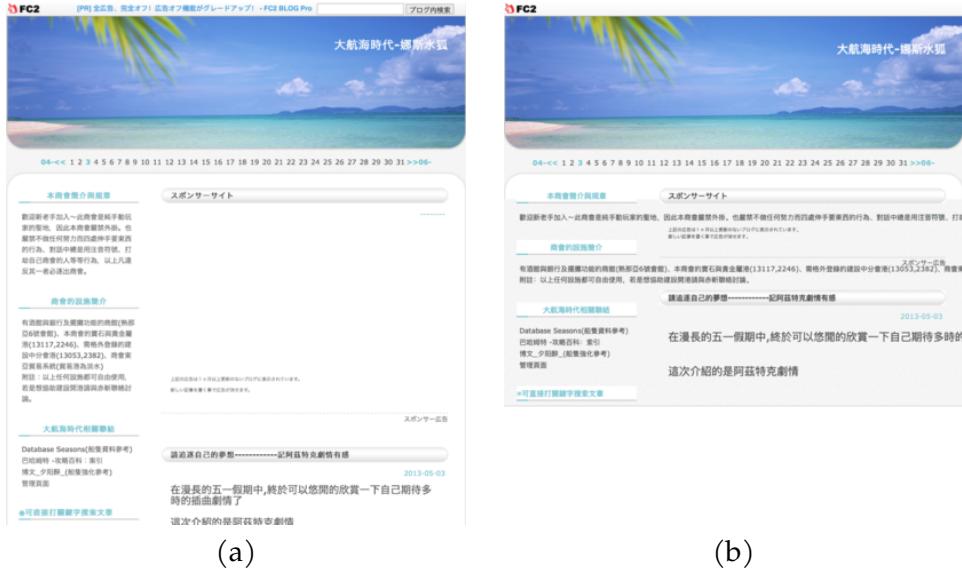


FIGURE 1.1: Rendering of the same excerpt of one web page (archived in 2017) in the then up-to-date (a) Safari Browser and (b) CSSBox rendering engine.

of time. A first problem is that the process for rendering web pages has grown considerably in technical complexity. Consider Figure 1.1: though CSSBox¹⁸ is an actively developed and widely employed software to render web pages, its rendering of what seems to be a simple web page differs considerably from that of a major browser. Another problem is that web pages are compiled from several different sources (and servers) and may request different content based on, among others, the viewer’s past interactions, the viewer’s location (often identified by IP addresses), or simply the time of day. Due to this problem, the web page one archives may be very different to the web page one expects for some specific URL [129].¹⁹ However, to assess the problems mentioned above for a web page, one has to keep in mind that web pages are not atomic objects but composed of several segments with different purposes and different levels of preservation difficulty. Research on segmentation has, however, stagnated until recently [134].

In the context of this challenge, Chapter 3 investigates the following research questions:

RQ1. How to measure web archive reproduction quality? Section 3.1 contributes the first operationalization of comparative web archive repro-

¹⁸CSSBox. About. <https://perma.cc/77EX-3F7Y>

¹⁹This problem connects web archiving to the Gesellschaft für Informatik’s Grand Challenge of “Verlässlichkeit von Software” (software reliability), <https://perma.cc/63RE-M5YZ>

duction quality. The underlying notion is that the quality of a reproduction of a web page from a web archive is higher the closer the reproduced web page is to the live web page (at the time of archiving) from the user's perspective. The presented operationalization employs human quality judgments on the differences between two screenshots of a web page, namely of the live and reproduced web page. Moreover, the section introduces the first dataset and benchmarks approaches for automating such assessment, with one approach based on a neural network from the field of computer vision reaching high correlation with human quality scores (Pearson's $r = 0.57$).

RQ2. How frequent are different kinds of unexpected content in web archives? Section 3.2 contributes a first analysis of unexpected content in general-purpose web archives, so-called web archive content errors. Complementary to reproduction quality, these errors refer to elements on the web page that do not belong to the given web page if working properly, but technical errors, crawl prevention methods, pop-ups, or other effects place them there, often replacing the content one would expect to see. The section suggests five signs of unexpected content (ads-only, loading indicators, page-sized pop-ups, access-blocking CAPTCHAs, and dominant error messages) and provides for a first statistical analysis of these by a crowd-sourced annotation of 10,000 web pages. The analysis indicates that the signs are quite common, suggesting unexpected content in ~10% of web pages.

RQ3. How to define and identify segments of a web page? Section 3.3 contributes an in-depth investigation of the task of web page segmentation. This task is critical for the preservation of web culture, yet has been neglected in the research community for more than a decade despite of the rapid developments in web technologies and web design demanding continuous updates. A usual web pages consist of several segments (e.g., advertisement, navigation, main content, related pages, or comments), which may require different methods for their preservation, are of different importance for judging reproduction quality, and may also be treated independently in retrieval (i.e., when accessing the digital culture). The section introduces a new conceptualization of web segments from the user's perspective and provides the largest human-annotated dataset for the task to date. In a benchmark of existing algorithms, the section shows that the nearly 20-year-old VIPS still performs best, reaching an F-Score of up to 0.75, but that new neural approaches have much potential for tackling the task that is yet unexplored.

1.1.2 Harnessing Web Archives to Critically Assess Information

The challenge of assessing (online) information as a private person arises from the intransparency at which information spreads on the World Wide Web. Factuality may be the first criterion for judging information that comes to mind, but, especially for non-factual information, criteria like controversiality and the source's authority are of importance. For example, Fuhr et al. [85] suggest nine different criteria for an "information nutrition label," illustrated in Figure 1.2. Though these criteria apply to all information, special considerations are needed for the World Wide Web due to its distinct cognitive attributes, especially its intransparent recommendations and personalization, lack of social cues (especially eye contact), unclear epistemic quality assurance (e.g., whether there are editors or who they are), and a tendency to evoke misperceptions about the prevalence of opinions [157, p. 20]. An analysis is thus crucial as many people gather information online.²⁰ Indeed, in terms of reaching political followers, the spread of the World Wide Web has been especially beneficial for populist parties [211]. Despite the global connections made possible through the web, the Global Risks Report thus finds that "societies are becoming more disconnected. Populations find themselves increasingly polarized and bombarded with misinformation" that "can erode community trust in science, threaten governability and tear the social fabric."²¹ The World Economic Forum's Global Issues describe trust as being "comprised of a range of attributes: credibility, faithfulness, information sharing, and the expectation of cooperation between partners."²² Providing people the means to assess information they find online, for example by key indicators as suggested with the information nutrition label [85], could be one piece for reversing the decreasing trust in institutions, especially for the news media.²³ Using the terms of Gollub et al. [93], which are based on Aristotle's modes of persuasion, these issues are related to assessing online information with respect to its kairos (the momentum of an article or a topic, symbolized as temperature in Figure 1.2 (b)), its pathos (its subjectivity, symbolized as sound pressure), and its ethos (its reliability, symbolized as rating class).

²⁰E.g., 72% of Germans read newspaper online. Statista. Reach of digital newspapers in Germany from October to December 2020. <https://perma.cc/S7T3-DJ7W>

²¹WEF. The Global Risks Report 2021. Pages 32f. <https://perma.cc/CZG2-V9AC>

²²WEF, Strategic Intelligence. Values, Trust as a Value. <https://perma.cc/X6DK-ZPGM>

²³40% of adults worldwide report a trust decrease in the media between 2018 and 2019. Statista. Share of adults who trust the media less than they did a year ago as a result of fake news worldwide as of January 2019, by country. <https://perma.cc/7N36-HDXV>

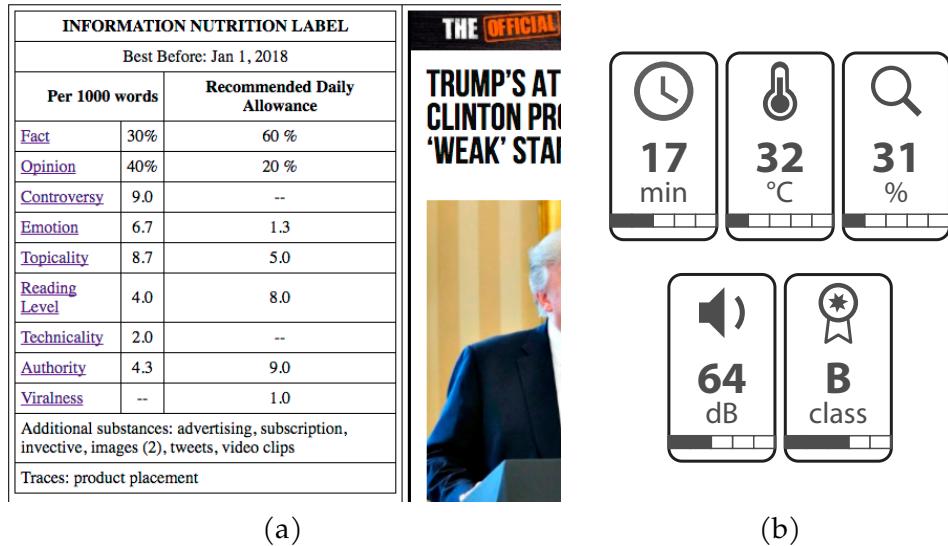


FIGURE 1.2: Visions for an information nutrition label. (a) The original from Fuhr et al. [85]; (b) the representation suggested by Gollub et al. [93].

Web archives preserve the context of information, which is valuable for both provenance and information assessment. Preserving the web page that contains a piece of information allows to track changes over time, for example corrections to a news article. Having a snapshot of a web page available from a trusted third party also provides for accountability and a reliable way of citing the information. Combined with technology that tracks responses of web pages to other web pages, this reliability allows for accurately recording the discussions around pieces of information and to make these discussions available to the visitor.²⁴ Some websites—most prominently Wikipedia—even build on such discussions to enhance their content quality, and archived discussions can provide insight into the controversiality of a piece of information, among others. Moreover, web archives allow to track the flow of information through the World Wide Web [12], which can provide further context to the pieces of information on a web page. For example, showing the tracked history could raise people’s awareness when they are increasingly digesting information from extremist web pages—a common path taken by people that end up adopting extremist views [157, p. 72] and a challenge for an inclusive and reflective society.²⁵

In the context of this challenge, Chapter 4 investigates the following research questions:

²⁴E.g., <http://rbutr.com/>; relies on contributors to archive the web page

²⁵European Commission. Horizon 2020, Work Programme 2018-2020, SU-Governance-11-2018. Page 78. <https://perma.cc/TW3Q-JA4W>

RQ4. How to employ edit histories to assess information? Section 4.1 contributes a novel approach to distinguish different types of benign and malign edits to Wikipedia based on the article history. The approach employs seven patterns of edit reverts for this assessment. Furthermore, the section shows how this assessment of past edits can provide insight into correlations of malign information editing with contextual variables—in this case the time-of-day and location of the editor—that can then be employed as part of on-the-fly assessment technologies. Our analysis based on the assessment reveals that the ratio of vandalism is highest on work days and during work hours, with some differences between countries. Using the symbols of Figure 1.2 (b), the section addresses the rating class.

RQ5. How to identify extremist (i.e., hyperpartisan) content in news archives? Section 4.2 contributes an investigation on the automatic assessment of the “hyperpartisanship” of news articles using archives of online news. The section introduces the task itself, a comparison of topic and style features for tackling the task, and two datasets. The analysis reveals the effectiveness of stylistic text features for distinguishing hyperpartisan (both conservative and liberal) from regular news articles. Moreover, it reports on the successful conduction of the first international competition on the task, in which 42 teams submitted approaches to tackle the challenge. Using the symbols of Figure 1.2 (b), the section addresses the sound pressure and temperature.

1.1.3 Harnessing Web Archives for Online Security and Privacy

The challenge of online security and privacy arises from the dangers towards private persons as they navigate the web. While online (or cyber) security is about mitigating direct risks and harms like identity theft and software manipulation, online privacy is about preventing invisible and unwanted tracking and data collection. Both are part of the grand challenge for the future Internet.²⁶ Indeed, though many solutions are very technical, online security and privacy does not concern computer science alone. The description for the corresponding global issue in the World Economic Forum list points to the cyber threats that arise through the increasing connection “everything and everyone” to the digital world, for example in the concept of a smart city.²⁷ The grand challenge of software reliability raises

²⁶Gesellschaft für Informatik. Internet der Zukunft – sicher, schnell, vertrauenswürdig (The Internet of the Future - secure, fast, trustworthy). <https://perma.cc/DZ52-RSYY>

²⁷WEF, Strategic Intelligence. Cybersecurity. <https://perma.cc/DP46-KJWF>

similar concerns.²⁸ Unsurprisingly, also the Horizon 2020 programme lists cyber-security as one main aspect of the challenge of securing societies.²⁹ At the same time, private persons become increasingly concerned about their online privacy, not just because of companies that trade data for profit but also because of governments.³⁰ The web indeed brings the possibility for large-scale and yet invisible surveillance. Specifically, the JRC science for policy report and technology and democracy points out in its executive summary: “Ensuring online privacy preserves three core components of democratically empowered voters: freedom of association, truth-finding and opportunities to discover new perspectives. Effective privacy online means a strengthened democracy offline.” [157, p. 4]

Web archives provide on the one hand the large-scale public data needed for some empirical security analyses and on the other hand an opportunity for privacy-preserving browsing. System security often depends on a tight definition of normal behavior to identify abnormal and potentially malicious activity. For example, a secure recommender system has to be robust against paid or automated reviews, which requires models to identify “normal” customer reviews. Naturally, web archives provide the largest datasets of human activity for many situations. Importantly, correlations in human activity can be employed to investigate the security of systems even if no data for a specific system is available. For example, a widespread advice for choosing a new password is to derive it from a randomly picked sentence.³¹ But no suitable dataset of real passwords exists to estimate the security of passwords generated accordingly. However, web archives contain billions of sentences written by a large variety of people for a large variety of purposes. Such sentences can thus be used as a proxy to estimate the security for randomly picked sentences. For privacy, on the other hand, web archives provide people with a means to access web content without being tracked. Though web archives can not be employed to this end when the user wants to send information to the server or when the web page loads content after user interaction, web archives allow to retrieve content that has been

²⁸Gesellschaft für Informatik. Verlässlichkeit von Software (software reliability). <https://perma.cc/63RE-M5YZ>

²⁹European Commission. Horizon 2020, Secure societies – Protecting freedom and security of Europe and its citizens. <https://perma.cc/RA2Y-LXPR>

³⁰66% of adult Internet users worldwide report an increasing concern for privacy in 2019. Statista. Share of internet users who are increasingly concerned about their online privacy due to their own government as of February 2019, by country. <https://perma.cc/Q6LW-U4ZJ>

³¹For example, the advice is given by the relevant German office. Bundesamt für Sicherheit in der Informationstechnik (Federal Office for Information Security). Sichere Passwörter erstellen (Creating secure passwords). <https://perma.cc/58YQ-9UAP>

accessed previously by the user or someone else, without anyone but the archive server noticing.³²

In the context of this challenge, Chapter 5 investigates the following research questions:

RQ6. How to estimate the security of passwords that were derived from randomly chosen human mnemonics? Section 5.1 contributes a comparison of password creation policies through the use of a web archive. Passwords are still the dominant form of online authentication, and different advice exists to generate secure passwords. However, security estimates for passwords picked randomly by humans are lacking. Using both the huge amount and large variety of sentences provided through web archives, the section quantifies the security of one particular advice—taking the first letter of each word in a randomly picked sentence—finding that these so-called “mnemonic” passwords provide roughly the security of a 12–13 sided dice per password character—much less than what the alphabet’s set of 26 characters could offer. Extending the character space by special characters yields only a minor increase in security.

RQ7. How to protect privacy when re-finding online information?

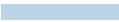
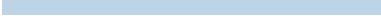
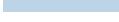
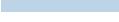
Section 5.2 contributes an investigation into the use of web archive technology for providing people with a searchable offline copy of what they saw online, allowing them to re-access web pages without being tracked. Web services (including search engines) often support re-finding information through tracking the user’s activity. Methods that disable user tracking for privacy concerns thus often also disable re-finding functionality. Private web archives, however, allow to re-find information without connecting to the real Internet and thereby re-enable such functionality while preserving the user’s privacy. Our case study highlights possibilities and current limitations of such a system.

1.2 The Data Analysis Pipeline in the Thesis

In an orthogonal view, all approaches for tackling societal challenges that this thesis presents follow the generic data analysis pipeline of acquisition,

³²However, such private access also reduces the traffic to the original website, with several side effects. Critically, reducing traffic also reduces ad revenue. Some thus suggest to use web archives against fake news publishers, e.g., <https://iffy.news/wayback/>. This thesis does not deal with this issue, as its consideration of privacy is limited to creating personal web archives, which are for this issue similar to taking a screenshot for private use.

TABLE 1.2: The contributions of this thesis' sections as per the web analysis pipeline.

Chapter/Section	Acquisition	Modeling	Analysis
2 The Technology of Web Archives			
2.1 The Web Archive Ecosystem			
2.2 Archiving for Reproducibility: The Webis Web Archiver			
3 Harnessing Web Archives to Preserve Digital Culture (Challenge I)			
3.1 Assessment of Web Page Reproduction Quality			
3.2 Analysis of Unexpected Content in Web Archives			
3.3 Segmentation of the Pages in Web Archives			
4 Harnessing Web Archives to Critically Assess Information (Challenge II)			
4.1 Identification of Vandalism on Wikipedia			
4.2 Identification of Hyperpartisan News			
5 Harnessing Web Archives for Online Security and Privacy (Challenge III)			
5.1 Security Estimate for Mnemonic Passwords			
5.2 WASP for Privacy-Aware Information Re-Finding			

modeling, and analysis. In the context of this thesis, these three steps correspond to (1) the acquisition, preservation, and annotation of web data; (2) the creation of models from the web data, exploiting the temporal information, fidelity, and volume and provided by web archives; and (3) the analysis of the models to gain insights on the societal challenges. Table 1.2 organizes the contributions of each thesis section to these steps. The following describes the steps and corresponding contributions in more detail.

Acquisition The analysis pipeline's first step, the acquisition of web data, is usually the step that executes the actual web archiving process. As introduced above, one can distinguish two approaches to web archiving: (1) the “classical” web archives that record and preserve all communication between the browser and the web for replay and, (2) the preservation in a page-specific format (e.g., a database and source code dump). This thesis focuses on the former—for which it describes the technical details in Chapter 2. Specifically, this thesis investigates on new or improved ways to create web archives for reproducible interactions (Section 2.2) and private use (Section 5.2); to ensure the quality of web archives per visual differences (Section 3.1) and per wrong or erroneous page identification (Section 3.2); and to post-process web archives by segmentation (Section 3.3). Moreover, to foster more research on these topics, several new web archive datasets are acquired and made available online (Sections 3.1, 3.2, 3.3, and 4.2). However, the thesis also employs one page-specific archive, namely the Wikipedia history dump (Section 4.1).

Modeling The data analysis pipeline’s second step, the creation of models, exploits the acquired web archives for creating time-sensitive, larger, and more reliable models. Web archives provide for rich features from various domains and thus for a variety of models at different levels. Specifically, this thesis investigates on new or improved models³³ of language (Section 5.1), web pages (Section 3.3), articles (Section 4.2), and article edits (Section 4.1). To this end, it demonstrates the usage of text (Sections 3.3, 4.2 and 5.1), markup language (Sections 3.3 and 4.2), visual renderings (Section 3.3), and spatio-temporal features (Section 4.1) of web archives.

Analysis The data analysis pipeline’s last step, the analysis, then applies the models for tackling selected societal challenges. Some of the analyses presented here make use of large amount of data available in web archives, whereas other focus on the replicability that web archives provide for. Broadly, the analysis steps of this thesis can be categorized into investigating proofs of concept (Sections 3.1 and 5.2), quantitative results (Sections 3.1, 3.2, 4.1, and 5.1), and automated classification (Sections 3.1, 3.3, and 4.2).

1.3 Thesis Structure

This thesis deals with the preservation of web content and its analyses to tackle different societal challenges. Chapter 2 provides background on web archiving and details the thesis’ methodological contributions to this area. Chapters 3, 4, and 5 then describe the thesis’ contributions to tackling the three societal challenges of preserving digital culture, critically assessing information, and online security and privacy, respectively. These three chapters are kept largely independent of each other to allow for reading in any order. The conclusions in these chapters again are summarized at the end of each chapter. Chapter 6 concludes this thesis with an outlook on future uses of web archiving technology to tackle societal challenges.

Table 1.3 shows the publications that underlie each chapter, as well as a selection of other publications produced during the time of this thesis on the challenge of designing highly interactive and intuitive human-computer interfaces. Though this area is considered also a societal challenge (e.g., it is one of the five grand challenges in the Gesellschaft für Informatik’s list [106]), the publications are not part of this thesis as they do not depend on web archiving.

³³Standard or trivial models that are used in the thesis sections are not listed here.

TABLE 1.3: A selection of peer-reviewed publications by the author and their usage within this dissertation.

Section	Venue	Type	Pages	Year	Publisher	Ref.
2.2, 3.1	JDIQ	journal	23	2018	ACM	[127]
	<i>Johannes Kiesel, Florian Kneist, Milad Alshomary, Benno Stein, Matthias Hagen, and Martin Potthast. Reproducible Web Corpora: Interactive Archiving with Automatic Quality Assessment. (FAIRest Dataset Award of the TKFDM)</i>					
3.2	JCDL	conference	2	2019	ACM/IEEE	[129]
	<i>Johannes Kiesel, Fabienne Hubricht, Benno Stein, and Martin Potthast. A Dataset for Content Error Detection in Web Archives. (Best Poster Award)</i>					
3.3	CIKM	conference	8	2020	ACM	[131]
	<i>Johannes Kiesel, Florian Kneist, Lars Meyer, Kristof Komlossy, Benno Stein, Martin Potthast. Web Page Segmentation Revisited: Evaluation Framework and Dataset.</i>					
3.3	ECIR	conference	14	2021	Springer	[134]
	<i>Johannes Kiesel, Lars Meyer, Florian Kneist, Benno Stein, and Martin Potthast. An Empirical Comparison of Web Page Segmentation Algorithms.</i>					
4.1	ICWSM	conference	10	2017	AAAI	[123]
	<i>Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein. Spatio-temporal Analysis of Reverted Wikipedia Edits.</i>					
4.2	ACL	conference	10	2018	ACL	[196]
	<i>Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A Stylo-metric Inquiry into Hyperpartisan and Fake News.</i>					
4.2	SEMEVAL	workshop	11	2019	ACL	[130]
	<i>Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. SemEval-2019 Task 4: Hyperpartisan News Detection.</i>					
5.1	NDSS	conference	13	2017	Internet Society	[124]
	<i>Johannes Kiesel, Benno Stein, and Stefan Lucks. A Large-scale Analysis of the Mnemonic Password Advice.</i>					
5.2	DESires	conference	6	2018	CEUR-WS	[126]
	<i>Johannes Kiesel, Arjen P. de Vries, Matthias Hagen, Benno Stein, and Martin Potthast. WASP: Web Archiving and Search Personalized.</i>					
-	CHIIR	conference	5	2022	ACM	[137]
	<i>Johannes Kiesel, Volker Bernhard, Marcel Gohsen, Josef Roth, and Benno Stein. What is That? Crowdsourcing Questions to a Virtual Exhibition.</i>					
-	DESires	conference	11	2021	CEUR-WS	[133]
	<i>Johannes Kiesel, Xiaoni Cai, Roxanne El Baff, Benno Stein, and Matthias Hagen. Toward Conversational Query Reformulation.</i>					
-	TOIS	journal	44	2021	ACM	[135]
	<i>Johannes Kiesel, Lars Meyer, Martin Potthast, and Benno Stein. Meta-Information in Conversational Search.</i>					
-	CUI	conference	5	2021	ACM	[136]
	<i>Johannes Kiesel, Damiano Spina, Henning Wachsmuth, and Benno Stein. The Meant, the Said, and the Understood: Conversational Argument Search and Cognitive Biases.</i>					
-	CHIIR	conference	10	2020	ACM	[132]
	<i>Johannes Kiesel, Kevin Lang, Henning Wachsmuth, Eva Hornecker, and Benno Stein. Investigating Expectations for Voice-based and Conversational Argument Search on the Web.</i>					
-	CHIIR	conference	5	2019	ACM	[128]
	<i>Johannes Kiesel, Arefeh Bahrami, Benno Stein, Avishek Anand, and Matthias Hagen. Clarifying False Memories in Voice-based Search.</i>					
-	SIGIR	conference	4	2018	ACM	[125]
	<i>Johannes Kiesel, Arefeh Bahrami, Benno Stein, Avishek Anand, and Matthias Hagen. Toward Voice Query Clarification.</i>					

2

The Technology of Web Archives

The rapid development of web technology makes web archiving far from trivial, and many problems still exist in its field. While in general, as described in Section 1, two approaches exist for the preservation of web archives—the “classical” web archives that preserve all communication between the browser versus the Web for replay and the preservation in a page-specific format—the technology described in this section concerns only the former, as solutions for the latter differ considerably from case to case. For classical web archives, however, an ecosystem of applications has been developed that center around (large amounts of) WARC files (Web ARChive), the standard file format for web archives. Section 2.1 provides an overview of this ecosystem. Section 2.2 describes in detail an own contribution to this ecosystem that this thesis employs throughout: the Webis Web Archiver, the first tool for creating and using WARC files with a focus on enabling scientific-level reproducibility of web interactions.

2.1 The Web Archive Ecosystem

Outside of science, web archiving has been a focus of many initiatives for a long time [24, 94]. Most prominently, the Internet Archive works since 1996 on archiving as much of the web (and other parts of the Internet) as possible, intending to build a comprehensive library of the web. The Internet Archive’s well-known Wayback Machine is a web service allowing to access and browse past versions of all web pages archived that the public frequently uses [10]. Both the Internet Archive and Perma.cc allow its users to create snapshots of web pages on demand, e.g., so that one can prove at

a later time how a given web page looked like at the time of archiving. As of recent, Wikipedia uses these services to archive linked sources¹ to fight link rot [233].

The first kind of tools within the ecosystem concern the archiving of web pages, typically as a standard web archive file (WARC). The tools usually feature the ability to store streaming data [107], to tailor the archiving to certain websites [77], and to simulate basic user interactions [200].² A classic tool for archiving is the Internet Archive’s man-in-the-middle archiving proxy warcprox.³ Moreover, archiving tools can be connected to several other tools that exist in the context of crawling, for example, tools that detect changes in web pages like Pagelyzer.⁴

Several approaches exist to reproduce an archived web page. The Memento protocol allows browsers to request specific versions of a web resource similar to HTTP content negotiation.⁵ The protocol provides a standardized interface for tools that reproduce web pages from an archive. It is implemented by the two most widespread such tools, OpenWayback⁶ and Python WayBack.⁷ Both allow access to archived web pages, either via specific URLs or by acting as a proxy server serving responses from the archive file instead of from the web. Besides the Wayback Machine, also other tools to browse web archives have been developed, some of which target specific user groups. For example, Warcbase [159] is intended for use by historians.

However, as web archives are often created on a large scale, they also provide for large-scale statistical analyses—a core theme in this thesis—for which another kind of tool exists. Common frameworks include ArchiveSpark⁸ and the Archives Unleashed Toolkit [203]. The WARCIO⁹ and Chat-Noir Resiliparse [32] libraries allow for efficient parsing of large amounts of WARC files. For accessing single records in a large archive, on the other hand, the CDX-format¹⁰ is usually used as a standardized format for an index with metadata.

¹Wikimedia, Meta-Wiki. InternetArchiveBot. <https://perma.cc/UMR2-4W74>

²Available open source web archiving services include Brozzler (<https://github.com/internetarchive/brozzler>) and Umbra (<https://github.com/internetarchive/umbra>)

³<https://github.com/internetarchive/warcprox>

⁴<https://github.com/openpreserve/pagelyzer>

⁵See the description by the Memento Development Group, <https://perma.cc/A7UT-7PHR>, and the corresponding Request for Comments (RFC 7089), <https://perma.cc/92QD-3H92>.

⁶<https://netpreserve.org/web-archiving/openwayback>

⁷<https://github.com/ikreymer/pywb>

⁸<https://github.com/helgeho/ArchiveSpark>

⁹<https://github.com/webrecorder/warcio>

¹⁰See the CDX specification by the IIPC, <https://perma.cc/2LKT-5NPK>

2.2 Archiving for Reproducibility: The Webis Web Archiver

This section presents the Webis Web Archiver, a tool to facilitate reproducible web analysis research. Specifically, the following discusses the Webis Web Archiver’s envisioned use cases, a requirements analysis derived from that, its software architecture, and details on user-page simulations and fuzzy request-response matching during reproduction plus a number of other technical challenges. Our web archiver is available as a readily executable Docker image,¹¹ and in source.¹²

2.2.1 Target Use Cases

The Webis Web Archiver has been designed to facilitate reproducible web analysis research. It specifically targets the following four use cases.

Reproducible web corpora A key use case of our archiver is the construction of reproducible web corpora, where each web page archived can be reproduced as close to its original as possible. This has two benefits: human inspection and annotation of web pages is not harmed, rendering conclusions drawn more realistic and hence experimental results based on them more valid. At the same time, algorithms can be developed against a static, yet close-to-realistic source of raw data. In particular, information can be extracted from all of a web page’s resources, including features extracted from a web page’s visual representation, e.g., when relying on recent advances in deep learning for computer vision. Even if the page appearance is not used in the development of some algorithm based on web data, one should not discard the supposed overhead, since new ideas at solving the algorithm’s task may emerge at a time when it is too late to re-crawl the previously omitted data.

Reproducible user experience In laboratory user studies, participants often have to accomplish a task that involves using a web service (e.g., a search engine) and web browsing (e.g., browsing search results). For instance, one may wish to analyze the reaction of different users to a specific situation without explicitly spoiling it to participants. Since using live web services and live web pages may allow for high variation, it has been technically challenging to ensure that participants will independently experience the

¹¹Docker image: <https://hub.docker.com/r/webis/web-archive-environment>

¹²Instructions, binaries, code: <https://github.com/webis-de/webis-web-archiver>

situation in question exactly the same, while not changing the general user experience they are used to (e.g., the web browser). Hence, another use case for our archiver is the creation of a reproducible user experience by allowing to manually or automatically pre-record the usage of web services as well as web browsing all in one web archive, including alternative predicted user behaviors. This way, participants of a user study may not notice a difference when actually running through an experiment, and the experiment itself can be repeated any time, even when the original web services and web pages involved have changed or vanished.

Reproducible user behavior In observational user studies, participants have to accomplish a task on their own, usually with as little guidance as necessary to set them in the right state of mind. It may even be the case that participants are asked to use a web service in any way they please (e.g., an entertainment system). In cases where the degrees of freedom for users are too large to be anticipated in advance, user sessions can be archived using standard tools (optionally including clickstream logging [181]). With our tool, the recorded sessions can be algorithmically revisited for an analysis at any time after the study, even when the web service or web pages underlying the study have meanwhile changed.

Reproducible user simulation The simulation of users has gained significant interest in information retrieval as of recent [22], where users of search engines are simulated to avoid the costs of collecting click data from real users. While recorded user sessions may serve as training data for user simulation, the simulations themselves become increasingly more sophisticated and dynamic. Conceivably, a simulated user may be envisioned which reacts to content it “sees” on a web page, so that the capability to reproduce the look and feel of a web page becomes important. Here, an additional use case for our archiver is to record user simulation runs, rendering them reproducible.

2.2.2 Requirements Analysis

Based on the target use cases listed above, we derive the following seven key requirements for the web archiver.

Scriptable user-page interactions Users of our archiver want to be able to easily specify the user-page interactions that happen during the archiving or reproduction of web pages. This includes taking screenshots, manipulating

and exporting a web page's DOM tree, firing any kind of event that occurs when a user interacts with a web page, such as click, tap, scroll, etc., entering text into and submitting forms, and so on.

Scalable archiving and reproduction Since web corpora nowadays contain millions or even billions of pages, parallel archiving and reproduction are mandatory. The archiving and reproduction of an individual page must be seamless, not blocking human user interaction, and not taking longer than loading the original web page did.

Long-term reproducibility To enable reproducible science on web pages, the reproduction of the web pages must not change over time. For example the rendering of a web page can change with newer browser versions, which has a negative impact on the reproducibility.

Adjustable level of archive granularity Web archivers typically store pages in archive files, each collecting hundreds of pages. This makes handling individual pages difficult, which is a requirement when constructing small, focused web corpora for specific research questions. The level of archive granularity must be adjustable, the lower boundary being exactly one page. This also makes splitting web corpora into sets for training and testing easier.

Local execution and storage One must be able to create and store web archives locally, without reliance on third parties. Possible reasons include that to-be-archived web pages are not accessible from the open web, privacy issues, or that experiments demand to process and manipulate the web pages before the reproduction.

Compatibility with other web archiving software Given the buzzing web archiving ecosystem, the web archives created by our web archiver must be compatible to be processed with third party tools. In this regard, the web archive file format (WARC) is the proper choice to ensure compatibility.

Sustainability The software stack of our web archiver must be maintainable, and easily adjustable by everyone, also when the original authors are unable to provide support. This is in fact a requirement for all of software development in science. Hence, the dependencies of our web archiver must

be chosen to ensure long-term support, proprietary dependencies should be avoided, and slick APIs must be defined to integrate and orchestrate third party components.

The Webis Web Archiver is designed to meet all of these requirements. Below, we overview its software architecture.

2.2.3 Software Architecture

Web archiving software typically comprises two major components, one for archiving a web page, and another for reproducing a web page from a pre-recorded archive. Figure 2.1 overviews the two components, and the respective processes for archiving and reproduction they implement.

The Webis Web Archiver depends on a number of carefully selected software libraries. A readily executable configuration of the archiver is encapsulated in an image for the widespread and industry-standard Docker¹³ virtualization platform. Docker allows to robustly run our archiver in parallel on any host where Docker is installed. It helps to separate the configuration and orchestration of our archiver’s dependencies without affecting, or being affected by other software that may be installed on a given host. Docker also ensures the reproducibility of our archiver’s execution environment by fixing the versions of each software library and especially the browser. Moreover, all of 2 GB worth of fonts available to Ubuntu are installed in the Docker image, which ensures that virtually all characters are properly displayed when taking screenshots, regardless the language. This way, using our archiver boils down to executing a Docker command with a small number of parameters. To avoid caching effects, a new docker container is started for each URL. Also, Unix shell scripts are provided to allow for easy execution of our archiver outside Docker.

The virtual screen software xvfb¹⁴ is used to run the browser without requiring a physical screen, thus allowing for server-side execution. Apart from two exceptions, the archiving process and the reproduction process are exactly the same (cf. Figure 2.1). In both processes, a user simulation script is read from disk and started. The Selenium browser automation software¹⁵ serves as an interface between the script and the browser. As browser, we employ a current version of Google Chromium, but others are supported as well. As the first difference between archiving and

¹³<https://www.docker.com>

¹⁴<https://www.x.org/releases/X11R7.7/doc/man/man1/Xvfb.1.xhtml>

¹⁵<http://www.seleniumhq.org>

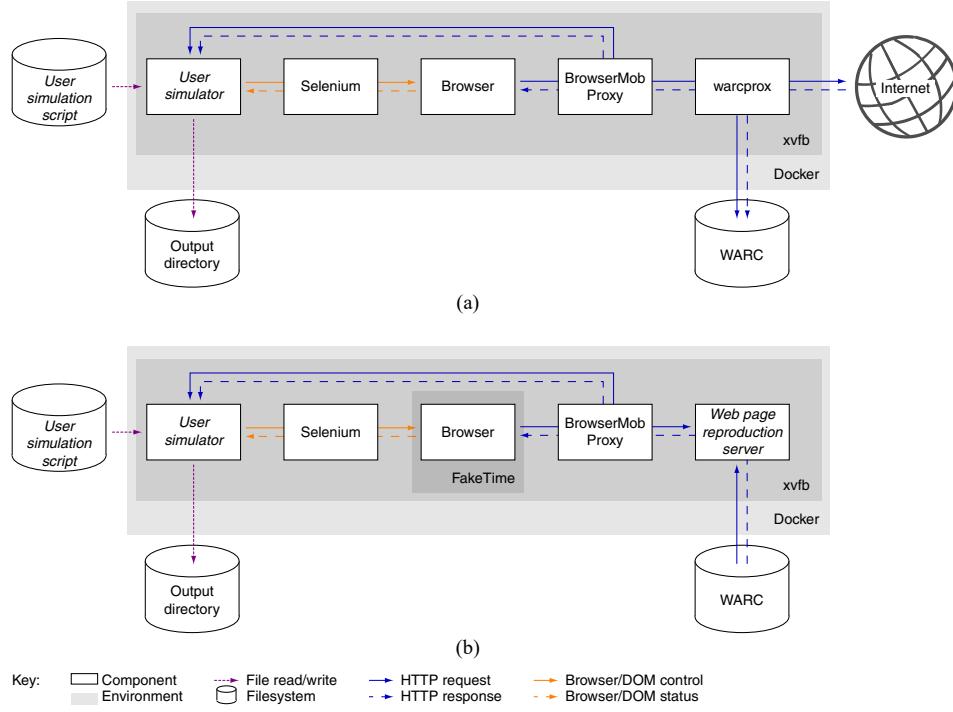


FIGURE 2.1: Software architecture of the Webis Web Archiver. Depicted are the components, environments, and communications between them (a) for archiving and (b) for reproduction. The components originally created for the Webis Web Archiver are highlighted in italics.

reproduction, the *FakeTime* Preload library¹⁶ is used during reproduction to pretend to the browser that it runs at the time of archiving, which affects all JavaScript calls that use the current date. The browser is for both processes set up to communicate with an instance of *BrowserMob Proxy*,¹⁷ which is used by the script to learn when network traffic ceases. The second difference concerns the final component of the archiver. During archiving, the *BrowserMob Proxy* communicates with the Internet via an instance of the *warcprox proxy*,¹⁸ which stores all requests and the corresponding responses that pass through it in a standard WARC archive file. During reproduction, a local server is started that pretends to be a proxy, but actually attempts to retrieve the previously recorded responses to requests made by a to-be-reproduced web page from its WARC files.

Important parameters include the start URL for the user simulation script, the script itself, the output directory, whether to archive or to re-

¹⁶<https://github.com/wolfcw/libfaketime>

¹⁷<https://github.com/lightbody/browsermob-proxy>

¹⁸<https://github.com/internetarchive/warcprox>

Algorithm 1: Built-in user simulation script

Data: Browser *browser*, start URL *url*, output directory *directory*

Result: Screenshot and HTML snapshot of web page at *url* in *directory*

```

begin
    window = browser.openWindowWithUrl(url)
    // Scroll down to load all content
    for i ∈ {1, ..., 25} do
        if !window.canScrollDown() then break
        window.scrollDown(window.viewport.height)
        browser.WaitForNetworkTrafficToStop()

    // Resize browser viewport to page size for
    // screenshot
    window.scrollTo(window.page.top)
    window.resizeViewportHeight(window.page.height)
    browser.WaitForNetworkTrafficToStop()
    // Save current state to output directory
    window.saveHTMLSnapshotTo (directory)
    window.saveScreenshotTo (directory)

```

produce, and where WARC files are to be stored or located. The following sections detail two key features of our archiver, namely its unique user simulation scripts and fuzzy request-response matching.

2.2.4 User Simulation Scripts

A key feature of the Webis Web Archiver is its capability of controlling the browser during archiving and reproduction using the Java-based API. Among other things, the scripts API allows to load any URL into the browser, to scroll, to resize the browser window, to manipulate the DOM, to click on elements, to print elements, to take screenshots, to wait for network traffic to stop, and to execute JavaScript code. Instead of implementing this kind of browser automation ourselves, we rely on Selenium, a software-testing framework for web applications. Selenium has been under development since 2004, and has since risen to become the single most important testing framework of its kind, integrating all major web browsers. Despite its obvious usefulness for web archiving tools, Selenium had never been considered for this purpose until the first release of the Webis Web Archiver. Rather, most of the existing solutions employed PhantomJS,¹⁹ a comparably

¹⁹<https://github.com/ariya/phantomjs>

limited scriptable browser. The interface between Selenium and the web browser has after the first publication of the Webis Web Archiver become the W3C WebDriver standard,²⁰ further increasing the cross-browser compatibility of Selenium and, consequently, our archiver.

We call the code used during archiving *user simulation scripts*, since they can be thought of as modeling a real user's behavior. Nevertheless, more can be accomplished within the use cases outlined above; for example, a user's behavior may be influenced in real-time by changing a web page while the user uses it in a recorded session, a walk on the link graph can be implemented, the web page may be classified, or information extracted from it. Our archiver ships with the built-in user simulation script shown in Algorithm 1: it simulates a user who wants to reach the bottom of a page, but gives up after a predefined maximum number of page down scrolls. During the process, a screenshot of the original page as well as its HTML source are stored. Any user simulation script used during archiving can be used without change during reproduction so as to allow a web page served out of an archive to reach the same state as its original did.

Parameters of a user simulation script are a browser object, a start URL, and an output directory where any extraneous script output can be stored (e.g., a screenshot). The browser object allows opening new browser windows. By default, browser windows have a size of 1366x768, which is the most widespread screen resolution for desktops as of 2018.²¹ Selenium also allows to emulate mobile devices, but this functionality had not been mature enough at the time of publication. Additionally, user simulation scripts have access to the directory in which their source code resides for accessing resources like dictionaries or machine learning models if needed. Our user simulation scripts are currently limited to Java, but Selenium offers more language bindings that could be utilized in the future.

2.2.5 Fuzzy Request-Response Matching

Although archiving and reproducing web pages the way described above may seem straightforward, the devil is in the details, and reproducing web pages from an archive is prone to errors. It is important to understand that a web page is not just a collection of static plain text files that only need to be systematically found and copied. Rather, it is a distributed piece of software with a client-server architecture. What is displayed in a browser is

²⁰W3C. Web Driver, W3C Working Draft 24 August 2020. <https://perma.cc/J2AK-N57W>

²¹W3Schools. Browser Display Statistics (2021). <https://perma.cc/5ES8-JXD4>.

StatCounter. Screen Resolution Stats Worldwide (2018). <https://perma.cc/GF5X-Z33S>

the user interface of such a software, which is assembled and executed just in time as the user visits a page. In this regard, web archiving is essentially the same as taking snapshots of the states of the user interface, hoping that the user simulation script visits all, or at least the important states of a page a real user can reach. As this view on web pages shows, archiving and reproducing a web page is a non-trivial task.

During our developments, we identified and tackled many technical challenges related to technologies used at client side, at server side, and in-between. For brevity, this section describes only one of the more interesting solutions: fuzzy request-response matching. When reproducing a web page, all requests sent by the archived web page must be answered with the same responses that have been captured during archiving. However, the requests sent during reproduction may differ in subtle ways from the ones originally sent, requiring a fuzzy match between request and response. For example, requests that involve random numbers or that are based on the current time will be dissimilar from their original. Furthermore, requests can change the server’s state, so that other requests result in different responses depending on whether they are issued before or afterwards. As the server is essentially a black box to the archiving process, no information exists on which requests have effects on responses, or which responses are affected. Furthermore, as requests are usually sent in parallel by the browser, race conditions will occur. The order in which requests are issued can drastically vary between two runs of the same user simulation script as this order often depends on which responses are received first. For illustration, Figure 2.2 shows an example where the screenshots taken by our archiver differ as a result of failed matches between requests and responses.

Our archiver currently allows to choose from three different tools for reproducing web pages (the “web page reproduction server” in Figure 2.1): warcprox, Python WayBack (pywb), and an implementation of our own. Warcprox²² focuses mainly on archiving and implements only a very basic request-response matcher based on URL normalization.²³ It serves as our baseline. Python WayBack²⁴ is one of the two most widely known web page reproduction tools available.²⁵ It implements a rule-based matcher, engineered specifically for this task, which aims at resolving differences in

²²<https://github.com/internetarchive/warcprox>

²³Reproduction in warcprox was buggy when we first tried it for this purpose, but we helped resolve this issue.

²⁴<https://github.com/ikreymer/pywb>

²⁵Since the most prolific contributors to Python WayBack and OpenWayback are the same people, but the development of Python WayBack is more active, we do not expect Open Wayback to achieve a higher reproduction quality.

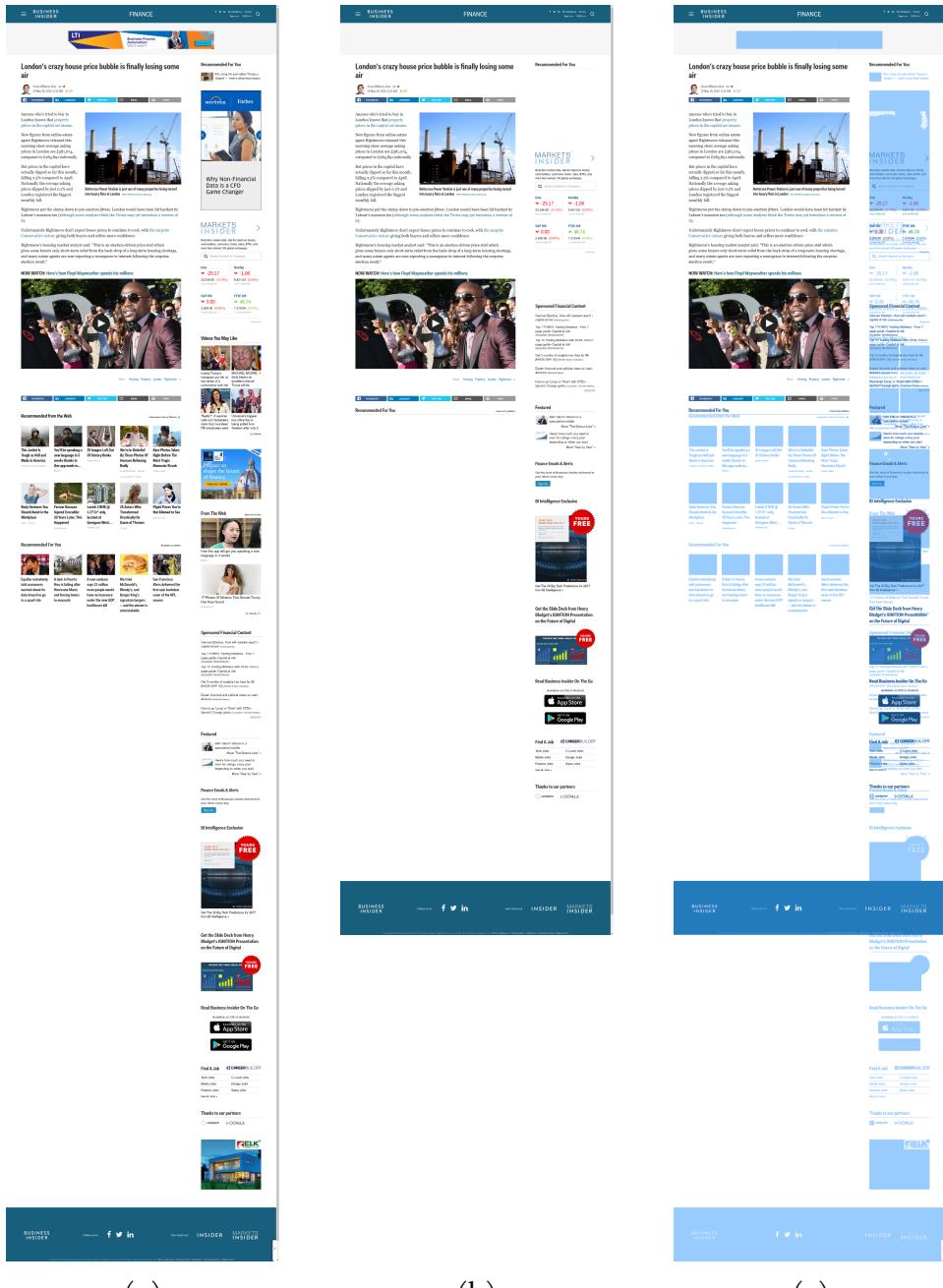


FIGURE 2.2: Screenshots of a web page archived under the user simulation script of Algorithm 1 (a) during archiving (i.e., the original page) and (b) during reproduction, as well as (c) highlighting differences between (a) and (b). The web page has been selected to show typical errors, namely visually shifted content due to missing advertisements and missing recommendations for further reading. Whether these errors degrade the reproduction is a question of defining web page reproduction quality (cf. Section 3.1).

requests that likely occur in a reproduction scenario, for example, by ignoring strings that resemble session IDs. In addition, Python WayBack offers support for organizing multiple archives and handling repeated archiving of the same page, which is not needed in the research scenarios we consider. However, by default, Python WayBack injects code into reproduced web pages, for example, to change the time reported by JavaScript to the time of archiving. We deemed code injection problematic for our purposes—since injected code may interfere with user simulation scripts—as well as for maintaining integrity of the archived data, so we disabled this feature. In preliminary tests, we noticed that some images are missing in the reproduction of Python WayBack without obvious reason. For an easier analysis, we thus implemented a custom request-response matching cascade that tries to find a request match by identity checking including URL normalization, then by partial matching ignoring headers, and if that also fails, by rules very similar to those used in Python WayBack. A comparison of these three reproduction tools has been part of our research on web archive reproduction quality assessment and is thus presented in Section 3.1.

Conclusion

Even though the accurate archiving and reproduction of web pages has made impressive progress, the state of the art for scientific corpus construction is still simply downloading the HTML source code of a page. Archiving and reproduction both face several challenges that need to be addressed by future software engineering and research. For example, during archiving, some web content is tailored toward an assumed location of the client, so this location should be another parameter for an archiving tool. Some websites block clients that perform unusual request patterns (also across websites) in an attempt to avoid bots, which can be problematic for large-scale archiving. However, using proxy servers to address these two problems is a double-edged sword, as some websites block known proxy servers.

Current problems for reproduction that can be tackled by software engineering are data streaming based on HTTP range requests (necessary for videos), where ranges during archiving and reproduction may differ; and reproduction of server-sent events (known as HTTP push), where the web server sends additional information without a request from the client.

A further improvement to the Webis Web Archiver would be the capability to capture the screen during archiving and to store the screen capture as a video. Such videos would add a new preservation element that enriches

the archive files. Furthermore, such videos would assist users in writing and debugging user simulation scripts.

2.3 Summary

This chapter provided an overview of the web archive ecosystem (Section 2.1) and detailed an own contribution to this ecosystem, the Webis Web Archiver (Section 2.2). The latter builds upon existing technologies in the ecosystem but extends them with customizable browser automation (“user simulation”) and containerization technology to provide a reproducible setup for web analysis experiments. Each of the following chapters uses the Webis Web Archiver (or parts of it) to create such a reproducible setup or for other purposes.

3

Harnessing Web Archives to Preserve Digital Culture

Web archives allow the preservation of digital culture as found on the World Wide Web, but, as discussed in Section 1.1.1, the ease of accessing web pages hides the difficulty of preserving them. Web pages are compiled from several sources and may request different content when rendered again later—content that is often not automatically preserved along. Critically, tackling this problem can not be postponed until the missing content becomes evident to some users, as the content might no longer be available then.¹ However, the rate of web page preservation is too high for human quality control. For example, as of July 2021, the Internet Archive’s Wayback Machine claims to give access to more than 590 billion web pages that it preserved or, as a synonym in this context, archived.²

Unfortunately, though, the automatic quality control of archived web pages is far from trivial. Amongst others, it is difficult to operationalize the “quality” of a web archive. Specifically, *how to measure web archive reproduction quality?* Section 3.1 contributes the operationalization that the quality of a web page’s reproduction from a web archive is the higher, the closer the reproduced web page is to the live web page (at the time of archiving) from the user’s perspective. Moreover, the section contributes a dataset and the first approaches for an automatic assessment. Second, it is difficult to identify at scale whether the web server actually provided the content

¹Though ideas exist to improve the availability of content through new addressing and caching technologies, e.g., the InterPlanetary File System <https://ipfs.io>, such technologies are far from being widely deployed, nor would they solve the problem completely

²<https://archive.org/web/>

that should be preserved and not something else (most prominently, an error page). Specifically, *how frequent are different kinds of unexpected content in web pages?* Section 3.2 contributes a first analysis of unexpected content in general-purpose web archives (ads-only pages, loading indicators, page-sized pop-ups, access-blocking CAPTCHAs, and dominant error messages). Its statistical analysis based on human annotations of 10,000 web pages indicates that these signs are pretty common, suggesting unexpected content in roughly 10% of web pages. Moreover, third, reproduction quality is an attribute of web page segments (e.g., navigation, main content, or advertisements). Some types of segments may require different preservation methods, may not be necessary at all to be preserved, or should be treated as separate “documents” for retrieval from the archives (e.g., different articles on one page). A related question is thus *how to define and identify segments of a web page?* Section 3.3 contributes an in-depth investigation of the task of web page segmentation, which has been neglected in the research community for more than a decade despite the rapid developments in web technologies and web design demanding continuous updates. The section introduces a new conceptualization of web segments from the user’s perspective and thus aligns with our operationalization of reproduction quality. In a benchmark of existing algorithms on our new dataset, the section shows that the nearly 20-year-old VIPS still performs best, though there is much opportunity to improve on it.

3.1 Assessment of Web Page Reproduction Quality

The preservation of digital culture is necessary both to provide historical artifacts³ to individuals and to allow for large-scale studies of the society’s past. This section shows that both use cases have similar requirements when it comes to the reproduction quality of web pages. Though this section starts with a focus on the latter, seemingly more demanding use case, it develops an operationalization of web archive quality that corresponds to the former use case of individuals viewing web pages.

The fact that much of the information can be obtained from the web at virtually no expense just by downloading it, renders the web an invaluable source of raw data for various disciplines of science. In particular, many forms of “applied” computer science benefit and hence rely heavily on web data, such as data mining and machine learning, web science and social network analysis, computer linguistics and natural language processing, com-

³In this case, the artifacts are web pages

putational social science and the digital humanities, data science and data journalism, and not least of all, information retrieval.

Nowadays, many web pages are a piece of software, composed of resources from several web servers, compiled and executed just in time in the browser when the user loads it, and dynamically modified and updated with additional content based on user interaction. This is becoming more and more the standard rather than the exception, as evidenced by the usage of JavaScript libraries in more than 70% of all websites,⁴ and the fact that Google has started rendering web pages during crawling.⁵ Reproducing web pages, as in showing an archived page in a browser that looks and behaves like the original page at the time of archiving, has hence become increasingly difficult.

However, simply downloading a web page's HTML source is not sufficient to ensure the reproducibility of a web page's "look and feel." Still, current web corpora like the ClueWeb09 and the ClueWeb12⁶ used within TREC employ this strategy. The styles, scripts, and multimedia files not served inline with the HTML code of a web page are almost entirely missing, and most of them have long since disappeared from the open web. Viewing a web page without these resources results in a completely different experience than its authors intended, which has a detrimental effect on human perception when reviewing and judging pages from these corpora [99], jeopardizing the validity of TREC's evaluation results. Yet, even downloading externalized resources may not be sufficient.

This section introduces the new task of *immediate automatic reproduction quality assessment* for web page archiving, where, given a recently archived web page, the task is to assess its reproduction quality compared to its original under a pre-defined user-page interaction. For this task, we evaluate three quality assessment approaches—the third of which an original contribution—which resort to (1) estimating the number of missing resources and their impact on the web page's utility, (2) the pixel-wise visual difference between archived web page and original, and (3) deep learning on the visual differences. With this task, we disentangle the development of archiving technology from its evaluation. The new dataset we present serves as a reference for future approaches to quality assessment, even after the original web pages have disappeared.

Altogether, this section presents the following three contributions:

⁴https://w3techs.com/technologies/overview/javascript_library/all

⁵<https://webmasters.googleblog.com/2015/10/deprecating-our-ajax-crawling-scheme.html>

⁶<http://lemurproject.org/clueweb09/> and <http://lemurproject.org/clueweb12/>

1. The Webis-Web-Archive-17, a publicly available dataset of 10,000 archived web pages, annotated for reproduction quality via crowdsourcing, and constituting the first benchmark dataset for
2. the new task of immediate automatic reproduction quality assessment, which helps to overcome the inherent irreproducibility of web archiving tool evaluations. And
3. the first in-depth analysis of web page reproduction quality, comparing reproduction software and human quality annotations with three approaches to measure reproduction quality that are based on missing resources, screenshot differences, and learned screenshot comparisons, where the latter constitutes a novel application of deep learning.

After Section 3.1.1 provides background on web page reproduction quality, Section 3.1.2 introduces our operationalization and Section 3.1.3 our dataset build thereon. Section 3.1.4 then introduces the approaches to automatic web page reproduction quality assessment that Section 3.1.5 evaluates.

3.1.1 Background

Web corpora are frequently used for scientific evaluations. However, the currently available ones have been criticized for not allowing the reproduction of web pages collected, undermining human annotation and user studies [99]. These issues have not been resolved until today. The two most widely used corpora for evaluating information retrieval systems, the ClueWeb09 and the ClueWeb12,⁷ contain only HTML files (excluding style sheets, scripts, and images), many of which are truncated to save space. For the TREC Web Tracks alone, more than 125,000 ClueWeb pages have been manually annotated, whereas most of them will not have been displayed as originally intended. Moreover, any algorithm (e.g., for main content extraction) relying on analyzing the appearance of a page and employed as part of a retrieval pipeline by a Web Track participant may as well not have worked as intended. Though it has been tried to fix existing web corpora by identifying, downloading, and extracting information from missing resources [210], this post hoc improvement suffers from the fact that most of the original web pages have since disappeared. The Common Crawl⁸ is also missing many of such resources. Other small web corpora only contain plain text without reference to the original web pages [29] or HTML files without associated resources [144].

⁷<http://lemurproject.org/clueweb09/> and <http://lemurproject.org/clueweb12/>

⁸<https://commoncrawl.org/>

Only few studies have dealt with assessing archiving quality so far. Even though the Web Curator tool⁹ exists for manual assessment, it has not been used to construct web corpora. Regarding automatic assessment, the CLEAR method tries to predict how well an entire website could be archived based on its accessibility, compliance with standards, cohesion, and use of metadata [26, 27]. Brunelle et al. [40] score the archiving quality after archiving, considering the estimated importance of missing images, multi-media objects, and style sheets. They find that quality estimation can be improved by differentiating the importance of missing resources (e.g., based on image size) instead of treating each missing resource the same. We reimplement this approach for our evaluation to compare it with our own approaches.

3.1.2 Defining Reproduction Quality

Intuitively, if a reproduced version of a web page looks and reacts exactly as the original one did at the time of archiving, the reproduction quality is maximal. But quantifying how much an archived web page differs from this ideal is far from trivial. Operationalizing the notions of “looking” and “reacting” the same or similarly is where the difficulties arise, especially when some differences in look or reaction might be negligible (e.g., a missing advertisement) while others render a reproduction unusable (e.g., a missing image in a web comic). Furthermore, the importance of any given difference depends on the context of a web page. For example, missing an image advertisement for a pair of shoes on a discussion board only marginally degrades its reproduction quality, whereas missing the same image on a shoe shopping site degrades its quality more significantly. For another difficulty, as the number of possible interactions with a web page can be enormous, it is infeasible to include all possible interactions in a quality assessment measure, let alone weighing the relative importance of each interaction, which would require a prediction of the likelihood of each interaction.

This thesis therefore adopts a pragmatic definition of reproduction quality based on a basic scroll-down user model (as in Algorithm 1, page 22) and on visual differences between an original page and its reproduction. We chose this user model as scrolling down results in a single image for a web page, whereas other interactions (such as clicking links or buttons) results in a more complex, multi-state representation. Extensions to this user model are straightforward to implement with the Webis Web Archiver (Section 2.2), but require a separate investigation into typical or “worst-case”

⁹<https://webcuratortool.org/>

user behavior. Under the scroll-down user model, we define the quality of a web page reproduction as follows:

The more individual users that scroll down a web page are affected in their perception or use of the web page by visual differences between the original web page and its reproduction, the smaller the reproduction quality for that web page.

3.1.3 The Webis-Web-Archive-17

The section details the creation of the Webis-Web-Archive-17 dataset, which contains a careful sample of 10,000 web pages that have been annotated according to the above definition of reproduction quality.

Sampling of Web Pages

To build a solid benchmark dataset for web reproduction quality assessment, we carefully sampled web pages with the goal of representing a wide cross-section of the different types and genres of web pages found on the web. As a population of web pages to draw a sample from, we resort to the billion-page Common Crawl 2017-04.¹⁰ From there, we primarily sampled pages from most of the well-known sites—as defined by the website’s Alexa traffic rank¹¹—to ensure that our sample encompasses pages using the most recent web technologies and design standards. Moreover, pages from a number of less well-known sites have been included. Altogether, the Webis-Web-Archive-17 comprises 10,000 web pages.

We drew a stratified sample of web pages from websites listed in the Alexa ranking according to the following restrictions: To avoid overrepresentation of organizations that host similar pages under several domains, we disregard pages from sites that are different-language versions or sub-domains of higher-ranked sites.¹² Furthermore, the sampling is restricted to yield exactly 50 pages from each of the top 50 sites, 10 pages from each of the next 100 sites in the ranking, 5 pages from each of the next 500 sites, and finally, 1 page from each of the next 1000 sites. In total, we thus sample 7000 web pages from the top 1650 sites. If the Common Crawl did not contain enough pages for a site, this site was skipped altogether and the next site in the ranking was used instead.¹³ Not all web pages from the Common

¹⁰<http://commoncrawl.org/2017/02/january-2017-crawl-archive-now-available>

¹¹As of March 29th, 2017, <https://www.alexa.com/topsites>

¹²For example, files.wordpress.com is subordinate to wordpress.com

¹³By manual analysis, we found that top-ranked sites with an insufficient amount of pages have either very restrictive crawling conditions (e.g., tmall.com), are link services (e.g., t.co), or are related to advertisement (e.g., onclkds.com)

Crawl still exist, so we continued to archive pages of a site until the desired amount of web pages could be archived successfully. However, as the sample for a website would be hardly representative of the site when most web pages fail archiving, we again skip the website if twice the amount of required web pages have been tried without yielding the desired amount of successes. In total, 429 websites have been skipped. To also include pages from a sample of less-known sites in the dataset, we include an additional random sample of 3000 pages from the remainder of sites.

The final sample of pages was re-crawled and archived using the Webis Web Archiver presented in Section 2.2, employing the user simulation script given in Algorithm 1 (page 22). On average, it took 86 seconds to archive a web page using this script. Reproduction was a bit faster, taking 67 seconds on average with only minor differences between the different reproduction approaches described in Section 2.2.5. For each page, the dataset contains on average 10.4 MB of data: 4.3 MB for the archive, 5.4 MB for screenshots (one screenshot of the live page as seen during archiving, and one when using each reproduction approach), and 0.7 MB for HTML snapshots (distributed like the screenshots).

For each page’s site in the dataset, we queried the Alexa Web Information Service API to retrieve its category and dominant language. Table 3.1 shows the distribution of categories of the websites and their corresponding web pages. All categories are present in the dataset and no single genre is dominating. Indeed, the categories with the most sites are “World” (which can be considered a “miscellaneous”-category), “Regional,” and “Computers,” all of which are rather generic categories enclosing several sub-topics. As Figure 3.1 illustrates, English is the dominant language in the dataset: about 70% of the pages for which the Alexa Web Information Service API returned site-wise language information are English, followed by Chinese (13%) and Russian (4%). While the dominance of English pages is unsurprising, the dataset still covers a reasonable variety of languages.

Human Annotation of Reproduction Quality

Employing the above definition of reproduction quality, we recruited human annotators to manually assess the 6,348 of the 10,000 pages in the Webis-Web-Archive-17 with reproduction errors, which were identified by comparing the screenshot taken during archiving with the ones taken during reproduction. For every web page where none of the three reproduction screenshot (captured using the three reproduction approaches described in Section 2.2.5) matches the archiving screenshot perfectly, we asked hu-

TABLE 3.1: Number of sites and pages in each category and their most frequent sub-categories according to the Alexa Web Information Service. Sites and pages can be in multiple (sub-)categories.

Category	Sites	Pages	Most frequent site sub-categories	Most frequent page sub-categories
Adult	40	129	Computers (25), Image Galleries (8), Society (2)	Computers (94), Image Galleries (27), Society (3)
Arts	187	471	Television (59), Movies (19), Music (18)	Television (152), Movies (75), People (50)
Business	229	489	Financial Services (39), News and Media (24), Arts and Entertainment (22)	Financial Services (116), News and Media (84), Arts and Entertainment (43)
Computers	444	1502	Internet (184), Software (143), Companies (60)	Internet (821), Software (371), Companies (261)
Games	58	150	Video Games (48), Gambling (6), Board Games (1)	Video Games (139), Gambling (6), Puzzles (2)
Health	28	36	Medicine (15), Conditions and Diseases (7), Nursing (5)	Medicine (21), Nursing (11), Conditions and Diseases (7)
Home	42	88	Consumer Information (19), Cooking (9), Personal Finance (7)	Consumer Information (44), Cooking (17), Personal Finance (15)
Kids and Teens	55	99	School Time (21), Games (13), Computers (7)	School Time (43), Games (28), Computers (10)
News	149	303	Newspapers (77), Breaking News (13), Magazines and E-zines (8)	Newspapers (110), Headline Links (55), Weather (24)
Recreation	70	106	Travel (32), Autos (8), Humor (5)	Travel (63), Autos (10), Pets (7)
Reference	164	275	Education (105), Dictionaries (25), Libraries (21)	Education (128), Ask an Expert (51), Dictionaries (47)
Regional	605	1182	North America (395), Europe (112), Asia (78)	North America (829), Europe (240), Asia (144)
Science	74	116	Technology (19), Social Sciences (16), Biology (12)	Technology (34), Social Sciences (27), Biology (20)
Shopping	182	413	General Merchandise (30), Clothing (26), Entertainment (16)	Entertainment (108), General Merchandise (105), Auctions (51)
Society	100	125	Issues (20), Religion and Spirituality (15), Government (12)	Issues (27), Religion and Spirituality (18), Government (16)
Sports	73	123	Resources (18), Soccer (9), Baseball (6)	Resources (48), Soccer (16), Cricket (10)
World	1453	3437	Chinese Simplified CN (305), Russian (245), Deutsch (213)	Chinese Simplified CN (866), Russian (731), Deutsch (593)

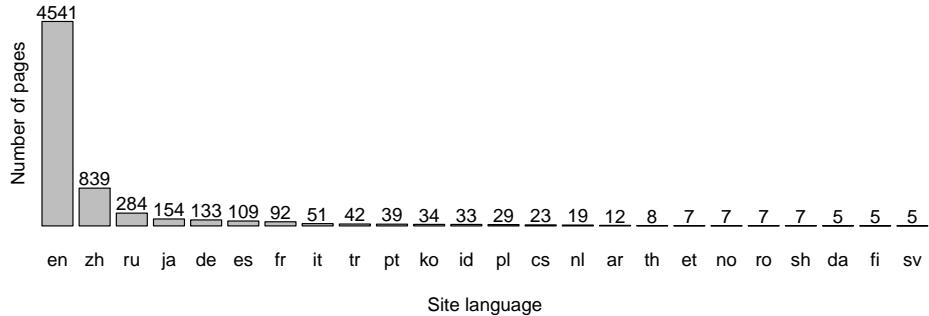


FIGURE 3.1: Number of web pages by their website’s language as determined by the Alexa Web Information Service (successful for 6,495 out of 10,000 pages). Languages with at least 5 pages are shown.

mans to annotate the reproduction quality. Only the reproduction screenshot with the smallest pixel-based difference¹⁴ to the archiving screenshot has been annotated. To match the scroll-down user model, annotators were shown the screenshots of the web page with the possibility to scroll down to the bottom of the page. To enable annotators to assess the effect of visual differences on their perception, the archiving and reproduction screenshots were shown side by side, while synchronizing their scrolling behavior and thus allowing for visual content matching. As some differences were rather small and might otherwise have been overlooked during annotation, the annotators were able to highlight the differences between the screenshots. The reproduction quality was assessed on a 5-point Likert scale to account for different levels of perceived severity, ranked from no effect (score 1) to unusable reproduction (score 5). The interface provided annotators with a short textual description and one example for each of the five quality scores. Figure 3.2 shows the annotation interface with activated highlighting.

To annotate each web page, we hired 9 annotators at Amazon’s Mechanical Turk. Amazon’s Mechanical Turk is a crowdsourcing market where requesters, like ourselves, advertise so-called “human intelligence tasks” (HITs) to workers for a per-task payment upon successful completion. The annotators received their tasks in batches of five web pages; each annotator could work on several batches, but not on the same one twice. To avoid order biases, we randomized the order of pages within a batch as well as whether scores (and examples) are in ascending or descending order. However, to not confuse annotators who annotate several batches, we used the annotator’s Mechanical Turk ID to seed the random number generator, es-

¹⁴As measured by ImageMagick’s RMSE, see Section 3.1.4.

Instructions

- Score the differences of 5 pairs of web pages (requires JavaScript).
- Read the **examples** carefully and make sure you understand them.
- Before you answer the question, have a look at all parts of the pages by **scrolling down** (if possible).
- You can check "Highlight differences" to **highlight differences** on the right page in blue.

Task 3

Left page

Right page

Highlight differences (ignore the blue highlight for answering the question)

Imagine someone wants to visit the left page, but gets the right page. How much would this difference affect the visitor?

Examples for each score:

- Score 1 (not affecting): Parts of the page are just moved up or down a bit.
- Score 2 (small effect on a few visitors): Social media buttons, ads, or unimportant images or text are missing.
- Score 3 (small effect on many or all visitors): Comments on the main content are missing.
- Score 4 (affects, but page can still be used): Striking difference in colors, background, or layout.
- Score 5 (unusable page): Important/main content is missing and/or visitors can't use the right page due to the differences.

Difference will not affect visitor 1 2 3 4 5 Difference makes page useless for visitor

Comments for this task (optional):

[1](#) [2](#) [3](#) [4](#) [5](#) [Next task](#)

[Complete all tasks](#)

FIGURE 3.2: Interface used by the human annotators to judge the reproduction quality of five web pages in a row, showing one at a time.

sentially causing the order to be fixed across batches for each worker. We manually reviewed all web pages where not all 9 annotators agreed on a score within one point of the median score for that web page. Annotation batches for which a score clearly does not fit the screenshot differences as per the provided example for the score were rejected. Overall, we rejected 1234 of 11,430 batches (10.8%). In order to assess the annotation, we calculated the majority agreement among the annotators. In 81% of the cases, more than half of the accepted annotators agreed on a score. Due to this high annotator agreement, we deem the annotations reliable. To derive a ground truth annotation from the set of individual annotations for each page, we

TABLE 3.2: Number of times each median reproduction quality score occurs in the final annotation, ranging from no effect on visitor (1) to useless as a reproduction (5), when considering only reproductions annotated by humans (Annotated), and including the ones with no difference in the screenshots (All).

Web pages	Score					Σ
	1	2	3	4	5	
Annotated	1942 (31%)	3307 (52%)	422 (7%)	318 (5%)	359 (6%)	6348
All	5594 (56%)	3307 (33%)	422 (4%)	318 (3%)	359 (4%)	10000

use the median of the scores that were assigned to a reproduction, which is a stable measure for such a small number of annotations as well as robust against outliers.

As high-quality annotations are very important for benchmark datasets, we then manually curated the web pages where the Mechanical Turk annotators disagreed (regarding only approved batches). Specifically, we took another look at all web pages where annotators gave at least one 1 and one 5, or where at least one annotator gave a score with an absolute difference of 3 from the median score of all annotators (e.g., score 4 for a median of 1). We found these web pages to be difficult cases that were not fully covered by the examples we provided. For example, these pages included videos or assets that were seemingly still loading in the reproduction screenshot (which we count as not reproduced), missing promotions for related products on shopping pages (which we count as content, not advertisement), or missing overlays that ask to accept cookies (which we count as a small effect as they can usually be dismissed easily). Additionally, we looked at all web pages from domains with many similar pages, and ensured that the quality is judged consistently across these pages. In total, this manual curation changed the ground-truth score for 717 of the 6348 reproductions (11%). For an overview, Table 3.2 gives the distribution of final scores, while Figure 3.3 shows one typical reproduction example for each score. As the table shows, the vast majority of all web pages (89%) were reproduced with a score of 1 or 2, demonstrating a sufficiently high reproduction quality for many applications. For example, in main content extraction, a quality score of 2 (small effect on few users) is likely sufficient. Nevertheless, reproduction is still far from perfect. To facilitate faster improvements to reproduction technology at development time, in what follows, we analyze methods for an automatic assessment of reproduction quality in real time, thus allowing for the immediate quantification and assessment of software improvements without having to repeat manual assessment.

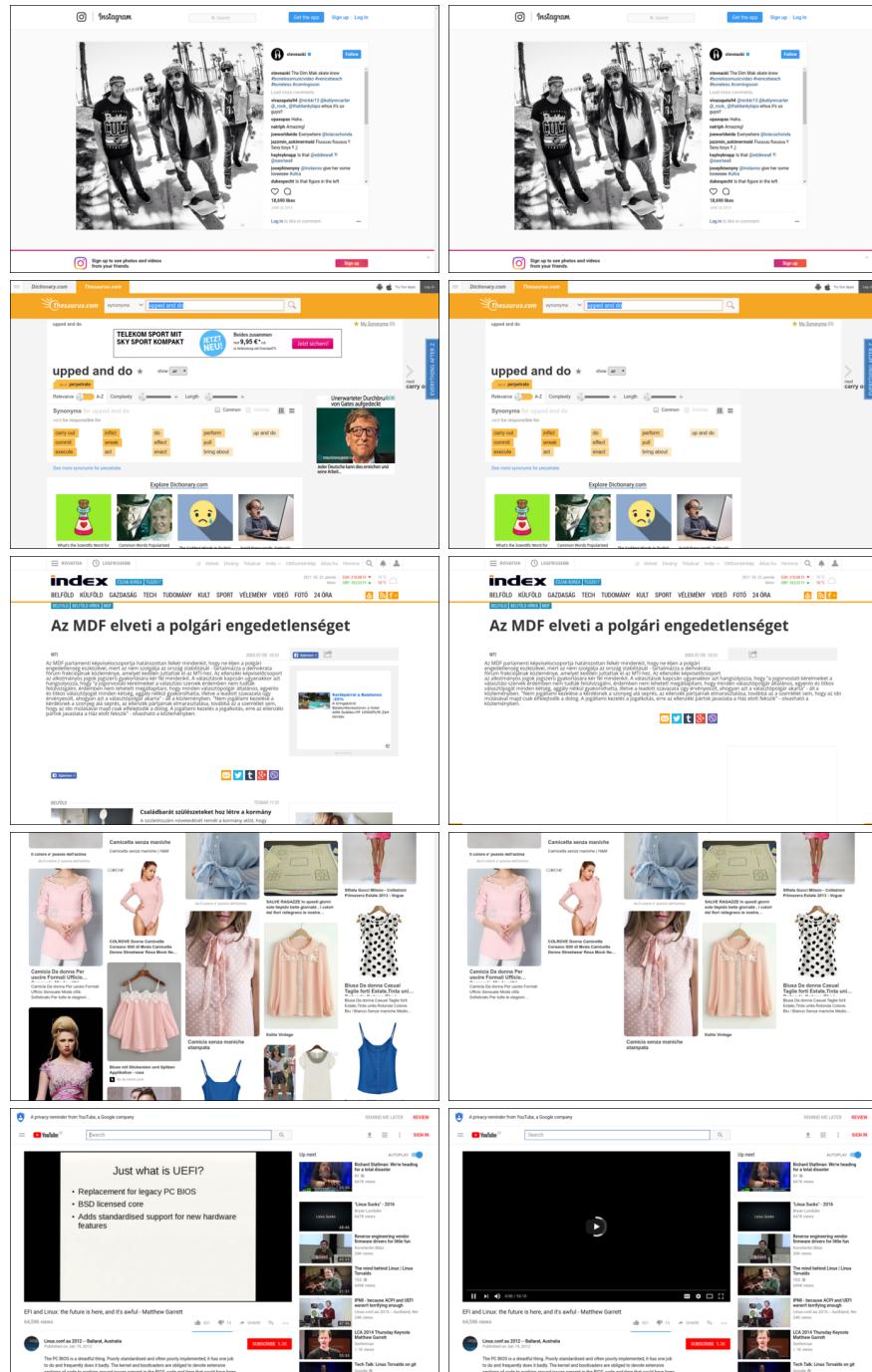


FIGURE 3.3: Typical screenshots (or parts thereof) taken during archiving (left) and reproduction (right) for each of the five available quality scores from 1 (top) to 5 (bottom): (1) an animated button in a different state; (2) missing advertisements; (3) missing links to related pages and missing social media buttons; (4) missing additional content further down the page; and (5) missing main video.

3.1.4 Automatic Reproduction Quality Assessment

The new task of automatic reproduction quality assessment aims at facilitating the fast debugging, quality assurance, and improvement of web archiving and reproduction technologies. The goal of solving this task is to integrate into existing archiving tools a feature that automatically evaluates the reproduction quality after archiving a web page, providing immediate feedback to the tool's user and developer. This section introduces three different approaches suitable for this task, one from related work, one standard measure, and one based on neural networks which is our own contribution.

The development of such assessment measures is key for the development of web archiving tools as, unlike traditional evaluation setups, the evaluation of web archiving tools cannot rely on benchmark datasets or corpora: compiling a dataset of static web pages is insufficient, since an archiving tool requires a functional web server that hosts web pages in order to archive them. Even if a sufficiently large set of web servers hosting a sufficiently large number of different web pages, each built with a diversity of web technologies, were conserved and made available, the rapid pace at which web technology evolves would soon render this resource outdated and evaluation results gained from it irrelevant. Therefore, direct comparative evaluations of web archiving tools that are developed by different parties at different points in time are virtually impossible at reasonable costs. A potential solution to this problem can be found in a standardized, manual evaluation procedure to be followed minutely by independent parties that develop web archiving tools. The operationalization of sensible notions of web archiving and reproduction quality criteria, however, has its own pitfalls, as our selection of measures below shows. Further, the comparability of such evaluations conducted at different points in time is weaker than with a direct comparative evaluation. Nevertheless, at least scale can be attained via crowdsourcing, which is how we were able to conduct a large-scale evaluation the Webis Web Archiver, demonstrating the use of Amazon's Mechanical Turk for this purpose for the first time.

Brunelle15 As a first measure of reproduction quality, we use the heuristic damage rating proposed by Brunelle et al. [40] for measuring the impact of missing resources. The impact of a missing resource is determined by its importance for a page. For example, a large missing image has usually a greater impact than a small missing image. Different to the quality assessment setting in this chapter, the authors consider the more general case where the archived page is the only available resource for the assess-

ment. The approach calculates the damage for a web page as normalized value between 0 (no missing resource) and 1 (all resources are missing), where a resource is an image, a multimedia object, or a style sheet. The approach weighs resources according to a heuristically determined importance, which is based on image or multimedia object size or on the number of HTML classes that are not referenced by any of the retrieved style sheets. With the help of the original authors, we reimplemented the damage rating calculation as a user simulation script for our web archiver, which first scrolls down the page just like the script we used for creating the Webis-Web-Archive-17, and then calculates the damage using the current web page state in the browser. Since this measure is restricted to the archived page and exploits no information how the page looked when all resources were retrieved, we expect it to perform worse than the other measures. Also, the heuristics do not account for script files, which can have serious impact on the look and feel of a web page.

RMSE The second measure uses the screenshots taken during archiving and reproduction, computes the squared color difference for each pair of corresponding pixels, and then uses the root of the mean as reproduction quality score.¹⁵ Specifically, we use the corresponding option of ImageMagick¹⁶ to calculate the difference, and then employ the normalized RMSE values that range from 0 (identical images) to 1 (for each pair of pixels in the two images with the same coordinates, one pixel is purely white and the other one purely black). If one screenshot is smaller than the other one, it is enlarged by adding white pixels at the bottom of the image so that pixels can be directly matched by their image coordinates. Note that all screenshots in the dataset have already the same width as they were taken using the same browser viewport width. While we expect some correlation with the ground-truth reproduction quality, RMSE does not consider the position of pixels nor the context of neighboring pixels. These features are useful for a quality analysis, as main content is often placed mid-screen. However, the shape of coherent regions of deviating pixels hint at what kind of content is missing. Furthermore, a single missing advertisement banner at the top of the page can shift up the entire web page, causing the RMSE to become unjustifiably high (see Figure 2.2 on page 25 for an example).

Neural network The final measure uses machine learning to predict the reproduction quality from the screenshot differences. As the first machine

¹⁵RMSE is the abbreviation for “root of the mean of the squared error”

¹⁶<https://www.imagemagick.org>

learning approach applied to this task, we see this measure as a baseline for similar and more sophisticated measures that are yet to be developed. As a direct conclusion from the drawbacks of RMSE, we decided to employ a machine learning model that has been applied successfully to different image classification tasks, namely deep convolutional neural networks [155]. Since, however, fine-tuning a network to a new task is a research topic in its own right, we use the widespread and straightforward VGGNet network [219] instead. As it was shown that the fully connected layers at the end do not contribute to the performance [120], however, we omit them. The network structure and some implementation details are further described in [127]. As input images to the network, we create a single two-channel image, where one channel corresponds to the screenshot taken during archiving (in greyscale) and the other channel corresponds to the screenshot taken during reproduction (in greyscale). As the structure of VGGNet can only be trained and applied to images of a single size, we scaled down all created images to 384×128 pixels. For this size, icons with the usual width of 32 pixels are scaled down to a width of 3 pixels, which matches exactly the receptive field of the neurons in the convolutional layers of the network. In order to avoid skewing the images, we extended or cropped all screenshots to a height of 4098 pixels before the scaling. This height was chosen as the closest multiple of the image width (the currently most common browser window width of 1366 pixels) to the average image height of the archive images (4388 pixels). We converted all images to greyscale, since we do not expect color to play a major role for reproduction quality assessment, and found this to be indeed the case in our preliminary tests. We use the network in a 10-fold cross validation setting including all web pages¹⁷ and using the annotations gathered as labels for both training and evaluation.

3.1.5 Evaluation

Instead of manually reevaluating each web archiving tool (or each version of a web archiving tool), the quality criteria underlying the human assessment of reproduction quality may be learned with machine learning from the Webis-Web-Archive-17. If sufficiently successful, the resulting reproduction quality model may be applied in real time as a web archiving tool is deployed, used, and developed, allowing for severely shorter turnaround time and significant cost-savings compared to a manual evaluation. This section thus first compares the automated reproduction quality measures

¹⁷We also include the web pages without differences in the screenshots for training

introduced above with the dataset’s ground-truth, and then compares the three reproduction methods described in Section 2.2.5 on page 23 as per the measure that matches the ground-truth best, the neural network.

Comparison of Reproduction Quality Measures

The main goal of an automatic measure of reproduction quality is to compute quality scores that correlate with the true reproduction quality as per human assessment. The higher the correlation of automatic scores and true quality, the better the measure. Therefore, to test the three measures described above, we analyze their correlation with the human annotations on the Webis-Web-Archive-17. Figure 3.4 gives a visual impression of the correlation. For a quantitative result, we also calculate the Pearson correlation r with the human annotations for each of the three measures. As shown by the horizontal line for the linear model in Figure 3.4, the Brunelle15 measure is uncorrelated with the human annotations ($r = 0.00$). We believe that the main reason for the low r is that the measure was developed for a setting at which no information on how the web page originally looked is available, restricting the measure to guessing the importance of resources on mere hints. Also, the employed heuristics may suffer from genre dependence, where single heuristics that work well for all genres of web pages might just not exist. For comparison, the conceptually much simpler RMSE, which uses information on visual changes for the reproduction, achieves a much stronger correlation of $r = 0.48$. Given the aforementioned restrictions on what RMSE considers, this good result is somewhat surprising. Therefore, the extent to which pixels changed in a reproduction seems to be a strong feature for reproduction quality, and the cases in which the changes are misleading (e.g., due to content shifting up, which leads to a lot of pixel changes) seem to be in the minority. Finally, the best correlation is achieved by the neural network ($r = 0.57$). This is not surprising, as it also incorporates the most information on the web page visuals, and since it employs labeled data and learning theory to reach an empirical understanding of reproduction quality.

Table 3.3 provides a more detailed look at the effectiveness of the neural network measure, showing the scores underlying the respective scatter plot in Figure 3.4. For 4,698 of the 6,348 annotated reproductions of the Webis-Web-Archive-17, the measure agrees on the annotated score. This corresponds to a classification accuracy of 74.0%. For comparison, classifying all reproductions with the majority score (score 2, cf. Table 3.2) would yield an accuracy of 52.1%. Moreover, if not correct, the measure is most of

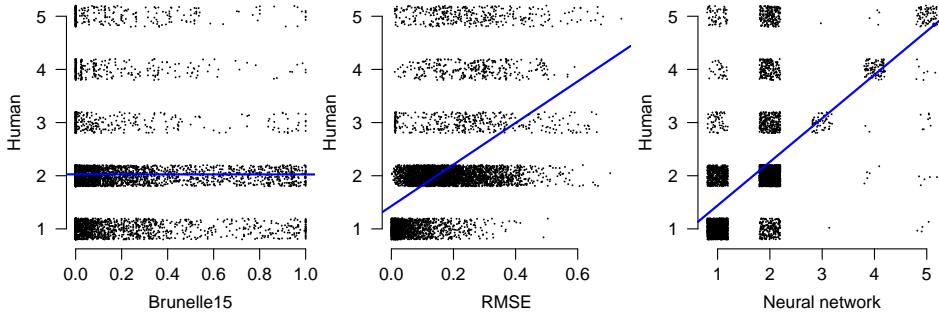


FIGURE 3.4: Scatter plots of human reproduction quality annotations over scores computed by the three automatic measures for each annotated reproduction in the Webis-Web-Archive-17. A small jitter is applied for human annotations and neural network scores to provide a better impression on the number of dots at each coordinate. The straight lines correspond to fitted linear models; the closer they are to the main diagonal, the better the quality measures.

TABLE 3.3: Confusion matrix of predicted and ground-truth reproduction quality scores for the neural network.

Truth	Estimated reproduction quality					Σ
	1	2	3	4	5	
1	1707	230	1	0	4	1942
2	535	2762	0	5	5	3307
3	62	294	55	2	9	422
4	33	183	0	95	7	318
5	57	218	1	4	79	359
Σ	2394	3687	57	106	104	6348

the time very close to the human annotation: only for 578 pages (9.1% of reproductions) the estimated score is more than one off of the ground-truth. Note that these calculations already omit the trivial cases without a difference in the screenshots. If the 3,652 trivial cases are included, the accuracy, for example, increases from 74.0% to 83.5%.

For an application of the neural network for immediate automatic reproduction quality assessment, also the trivial cases are relevant, but usually only two classes exist: the reproduction quality is still acceptable or not. Which score is still acceptable, however, depends on the specific application. Table 3.4 shows the achieved effectiveness as measured by standard metrics for different choices of a minimum desired quality. For debugging and fast improvement of the reproduction approach, the recall—percentage of all not acceptable reproductions that were identified—is the most important metric. A high recall (84.4%) is only achieved for the case of separating

TABLE 3.4: Effectiveness of the neural network quality measure for identifying not acceptable reproductions depending on what scores are considered to be minimally acceptable. Effectiveness is measured as accuracy, precision, recall, and the F-Measure (F_1). All values are calculated from the confusion matrix in Table 3.3 (including the 3,652 trivial cases for accuracy).

Minimum Quality	Accuraxy	Precision	Recall	F-Measure
1	90.8%	94.1%	84.4%	89.0%
2	91.4%	94.4%	22.9%	36.9%
3	94.8%	88.1%	27.3%	41.7%
4	97.0%	76.0%	22.0%	34.1%

quality-1 reproductions from others. Thus the neural network can be employed for debugging reproduction approaches that aim at this quality—which are probably most if not all approaches. On the other hand, the good precision values indicate that the neural network can be used to automatically detect which web pages fail to be reproduced with acceptable quality, which is important for generating large-scale web archives.

Comparison of Reproduction Approaches Using the Quality Measures

Section 2.2.5 discusses three different web page reproduction approaches—one of which we created. This section evaluates their performance.

As described in Section 2.2.5, we noticed differences in the reproduction quality of the existing reproduction approaches. In order to substantiate this impression, we used the reproduction quality measures to automatically assess the reproduction approaches over the entire Webis-Web-Archive-17. Since the neural network reproduction quality measure performed best, we used it for the quality assessment, demonstrating its usefulness for a comparative evaluation of alternative web archiving and reproduction approaches. While the use of the still less-than-optimal model may introduce some bias, we believe that all three reproduction approaches are similarly affected, yielding a fair comparison. This saves us significant resources, since otherwise we would have had to repeat the crowdsourcing assessment for each of the three alternative reproduction approaches in question.

Figure 3.5 shows the distribution of the quality scores for each reproduction approach. The achieved quality of the three approaches is rather similar with the custom approach reproducing the most pages with the best score. This visual impression is confirmed by the average scores that the reproduction approaches achieve: 1.48 (custom), 1.51 (warcprox) and

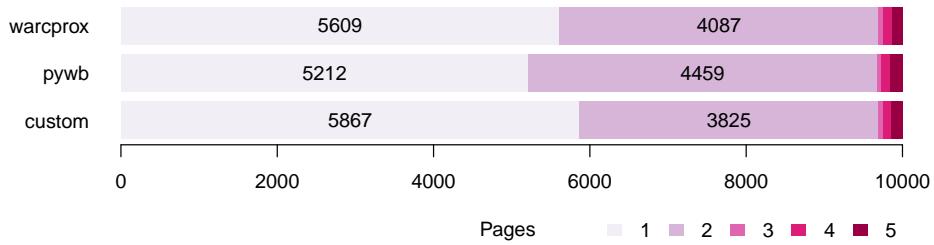


FIGURE 3.5: Distribution of reproduction quality scores for each of the tested reproduction approaches as assessed by the neural network measure.

1.55 (pywb). While this result does not suggest that a single reproduction approach is clearly preferable over the others, it allows for future analyses to pinpoint the common and different problems of these approaches.

3.1.6 Conclusion

The preservation of digital culture relies heavily on web data. However, for large-scale analyses, the ephemeral nature of the web poses a challenge to the reproducibility of insights gained from such data. While tools for web archiving exist, they are not tailored to the needs of scientists, nor has their effectiveness been systematically evaluated to date. We define a new quality assessment task, provide a tailored dataset for it, and introduce an evaluation measure.

As our analysis on 10,000 carefully sampled web pages shows,¹⁸ the reproduction of web pages from archives is far from perfect. To facilitate improvements in this regard, we cast the problem of assessing the effectiveness of web page reproduction software into the new task of immediate automatic reproduction quality assessment, where screenshots taken from the original web page and its reproduced version are used to judge the reproduction quality. With the Webis-Web-Archive-17, we provide the first benchmark dataset for this task, compare three measures for a respective automatic evaluation, and show that a neural network-based measure is able to automatically detect low reproduction quality, while achieving a high correlation with human annotations (Pearson’s $r = 0.57$), and while reaching a recall between 22% and 84% as well as a precision between 76% and 94%, dependent on the desired reproduction quality threshold.

Specifically, the approaches to fuzzy request-response matching need improvements, for example, by employing better heuristics, specifically tailored request-similarity measures, machine learning to classify which parts

¹⁸The dataset is publicly available at <https://doi.org/10.5281/zenodo.1002203>

of a request are important, or a mixture of all these. Moreover, the off-the-shelf neural network used for automatic reproduction quality assessment—which presents a strong baseline—should be tailored more to the task. For example, extending and cropping the screenshots might not be optimal for the assessment. Instead, more sophisticated approaches that use recurrent networks after the convolutional layer to handle inputs of arbitrary size might be a better fit [68]. Finally, it may be worthwhile to fine-tune the number of layers and their parameters.

As another improvement, popovers or splash screens should usually not be part of the archive, which could be handled by adding (possibly reactive) simulated interactions to the user simulation script that dismiss these overlays during reproduction. Furthermore, as content vanishes, archiving software should include a technique for detecting error pages that do not send the appropriate error code and alert the user if they are attempting to archive such an error page. Section 3.2 follows this lead.

Regarding the analysis of our tool, more detailed case studies for the different web page genres within our dataset will probably yield new insights into the difficulties and problems of current archive and reproduction technologies. While the automatic quality assessment is a solid first step towards an improvement of these technologies, an in-depth error analysis is needed to pinpoint the current flaws. However, as long as web technology continues to evolve, new challenges for web page archiving and reproduction will arise, demanding a continuous development of the tools and techniques that aim to create reproducible web corpora.

3.2 Analysis of Unexpected Content in Web Archives

Next to reproduction errors, a second challenge to the high-quality preservation of web pages is that, in some situations, web servers do not or only partially respond with what a human user would expect. Amongst others, the server may respond with unexpected content due to moved or removed pages (or sites), missing authorization, server errors, attack prevention mechanisms, service restrictions (e.g., by region and privacy laws), malfunctioning scripts, or random pop-ups (e.g., full-screen advertisements for related products). Though the HTTP protocol provides by status codes a means to tell the web client of some of these errors, which would make their identification straightforward, this best practice is frequently ignored.

To further improve current web archiving technology, this section introduces the concept of *content errors*, which refers to web pages whose

archived versions have unexpected content different from their originals. This section presents the first large scale analysis of a web crawl of 10.000 pages for content errors—the Webis Web Archive 2017 [127]. Using manual inspection and small annotation studies, we identified 5 different classes of content errors, and then annotated the entire crawl for these classes using crowdsourcing: error messages (4.5% of pages), pop-ups (3.9%), pages that largely consist of advertisements (1.1%), CAPTCHAs (0.8%), and loading indicators (0.5%). Combined, about 10% of pages are affected by content errors, which underlines the relevance of the problem. Given the large amount of web pages archived every day by the aforementioned initiatives, the detection of archiving errors in real time becomes crucial: content errors that are detected later on may not be repaired anymore, since the original page resources probably have disappeared by then.

As a step towards the automated detection of content errors at the time of archiving, we release the crowdsourced annotations as supplemental dataset to the Webis Web Archive 2017.¹⁹ The annotations can also be visually explored using our web service at <https://wwa17.webis.de>. As the Webis Web Archive 2017 contains the crawled web pages as HTML DOM, screenshot, and in WARC archives, the presented annotations allow researchers to develop, test, and compare content error detection technology using features based on all information that is available to an archiving tool, even the bare HTTP messages that were exchanged during the page’s archiving.

3.2.1 Content Errors

From the perspective of the user of an archiving tool, we say that a *content error* occurred if a to-be-archived URL yields a web page that is different from what the user expected. In particular, content errors occur (1) before or during archiving, (2) are always linked to single pages, and (3) depend on what the user sees as “normal” content for the page. This definition is in contrast to, for example, spam pages [58], for which a user would not expect content in the first place. It also distinguishes content errors from reproduction errors, which occur due to incomplete archiving (cf. Section 3.1). For the most part, archiving tools cannot prevent content errors, but only detect and then alert the user about them. In some cases, recovering from content errors is possible (e.g., by trying again later, investing more time, or automatically closing pop-ups). However, even the sophisticated archiving

¹⁹The Webis Web Archive 2017 is available at <https://doi.org/10.5281/zenodo.1002204> (includes web archive files, screenshots, and DOM trees for each page), whereas the content error annotations are available at <https://doi.org/10.5281/zenodo.2549837>

tools that are currently used by the Internet Archive or other initiatives perform no error detection at this moment. Nevertheless, some start to employ browser automation technology, which opens the door for automatic rectification of some types of content errors—especially pop-ups—in the future.

A classification by content error requires an error model that captures what a user does or does not expect. We adopt a *page-agnostic error model*: our hypothesized archiving tool user has a list of web page states and elements they do not expect on any archived web page (e.g., error messages). This is in contrast to a *page-specific error model*, where sets of erroneous states may be defined for an individual web page or website. For generic web archiving, devising page-specific error models may only be feasible for important pages. Our model results from a manual assessment of a sample of 300 screenshots of web pages contained in the Webis-Web-Archive-17, combined with the results of two pilot crowdsourcing studies that preceded the one described below:

Error messages The web page is not displayed correctly as indicated by an *explicit* error message. Clearly, a web page may also be displayed incorrectly without an error message, however, since a detection of this case would require prior knowledge of the “normal” state of the web page, i.e., a page-specific error model, we restrict ourselves to detect explicit error messages. A frequent cause are web pages that no longer exist, but where the server returns a substitute page with an error message (so called *soft 404* [28]). We distinguish web pages where the error message replaces the content (label: *very*), where the web page is still usable (label: *a bit*), and without error messages (label: *not*). Both *a bit* and *very* are content errors, as they suggest that the page’s functionality is impaired.

CAPTCHAs The web page asks the user to perform a task that is easy to solve for humans and supposedly very difficult for algorithms in order to block bots [238]. Archiving tools should give a warning for CAPTCHAs, so that their users can inspect the page and decide how to cope with the situation. A CAPTCHA may prevent access to the web page’s content (label: *very*) or just prevent certain actions (most often registration and commenting) on an actually well-working page (label: *a bit*). Thus only *very* signals a content error.

Pop-ups The web page shows a pop-up (e.g., overlay, banner, or modal). We distinguish pop-ups that prevent interaction with the page until closed (label: *very*) from those that do not (e.g., banners, cookie hints, or service

chats; label: a bit). Pop-ups lead to problems as some websites load content after closing them only. Pop-ups can also derange user simulation scripts that are used in web archiving to request all relevant resources [127]. If detected, the web archiving tool may try to automatically close the pop-up. Only *very* is a content error, as for *a bit* the main functionality of the web page is still intact.

Ad page The web page shows no real content but only ads. Such pages include domain parking pages, but also pages set up under a name similar to that of a well-known site to catch traffic arising from misspellings. There is no reason to keep them in an archive. This is a binary decision, so we distinguish *yes* and *no* only.

Loading indicators The web page has not been fully loaded and some placeholder is shown to signal that resources are still being loaded. Note that missing resources are not archived, as well. Loading indicators can usually be resolved by prolonging the archiving (or they turn into error messages if loading fails). This kind of error is relatively rare in our dataset as the employed archiving tool uses browser automation to scroll down the web page—thereby triggering all resources to be indeed requested—and then to wait for network traffic to cease [127]. As the annotator agreement for three classes (as used for error messages, CAPTCHAs, and pop-ups) was very low for loading indicators, we distinguish *yes* and *no* only.

3.2.2 Annotation Process

Using the aforementioned error model, we employed crowd workers to construct the first dataset of content errors. The Webis-Web-Archive-17 [127] contains 10,000 web pages sampled from the Common Crawl in a way which ensured that both well-known and less-known websites are included. Table 3.5 shows the distribution of content errors we identified in the web pages. Every web page was annotated by at least 5 different annotators who we recruited using Amazon’s Mechanical Turk. The annotation interface (cf. Figure 3.6) contained a scrollable and zoomable screenshot of the web page, radio buttons to label the content errors, and a text box for comments.

For quality assurance, we monitored annotators closely. If they took less than 10 seconds for a web page or mostly disagreed with others, we took a closer look at their annotations and—if we came to the conclusion they did not work honestly—rejected their results to be replaced by other annotators.

TABLE 3.5: Annotator agreement [110], post-annotation corrections, distribution of labels (content errors marked **bold**), and percentage of pages with the respective content error identified in the 10,000 web pages of the Webis-Web-Archive-17.

Content error	Agreement	Corrections	Distribution		% Error	
			No	Yes		
Ad page	0.65	329	9895	105	1.1	
Loading indicators	0.89	48	9950	50	0.5	
			Not	A bit	Very	
Pop-ups	0.82	394	9297	315	388	3.9
CAPTCHAs	0.91	124	9865	60	75	0.8
Error messages	0.89	331	9554	83	363	4.5

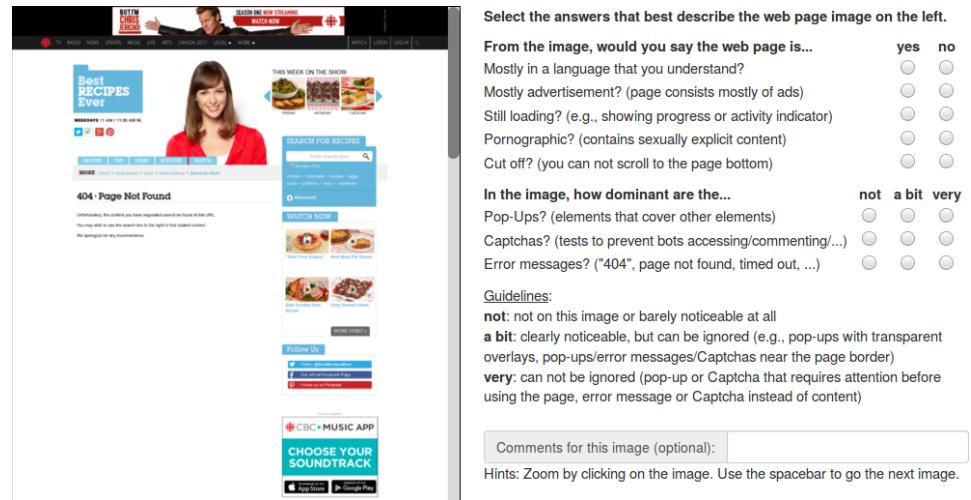


FIGURE 3.6: Annotation interface used by crowd annotators to label each of the 10,000 web pages with content errors.

Filtered items: 362 (15 page(s))

ID	Screenshot	Ad	Cut	Load	Porn	Popup	Captcha	Error	Tags
007442		no	no	no	no	not	not	very	
007166		no	no	no	no	not	not	very	
008556		no	no	no	no	not	not	very	
006331		no	no	no	no	not	not	very	
008148		no	no	no	no	not	not	very	
001630		no	no	no	no	not	not	very	
000622		no	no	no	no	not	not	very	
003104		no	no	no	no	not	not	very	

Filters are active [clear](#)

Filter by tag(s)

Enter tag name(s)

ID

id [Add](#)

Ad

no/yes [Add](#)

Cutoff

no/yes [Add](#)

Loading

no/yes [Add](#)

Pornographic

no/yes [Add](#)

Popup

not/abit/very [Add](#)

Captcha

not/abit/very [Add](#)

Error

not/abit/very [Add](#)

very [i](#) [x](#)

FIGURE 3.7: Curation interface used by us to assure consistent annotations.

About 10% of annotations were rejected, and a total of 747 unique annotators were recruited. Employing MACE [110], we measure a high worker agreement on all but ad pages, where only 0.65 agreement is achieved. To further improve consistency, we manually checked all cases where the annotators did not largely agree on a category using a special curation interface (cf. Figure 3.7) and corrected the annotations, if necessary. In total, we changed 1226 annotations in this step.

TABLE 3.6: Number of errors as identifiable by the error message.

Error	Status Code	Amount
Bad request	400	17
Unauthorized	401	4
Forbidden	403	17
Not found	404	204
Gone	410	23
\sum Client errors	4XX	265
Internal server error	500	14
Bad gateway	502	22
Service unavailable	503	13
Gateway timeout	504	8
Web server returns unknown error	520	1
Connection timeout	522	1
Invalid SSL certificate	526	1
\sum Server errors	5XX	60
Missing browser plugin		82
JavaScript error		7
\sum Browser-related errors		89

3.2.3 Analysis of Error Messages

Error messages describe a very broad category of content errors, so a more fine-grained categorization seems necessary to further the understanding of this kind of problems. We used the categorization provided by HTTP error codes as it is widely used and of course highly related to web pages. As Table 3.6 shows, we were able to match 325 of the 446 web pages that contain an error message to HTTP error codes (none of which appear in the actual web traffic). Of these, the most frequent error was that a resource did not exist (HTTP 404, 204 web pages). From the messages that we could not match with HTTP error codes, 89 were related to the employed browser, most prominently 82 web pages that required a certain browser plugin (e.g., a flash player) could be resolved by installing the respective plugin in the browser that is used for archiving. In the case of flash, which has been discontinued but is still widely used on the web, older browser versions are required that still support flash plugins. Finally, we were not able to interpret the remaining 32 error messages (e.g., “Error”).

In order to gain an intuition on how to detect web pages with error messages, we additionally created word clouds to visualize the differences



FIGURE 3.8: Word clouds of text extracted from (a) all web pages and (b) web pages with error messages that can not be ignored.

in word choice between web pages without errors from web pages with error messages that can not be ignored (annotation: very [dominant]). As Figure 3.8 shows, the web pages with errors contain more technical terms (“web server”, “live version”) and suggestions for resolving the errors (“please try”, “try minutes”). In particular, the frequent occurrence of “porn videos” is due to many errors being related to missing videos, many of which seem to be from porn sites. Still, the differences in the word clouds seem mostly intuitive, which may render a text-based automatic classification a sensible first step in future work.

3.2.4 Conclusion

This section defines content errors and shows that they are not uncommon, as they appear in roughly 10% of the web pages in the broadly sampled dataset we employ. Our crowdsourced annotations for 10,000 web pages presents the first step towards an automatic detection of content errors. We envision that such automatic detectors will be used as part of web archiving tools to alert their users of the errors or even to resolve them automatically. The analysed dataset allows to devise features using the text content, HTML DOM, screenshot, or HTTP messages exchanged for a page. For error detection, where automatic approaches already exist (e.g., [172]), the presented dataset allows to incorporate new features and re-evaluate on a more recent dataset. For other types of errors, the dataset allows to develop new approaches in the first place. Therefore, the dataset allows to improve over existing methods. As content errors are noise to many analyses of web pages, detecting such errors will benefit other applications, for example web search, as well.

3.3 Segmentation of the Pages in Web Archives

When visiting a web page, a key step for human comprehension is to identify its semantic units. Eye-tracking studies show that participants identify such units immediately upon perceiving a web page, then inspect them one at a time, often starting with navigation elements [179]. To create a comprehensible web page, it is thus important for its author to group its content into such comprehensible semantic units that are easy to identify by its visitors. Though qualified web designers do so in a professional manner, every web page author possesses an intuitive understanding of the basic principles of Gestalt that apply here, as these principles form an integral part of human perception [92]. Naturally, these semantic units, then called web page segments, also form the basis for various web content analysis tasks, like content extraction [20], template detection [165], and design mining [150]. Similarly, it is important for the preservation of a web page to preserve these units, too. Consequently, several approaches for web page segmentation have been developed over the past two decades.

The ongoing and rapid development of web technologies like Cascading Style Sheets (CSS) and JavaScript (JS) has considerably increased the possibilities of web design over the past years. The elements of a web page encoded in its HTML source code can be more or less arbitrarily rearranged in its visual appearance in the browser, so that no correspondence between the linear order of elements in the source code and its visual ordering can be presumed. Since the focus of web page authors are mostly the human visitors and much less so web content analysis algorithms, there is hardly any incentive to emphasize the semantic units in the web page's HTML code. Web page segmentation algorithms thus increasingly focus on the visual rendition of a to-be-segmented web page; a recent algorithm completely disregards the HTML code [59]. But even the classic VIPS algorithm [43], which was introduced in 2003, uses the positions of elements in the rendered web page as features for its segmentation.

Web page segmentation has been applied for various purposes throughout the information retrieval pipeline and beyond: Our review of related work—as well as the reviews of Fernandes et al. [80], Akpinar and Yesilada [7], and Bing et al. [34]—show that web page segmentation is used to improve crawling (template, duplicate, and change detection), information extraction (indexing, snippet generation, summarization, main content extraction, entity mining), page analysis (link analysis, design mining), and page synthesis (mobile screen adaptation, screen reading).

But despite the many publications that employ web page segmentation, the segmentation approaches have hardly been evaluated. Rather, the approaches have been judged “implicitly” by the increased performance induced in some downstream task that employs segmentation. Similarly, most segmentation algorithms have not been compared directly, and, in particular, no recent evaluations are at hand despite the constant evolution of web layouts. One reason for the missing evaluation is the lack of standard performance metrics for web page segmentation as well as suitable datasets in terms of size, diversity, and completeness of resources. Especially, we are unaware of any prior publication to assess web page segmentation in the context of preserving web pages. Moreover, the reliance of algorithms on rendering the web page has limited the reproducibility of web page segmentation experiments, but we here demonstrate how to overcome this problem through the use of web archiving technology. In essence, several algorithms use JavaScript to segment the web page as it is rendered in a browser. However, to reproduce this situation properly, the following elements have to be kept constant: (1) the web page’s complete source code (HTML, CSS, JS, images, etc.); (2) the browser, since different browsers and even different versions thereof render the same page differently; and (3) the browser’s environment variables, like the date or random numbers, which the web page might request from the browser. These are not trivial requirements to meet, but modern web archiving technology can provide for a stable reproduction of web pages as they were rendered in the past (cf. Section 2.2).

This section lays new foundations for the large-scale evaluation of web page segmentation algorithms. Our main contributions are as follows: (1) We revisit the concept of a web page segment (Section 3.3.2) and, based on this, propose an evaluation framework for web page segmentation that builds upon a single similarity measure for segmentations (Section 3.3.3). This measure is applicable for the calculation of annotator agreement, the fusion of segmentations into a ground truth, and the evaluation of a segmentation against such a ground truth. The framework can be adapted to specific downstream tasks and is publicly available.²⁰ (2) Based on a reference dataset of 8,490 archived web pages we construct the Webis Web Segmentation Corpus 2020 (Webis-WebSeg-20), a publicly available dataset of 42,450 manually created web page segmentations (five per page), via crowdsourcing (Section 3.3.4).²¹ This dataset outranges prior resources by an order of magnitude while being more objective at the same time through

²⁰Code: <https://github.com/webis-de/CIKM-20>

²¹Data: <https://doi.org/10.5281/zenodo.3354902>

the use of five independent segmentations that are fused into one. (3) We develop and present a reproducible empirical comparison of five segmentation algorithms, as well as an ensemble of them (Section 3.3.5). Moreover, we report on and show the importance of parameter tuning for the different algorithms. Code, documentation, and provenance data of these experiments are available online.²²

Among others, the results (Section 3.3.6) of the comparison show that the classic VIPS algorithm still performs best when tuned to the dataset, but also that purely visual approaches can reach a competitive performance. Moreover, in adjusting the evaluation to the requirements of different downstream tasks of web page segmentation, we find that purely visual approaches are already the new state-of-the-art for downstream tasks that rely on pixel-based segments, like design mining. One of these purely visual approaches, the MMDetection algorithm, is able to reach this high performance despite being trained for a very different kind of input document than web pages: photos. The ensemble of four of the algorithms under consideration, however, does not outperform its base algorithms. Upon closer inspection, most of the ground-truth segments are identified by at least one of the algorithms.

3.3.1 Background

Research on web page segmentation goes back almost two decades. First defined by Kovacevic et al. [145], similar problems have even been tackled beforehand in information extraction (e.g., cf. [75]). Still, the community has not agreed on evaluation procedures, nor created commonly used benchmarks, as we detail below.

Algorithms for web page segmentation Algorithms use structural features based on the DOM tree and the textual content, and visual features extracted from renderings of individual nodes as well as the entire web page. Most algorithms use the DOM tree structure in some way, for example to identify headings [158], block nodes [7, 43], or regularities [80], and to compute the tree depth [146] or the tree distance [104] of nodes. Other algorithms use the text density [143] or visual appearance of DOM nodes when rendered (e.g., their size or color; Baluja [25], Zeleny et al. [256]). Few algorithms exclusively exploit visual cues, e.g., using edge detection on screen-

²²Code, documentation: <https://github.com/webis-de/ecir21-an-empirical-comparison-of-web-page-segmentation-algorithms>
Provenance data: <https://doi.org/10.5281/zenodo.4146889>

shots [45, 60]. Indeed, recent publications argue that only visual features provide for the necessary robustness for a generalizable algorithm [60, 256], but this claim has not been verified. Our dataset provides the resources required by all the various approaches, enabling a fair and comprehensive comparison. Although a hierarchical segmentation of a web page is conceivable, and although some algorithms hierarchically split a web page into smaller segments (e.g., [25, 34, 43]), all proposed algorithms output a single segmentation per page. We therefore adopt this view, and leave hierarchical segmentation for future work.

Datasets for web page segmentation Several datasets have been created for web page segmentation, but none has become a standard benchmark. Instead, most algorithms come with a new dataset for their evaluation (cf. Table 3.7). Issues that prevent the reuse of the existing datasets include missing data sources (e.g., no screenshots), bias due to heuristic annotations, no ground truth annotations, unavailability, and a non-representative sample (e.g., only a few specific websites). None of the previously published datasets combines completeness, reliability, diversity, and scale. Even the very large dataset of Fernandes et al. [80] lacks diversity, since their annotation process presumes all web pages to be homogeneous. Regarding tools for manual segmentation, only Sanoja and Gançarski [207] have proposed one: it allows to create, resize, and move rectangles on a screenshot to specify segments. Inspired by their approach, we integrate our version with Mechanical Turk to enable manual segmentation at scale via crowdsourcing.²³

Evaluation of web page segmentation Previous attempts to evaluate web page segmentations fall short in some respects or others. Some resort to a posteriori human judgment of detected segments (e.g., Cai et al. [43]), which does not scale well and yields hardly replicable measurements. Others evaluate based on the web page’s text only, which allows for using existing evaluation measures for this task (e.g., Kohlschütter and Nejdl [143], Manabe and Tajima [167]), but restricts the evaluation to text-only segments. Yet others measure the overlap between an automatic segmentation and a ground truth [256], or count matching cases (one-to-one, one-to-many, zero-to-one, etc.; Sanoja and Gançarski [208]). Such matching measures, however, unfairly handle cases of over- and undersegmentation: The measure proposed by Zeleny et al. [256] penalizes splitting a ground truth

²³Mechanical Turk features a built-in image segmentation interface, but we found that an interface tailored to web page segmentation allows for a much quicker annotation.

TABLE 3.7: Overview of existing segmentation datasets; their enclosed data: HTML code, resources (CSS, images, etc.), screenshot; segment annotations: manual, heuristic, or a posteriori judgment; availability: publicly, on request, or not anymore as per correspondence with the authors; and the numbers of websites (if given) and web pages.

Author	Reference	Year	Characteristics	Sites	Pages	
Kovacevic et al.	[145]	2002			n/a	515
Cai et al.	[43]	2003			n/a	140
Vadrevu et al.	[235]	2005			n/a	240
Hattori et al.	[104]	2007			100	100
Chakrabarti et al.	[47]	2008			n/a	105
Kohlschütter and Nejdl	[143]	2008			102	111
Cao et al.	[45]	2010			n/a	20
Spengler and Gallinari	[222]	2010			177	604
Fernandes et al.	[80]	2011			15	457,542
Pasupathi et al.	[187]	2012			10	15
Sanoja and Gançarski	[207]	2013			n/a	100
Bing et al.	[34]	2014			n/a	1,000
Kreuzer et al.	[146]	2015			59	152
Manabe and Tajima	[167]	2015			981	1,219
Sanoja and Gançarski	[208]	2015			125	125
Cormier et al.	[59]	2016			50	50
Cormier et al.	[60]	2017			100	100
Sanoja and Gançarski	[209]	2017			n/a	40
Zeleny et al.	[256]	2017			5	800
Andrew et al.	[15]	2019			n/a	50
Webis-WebSeg-20		2020		4,824	8,490	

segment into several small ones more than returning just one of the small segments. The measure of Sanoja and Gançarski does not penalize splitting a ground truth segment at all, making it trivial to achieve the maximum score.

While most authors use evaluation measures of some kind to assess how well an automatic segmentation matches a human one, the assessment of human segmentations is typically lacking. For most datasets, each page was annotated by a single annotator only. While Zeleny et al. [256] employed three annotators per page, they treat each segmentation as alternative ground truth. Manabe and Tajima [167] calculated the annotator agreement for a few test pages, but only for segments that directly correspond to HTML block elements. An algorithm to fuse annotations of different workers into a single ground truth segmentation has not been considered.

A number of publications that propose a new web page segmentation algorithm compare it with the classic VIPS algorithm [43] (e.g., [59, 167,

256]), which can thus be considered closest to a standard baseline. In the original publication, VIPS has been evaluated with a three-scale human assessment on only 140 web pages: According to the assessors, 61% of web pages were segmented “perfectly,” whereas just 3% “failed.” Such an assessment is unfortunately hardly reproducible. Zeleny et al. [256] perform an empirical comparison of their algorithms with VIPS on 800 semi-automatically annotated web pages. Their performance measure, F , is closely related to $F_{B^3}^*$ (nodes), employed in this chapter, and indeed, a similar performance is measured for VIPS: 0.71 by Zeleny et al., and 0.70 here. For their visual-edge-based algorithm, Cormier et al. [59] compare its segmentations with that of VIPS on 47 web pages using an adapted Earth Mover’s Distance as performance measure. They find, that, though there is some agreement, their algorithm “tends to produce results significantly different from VIPS.” Our evaluation in Section 3.3.6 also shows such a difference. Manabe and Tajima [167] compare the performance of their HEPS algorithm with that of VIPS for the task of identifying web page blocks—i.e., textual segments with headings. In their comparison on 1219 web pages, they find that HEPS clearly outperforms VIPS for exactly identifying such blocks: block precision is 0.59 (HEPS) vs. 0.22 (VIPS), and block recall is 0.56 vs. 0.07. This is in contrast to our results, which indicate a superior performance of VIPS over HEPS, not only for a text-based evaluation. A possible explanation lies in their different approach to ground-truth creation, which is tailored towards the mentioned header-based blocks.

However, no large-scale comparison of web page segmentation algorithms exists so far. We attribute this situation to a lack of generic, standardized datasets, a lack of a common view on how to measure algorithm performance, and a lack of reproducible evaluation procedures. Reviewing the related work beyond the aforementioned papers, evaluation datasets and performance measures have usually been created in an ad-hoc manner, and with respect to just one of the various downstream tasks of web page segmentation, which has led to several very focused datasets and many incompatible performance measures. The problem of reproducibility has, to the best of our knowledge, scarcely been tackled in the relevant literature so far: Only Zeleny et al. [256] attempt to reduce the influence of different browsers by using the same rendering engine for all algorithms. This work is the first to consider web archiving technology for web page segmentation, addressing its reproducibility problem for the first time.

Segmentation outside the web Beyond web pages, segmentation tasks are studied for scanned print documents and generic images. Unlike for

web pages, typically no semi-structured representation like the HTML source is available for either. At ICDAR, a long-running competition addresses the segmentation of scanned print documents featuring complex typesetting [56]. However, there is much less ambiguity about the level of granularity in print documents and the evaluation measures thus focus on segment matching. Generic image segmentation (e.g., of photos) is often cast as an object recognition task: images rarely contain text and objects are rarely rectangular. Unlike for web page segmentation, the evaluation measures employed for generic image segmentation match boundary pixels [168] or objects directly, using huge datasets of hundreds of thousands of images like Microsoft COCO [160]

3.3.2 Concept Formation: Web Page Segment

Nine of the 19 publications listed in Table 3.7 give—explicitly or implicitly—a definition of what a web page segment is. The most common one (though used in only four publications) is that of a visual “block” with coherent content [47, 146, 158, 256]. Other definitions characterize segments by their edges [59, 60], as being semantically self-contained [80], as distinct [187], or as labeled with a heading [167]. Only two papers resort to HTML/DOM elements or sub-trees as segment building blocks [47, 146]. Seven of the nine definitions require a segment to be cohesive, and two define a segment as being “different” to other parts of a web page. Most of the definitions do not include information about the desired level of granularity, probably because different downstream applications have different requirements [256].

Altogether, the concept of a segment is not precisely captured: Does an individual menu item count as a segment, or need all menu items be combined, or is the entire sidebar to which the menu belongs the “true” segment? It is also unclear whether a more precise specification would be meaningful across web page genres. Note in this regard that even the terminology to describe granularity levels is used inconsistently: Kreuzer et al. [146], for instance, differentiate between high-level and sub-level segments, while other authors resort to exemplifying the desired level of granularity, such as “header”, “left menu”, etc., as in Kovacevic et al. [145].

In light of the ambiguities and limitations of the existing segment definitions, we refrained from proposed a tenth definition, but opted instead for a concept formation approach based on crowdsourcing. For this purpose, each page in our dataset has been annotated by five annotators, providing us with a rich source of information to analyze what a human onlooker considers a plausible segment and granularity level, respectively:

A web page segment is a part of a web page containing those elements that belong together as per agreement among a majority of viewers.

This concept of a web page segment is grounded on well-known layout patterns (“header”, “main”, “footer”, etc.) and human perception habits (such as Gestalt principles like proximity [92]). As we show below, we indeed get strong agreement among independent annotators. Our dataset thus allows for the development of web page segmentation algorithms that operationalize this concept.

3.3.3 Evaluation Framework for Web Page Segmentations

The creation and usage of a dataset that adheres to the concept of page segments as introduced above requires answers to the following three questions: How to measure the agreement of users? How to fuse single segmentations into a coherent ground truth? How to evaluate this ground truth?

As shown at the end of this section, the answer to all of these questions boils down to measuring the similarity of two page segmentations. To choose a measure of segmentation similarity, we cast web page segmentation as a clustering task and draw from the theoretical foundations of cluster similarity measures. In order to identify the objects that are to be clustered into clusters corresponding to page segments, we begin by studying alternative candidates for atomic elements of a web page.

Atomic Elements of a Web Page

The first component of our framework is the selection of the “atomic” elements of a web page—the nature of which is deliberately left open in our concept of a web page segment. We identify three alternative sets of atomic elements that can be clustered to form segments of a web page: (1) pixels, (2) DOM nodes, and (3) characters. Besides the entire sets, also defined subsets might be considered dependent on the downstream task; for instance, background pixels may be considered unimportant. As our dataset is task-agnostic, we repeat all analyses for a selection of five sets that cover a variety of intuitions, namely three pixel subsets, and one each of DOM nodes and characters (see Figure 3.9 for an illustration):

- **Pixels** The three pixel subsets include (1) all pixels of a web page’s screenshot, and pixels at (2) fine-grained, and (3) coarse-grained visual edges as per Canny’s edge detection algorithm [44], which is best-suited for web pages [45]. Fine-grained edges include the outline

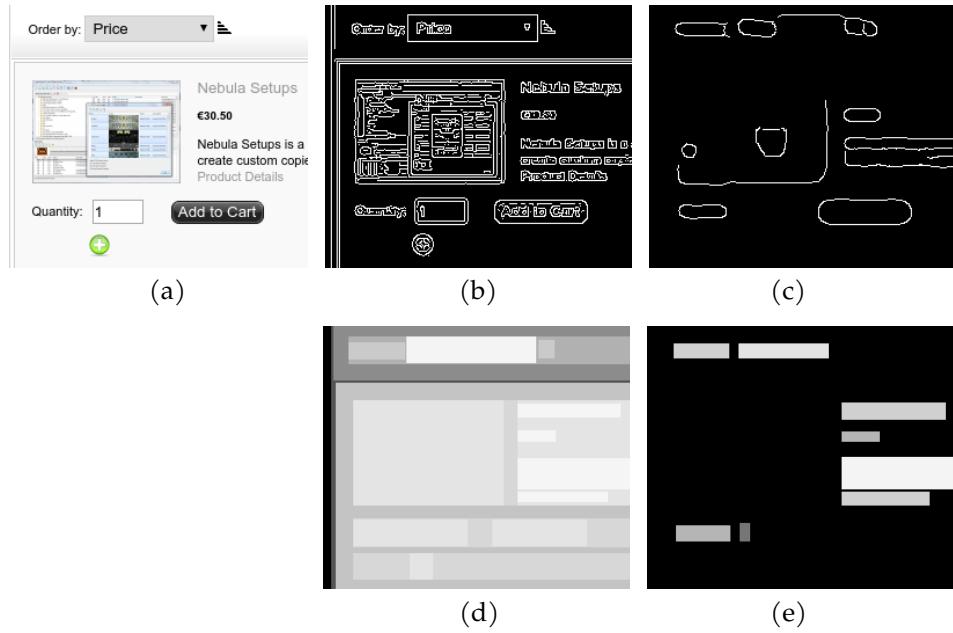


FIGURE 3.9: Visualization of atomic elements for (a) an exemplary page excerpt: (b) fine- and (c) coarse-grained edges; (d) DOM nodes; and (e) characters (text nodes). Lighter pixels indicate more elements at the respective position in (a). The image for all pixel elements would be completely white.

of characters at 10pt font size, and coarse-grained edges only lines of text.²⁴ Edges are here used as an indicator for the content density of segments.

- **DOM nodes** All visible DOM nodes of a web page, i.e., element and text nodes. As the DOM is organized hierarchically, more nodes lie in regions that are more deeply structured.
- **Characters** All characters on a web page.

These sets of elements capture intuitive choices for generic image-based web page segmentations (all pixels, e.g., for design mining), for specific image-based segmentation where background pixels are irrelevant (edge pixels, e.g., for mobile screen adaptation), for structure-based segmentation (visible DOM nodes, e.g., for information extraction), and for text-based segmentation (characters, e.g., for screen reading).

²⁴Both parameter sets have the same radius of 0 and lower percentage of 1. For fine-grained edges, the upper percentage is 2 and $\sigma = 1$; for coarse-grained edges, the upper percentage is 16 and $\sigma = 5$ to counteract the increased roughness of edge lines.

Web Page Segmentation Similarity

Like in clustering, measuring segmentation similarity in our framework is agnostic of the nature of the atomic elements. Given a web page p , let $E = \{e_1, \dots, e_n\}$ be the set of its atomic elements. Then $S = \{s_1, \dots, s_m\}$ denotes a (possibly partial and/or overlapping) segmentation of p into segments $s_i \subseteq E$. Given two segmentations S and S^* of the same page p , we identify the extended BCubed F_1 -score by Amigó et al. [14], F_{B^3} , an extrinsic cluster evaluation measure, as particularly suited to measure segmentation similarity. Specifically, F_{B^3} has been shown to fulfill all requirements for a segmentation similarity measure that we gathered in our literature review and some more [14]: it handles both partial segmentations and overlapping—and thus also nested—segments, and is robust to trivial segmentations (e.g., using every pixel as own segment, or one segment that covers the entire page). Moreover, as F_{B^3} is symmetrical, it can be used as a generic segmentation similarity measure, too.

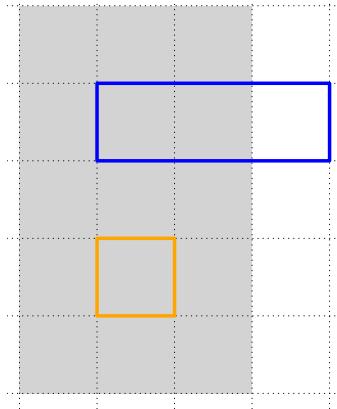
Analogous to the well-known F_1 -score, F_{B^3} is the harmonic mean of the extended BCubed precision (P_{B^3}) and recall (R_{B^3}):

$$P_{B^3}(S, S^*) = \frac{1}{|E^S|} \sum_{e \in E^S} \left(\frac{1}{|E_e^S|} \sum_{e' \in E_e^S} \left(\frac{\min(|S_e \cap S_{e'}|, |S_e^* \cap S_{e'}^*|)}{|S_e \cap S_{e'}|} \right) \right);$$

$$R_{B^3}(S, S^*) = P_{B^3}(S^*, S); \quad F_{B^3}(S, S^*) = \frac{2 \cdot P_{B^3}(S, S^*) \cdot R_{B^3}(S, S^*)}{P_{B^3}(S, S^*) + R_{B^3}(S, S^*)};$$

where $S_e \subseteq S$ is the subset of segments that contain element e , $E^S \subseteq E$ is the subset of elements that are part of at least one segment in S , and $E_e^S \subset E$ is the subset of elements that accompany element e in at least one segment in S . Formally, $S_e = \{s \mid s \in S \wedge e \in s\}$, $E^S = \{e \mid e \in E \wedge S_e \neq \emptyset\}$, and $E_e^S = \{e' \mid e' \in E \wedge S_e \cap S_{e'} \neq \emptyset\}$. For illustration, consider the case of non-overlapping segments: $|S_e \cap S_{e'}|$ is 1 if and only if e and e' are in the same segment in S , whereas $\min(|S_e \cap S_{e'}|, |S_e^* \cap S_{e'}^*|)$ is 1 if and only if they are in the same segment in both S and S^* .

Precision (P_{B^3}) and recall (R_{B^3}) offer helpful insight into the achieved F_{B^3} . Specifically, P_{B^3} ignores errors of strict oversegmentation (i.e., a ground truth segment is split), while R_{B^3} ignores errors of strict undersegmentation (i.e., ground truth segments are merged). Therefore, by comparison with F_{B^3} , these measures show the extent of over- and underseg-



$$\begin{aligned}
 P_{B^3} &= \frac{2}{4} \left(\frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 0 \right) + \frac{1}{4} \left(\frac{2}{3} \cdot 0 + \frac{1}{3} \cdot 0 \right) \\
 &\quad + \frac{1}{4} \left(\frac{1}{1} \cdot 1 \right) = \frac{7}{12} \\
 R_{B^3} &= \frac{12}{15} \left(\frac{12}{15} \cdot 0 + \frac{2}{15} \cdot 0 + \frac{1}{15} \cdot 0 \right) \\
 &\quad + \frac{2}{15} \left(\frac{12}{15} \cdot 0 + \frac{2}{15} \cdot 1 + \frac{1}{15} \cdot 0 \right) \\
 &\quad + \frac{1}{15} \left(\frac{12}{15} \cdot 0 + \frac{2}{15} \cdot 0 + \frac{1}{15} \cdot 1 \right) = \frac{5}{225}
 \end{aligned}$$

FIGURE 3.10: Toy example for calculating the BCubed precision and recall of a segmentation with two segments (colored frames) compared to a one-segment ground truth (shaded area) for pixels. The equations are highlighted with the corresponding color frames and shades.

mentation, and can thus directly inform the parameter optimization of the employed approach.

Optimizations The number of operations to calculate F_{B^3} grows quadratically with the number of atomic elements, so we developed optimizations to speed up the calculation. For pixels as atomic elements, we determine all largest regions where no segmentation divides these regions. The fraction in the calculation of P_{B^3} is the same for all elements of such a region. Therefore, we need to calculate this fraction only once for each pair of regions and just need to multiply the result by the product of the regions' areas. Figure 3.10 shows a toy example to exemplify the calculation. The same applies when using edges as atomic elements, except for using only the number of edge pixels in the regions instead of the area. For characters as atomic elements, we resort to DOM text nodes weighted by the number of characters within—analogous to how we use areas instead of pixels. Note that this method is an approximation for the very rare segmentation approaches that could potentially divide up text nodes in segmentation.

Unlike in the images that an edge detector produces, visual edges actually have no width, which frequently causes the edges of DOM nodes to appear a bit outside of the nodes' area in a generated image. To account for this fact, we grow the regions by two pixels before counting the number of contained edge pixels, which we tested to indeed capture nearly all relevant edge pixels.

Application of the Segmentation Similarity

The measure of segmentation similarity introduced above can be applied to answer the three questions from the start of this section: How to measure the agreement of users? How to fuse single segmentations into a coherent ground truth? How to evaluate this ground truth? Our released code contains a program for each of them.

Annotator agreement To judge whether two or more annotators were able to work consistently, their agreement is measured. Using F_{B^3} , we follow the example of popular agreement measures for text annotations like Krippendorff's α [148] and compute the average pairwise similarity of the segmentations. Specifically, for a set of segmentations \mathcal{S} of the same web page, we define segmentation agreement as follows:

$$\text{Agreement}(\mathcal{S}) = \frac{1}{|\mathcal{S}| \cdot (|\mathcal{S}| - 1)} \sum_{S \in \mathcal{S}} \sum_{S' \in \mathcal{S} \setminus S} F_{B^3}(S, S')$$

The agreement of an entire dataset is then calculated as the average agreement over all web pages.

In order to analyze how much of the quantified disagreement (calculated using F_{B^3}) is due to different annotation granularities, F_{B^3} can be replaced by $\max(P_{B^3}, R_{B^3})$. As mentioned above, P_{B^3} and R_{B^3} ignore errors from strict over- and undersegmentation, respectively. Therefore, an additional analysis with $\max(P_{B^3}, R_{B^3})$ yields insights into whether annotators disagreed on which elements belong together, and on the level of granularity.

Segmentation fusion Our concept of web page segments stated above is based on the majority agreement, and our framework employs F_{B^3} to fuse several segmentations into such a majority segmentation. Fused segments should contain those atomic elements which also the majority of annotators put in one segment. This intuition provides for a similarity of two atomic elements: the fraction of annotators who put those two elements in one segment. We then fuse elements and segments with a similarity exceeding a threshold (θ_s) of one half. This fusion process corresponds to the well-known family of hierarchical agglomerative clustering algorithms [114], which relies on the similarities only and does not need a vector representation of segments. Further following the principle of majority, we fuse only those atomic elements that are in segments for a majority of annotators. We analyze the effect of both θ_s and this annotator threshold for our dataset in Section 3.3.4.

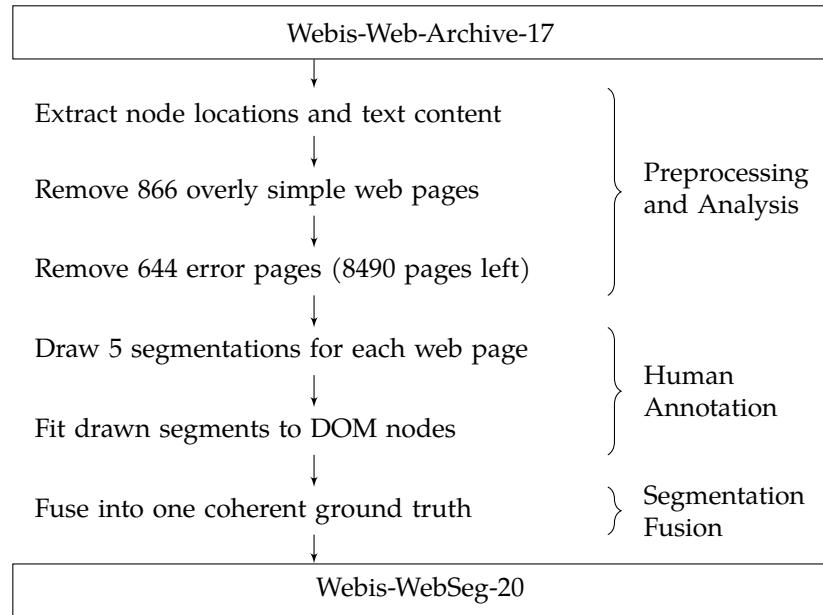


FIGURE 3.11: The Webis-WebSeg-20 construction process.

The choice of a specific hierarchical clustering algorithm matters only in the rare case of disagreeing majorities.²⁵ In such cases, the most basic hierarchical algorithms segment all elements together (single-link), or arbitrarily choose the segmentation of one majority (complete-link). Since both is not desirable, we employ the simple average or UPGMA algorithm [220], which tends toward the segmentation that groups more elements together.

Segmentation evaluation Analogous to its purpose in clustering, we use F_{B^3} as a measure of the quality of some segmentation S compared to a ground truth segmentation S^* of the same page.

3.3.4 The Webis-WebSeg-20 Dataset

Starting point for the construction of our dataset is the Webis-Web-Archive-17. It is a web archive comprising 10,000 pages from 5,516 sites, obtained via a stratified sample from top-ranked and low-ranked sites as per their Alexa ranking (alexa.com). Our dataset has been constructed in three steps: preprocessing, human annotation, and segmentation fusion. Figure 3.11 provides an overview of this process as it is detailed below.

²⁵Given three element sets, E_1, E_2, E_3 , let $\alpha(E_i)$ be the fraction of annotators that put all elements of E_i in one segment. Then “disagreeing majorities” in its simplest form is a case where $\alpha(E_1 \cup E_2) > 0.5, \alpha(E_2 \cup E_3) > 0.5$, but also $\alpha(E_1 \cup E_2 \cup E_3) < 0.5$.

Preprocessing and Web Page Analysis

Although the web pages of our dataset are already contained in the original web archive, not all resources for web page segmentation are readily available, and not all pages in the archive are suited for a web page segmentation dataset. Specifically, we reproduced all pages within a browser and extracted all DOM nodes from the rendered pages, their textual content, and their locations (i.e., bounding boxes) on the accompanying screenshot. In spot checks on 100 pages, we manually verified the locations' accuracy.

During our review, we identified two problematic cases of web pages with respect to segmentation, namely simple pages and error pages. We use "simple" to refer to web pages that have not enough content to justify a segmentation, and we therefore exclude from our dataset. Error pages are pages which clearly miss or have wrong main content. We expect that, in a page analysis pipeline, such pages will be identified and re-crawled ahead of the segmentation. We thus omitted 866 simple pages and 644 error pages, which were identified via an analysis of page complexity outlined below, and the annotation from Section 3.2, respectively.

To check that the dataset represents a broad sample of web pages, and to investigate page complexity, we analyze the amount of DOM nodes and the pixel height.²⁶ Further spot checks confirmed the intuition that simple pages have only a few DOM nodes, which allows to adjust the threshold for inclusion in our dataset accordingly: Figure 3.12 a shows a page bordering on simplicity. In Figure 3.12 b and 3.12 c, we observe a seemingly natural log normal distribution both for pages across the amount of DOM nodes and pixel heights. The exceptional high number of pages with a height of 16,384 pixels is due to infinite scrolling pages, where the archiving tool stopped scrolling. Error pages follow somewhat the overall distribution of pages. As one would assume, the correlation of number of DOM nodes and pixel height (Figure 3.12 d) is fairly strong, as indicated by the fitted log-linear model (straight line) and Pearson correlation.

Human Annotation

For humans, segmenting a single web page is fairly straightforward. As Kreuzer et al. [146] observe: "Human beings are very good at partitioning: even if a website is in a language we are not familiar with, it is clear to us what is an advertisement, what is a menu, and so on." In order to scale up such manual segmentation to 8,490 web pages while avoiding annotator

²⁶All pages have the same width of 1366 pixels as per the web archiver employed.

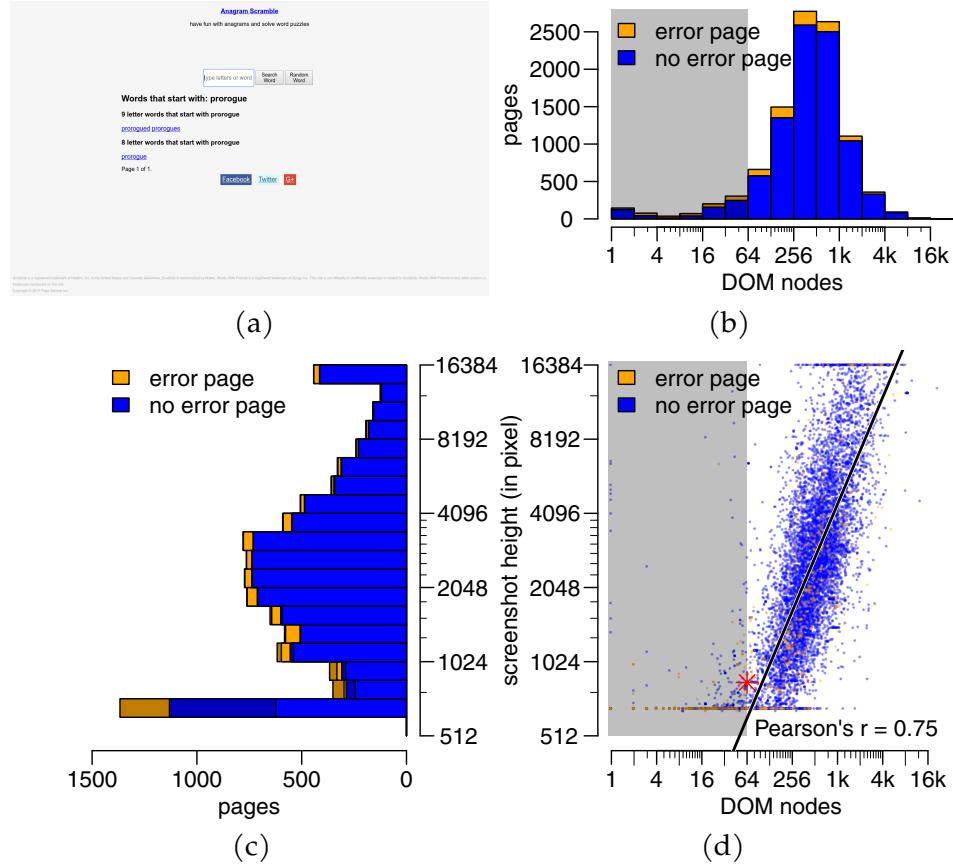


FIGURE 3.12: (a) Example page bordering on simplicity; (b,c) page frequency distribution, and (d) scatter plot over DOM nodes and pixel height. The page of (a) is marked as a red star in (d). The simple pages are within the shaded area and the shaded bars, respectively, and error pages are depicted orange.

bias (e.g., systematic errors), we employ crowdsourcing. We used Amazon’s Mechanical Turk and developed a tailored annotator interface that allows for drawing bounding boxes on web page screenshots, as well as a reviewer interface that allows for quality control by visualizing the segmentations. We further developed a reliable mapping of hand-drawn segments to their corresponding DOM nodes and assessed the annotation quality through measuring inter-annotator agreement.

Task setup Like in Section 3.1.3, we employed Amazon’s Mechanical Turk for the annotation. In pilot experiments, we found that our task does not require expert workers, so we only required workers to meet a low bar of having at least 100 previously approved HITs—a very low bar.

To ensure equal workload per HIT despite the vastly different pixel heights of the screenshots (see Figure 3.12), we employed a bin-packing algorithm to distribute the web pages so that every HIT contained web pages that have a combined pixel height of approximately 16,384 pixels, the maximum pixel height of the screenshots in our dataset. On average, a HIT contained five web pages. During our pilot experiments, we determined that workers needed 11.2 minutes per HIT on average. The payment per HIT was set to \$0.75 for an hourly rate of \$4, which is 13 times the minimum wage of India, and 3 times that of the Phillipines,²⁷ the two top countries of origin of workers from developing countries [71].

Regarding the potential ambiguity in web page segmentation due to different levels of granularity, and to study this phenomenon, every web page has been annotated by five independent annotators. Altogether, with 5,231 assignments, we collected 42,450 segmentations (encompassing 627,080 segments) for the 8,490 pages, which took 976 annotator hours at a total cost of about \$8,500.

Annotator interface Figure 3.13 shows the instructions given to the annotators, which include an animation that exemplifies the creation and adjustment of segments. Below the instructions, the screenshot was displayed on which the annotators had to draw segments as translucent blue rectangles just like in the animation.

The design of the annotator interface was optimized in pilot experiments for simplicity and physical ease. We first used a direct selection of DOM nodes to specify segments, but this interface required complex multi-selections and also confused annotators who lack knowledge in HTML. We hence asked the annotators to draw free-hand rectangles instead, requiring a subsequent step to resolve inaccuracies from the drawing and to map the annotations to the DOM. To ease drawing, we employed a click-move-click interaction, which allows the index finger to remain relaxed almost entirely, enabling fast work for a prolonged time. Though annotators could nest rectangles, this happened in less than 3% of annotations.

Reviewer interface Figure 3.14 illustrates the reviewer interface that we built in order to monitor annotation progress and quality.²⁸ To quickly check up on annotators, we introduced one test page in each HIT, for which

²⁷https://en.wikipedia.org/wiki/List_of_minimum_wages_by_country

²⁸Mechanical Turk does not include sophisticated reviewing interfaces, so that we used our MTurk Manager (<https://github.com/webis-de/mturk-manager>), an open-source project that assists requesters in carrying out large-scale crowdsourcing tasks.

What to do

- Draw rectangles around parts of the page that belong together.
- Draw separate rectangles for different parts, like for important content, controls, and ads.
- Make sure not to miss any part.

How to do it

Click anywhere on the screenshot to start drawing a rectangle. Move the mouse to draw the rectangle (it will stick to your mouse pointer). Click another time to end drawing the rectangle. You can rearrange, resize and delete rectangles. If you are drawing a rectangle and want to cancel press the escape key **ESC**.



FIGURE 3.13: Annotator instructions including a GIF animation which exemplifies the task and annotation process.

		6:32	Info Anno	Approve	Reject internally	Reject ▾
		15:29	Info Anno	Approve	Reject internally	Reject ▾
		54:50	Info Anno	Approve	Reject internally	Reject ▾
		2:31:12	Info Anno	Approve	Reject internally	Reject ▾
		28:56	Info Anno	Approve	Reject internally	Reject ▾
		40:24	Info Anno	Approve	Reject internally	Reject ▾
		16:14	Info Anno	Approve	Reject internally	Reject ▾

FIGURE 3.14: Reviewer interface, showing one assignment per row: annotator ID as image; reference (blue boxes) and annotation (red frames) for the test page (rotated and scaled); number of comments; time taken; buttons to show comments and annotator information, to show annotations for non-test pages, and to approve or reject the assignment.

TABLE 3.8: Annotator agreement by type of atomic elements and pairwise measure within the agreement measure.

Agreement measure	Atomic page elements				
	<i>pixels</i>	<i>edges_{fine}</i>	<i>edges_{coarse}</i>	<i>nodes</i>	<i>chars</i>
F_{B^3}	0.65	0.73	0.73	0.74	0.78
$\max(P_{B^3}, R_{B^3})$	0.94	0.96	0.96	0.95	0.97

the reviewer interface shows both reference and annotator segmentation. As test pages, we started with segmentations created by ourselves, and then iteratively integrated more test pages where annotators largely agreed. If an annotator segmented the test page badly, we inspected the other annotations and meta data to judge whether they spent effort on solving the task. If so, we still excluded the annotation from the dataset, but paid the annotator for their fair work (“internal rejection”). In order to gather especially good annotators, all annotators were limited to ten tasks until we reviewed their tasks, and could only continue if most were approved. In total, we approved 5,231 assignments, internally rejected 6,152, and openly rejected 540.

Fitting to DOM nodes In order to map the inaccurately drawn segment rectangles to DOM nodes, we treat each DOM node as part of the segment if at least a fraction θ_c of its visible area overlaps with the rectangle. We optimize θ_c so that the visible area of all DOM nodes of a segment—a multi-polygon—best matches the original rectangle in terms of the area F_1 -score (cf. Figure 3.15 a). The recall of 0.79 shows that 1/5 of the rectangles’ area is not part of segments, which is sensible as (1) annotators tended to draw rectangles that are a bit larger than necessary for speed, and (2) a multi-polygon naturally provides a tighter fit to DOM nodes than one rectangle. The precision, on the other hand, is very high (0.94), indicating only a few cases of rectangles drawn a bit too small. By adding DOM nodes that were nearly contained in the rectangles, however, the number of empty segments, which contain no DOM node and would thus be discarded, drops from 7% to just 2%. Figure 3.15 b shows the distribution behind these averages: most multi-polygons match the drawn rectangle indeed accurately.

Annotation quality assessment Table 3.8 shows the annotation quality in terms of the agreement measure we developed previously. Annotators largely agree which text nodes belong together, as indicated by the very high F_{B^3} (0.78) for *chars*. Indeed, the rather large difference between *pix-*

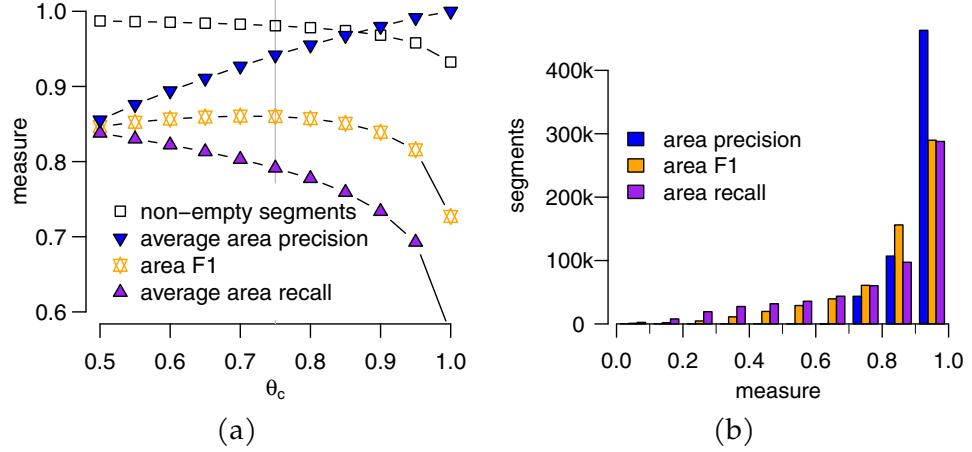


FIGURE 3.15: (a) Overlap of the annotated segments and DOM-based multipolygons as well as fraction of non-empty segments at different thresholds θ_c , and (b) histograms of segments by the measures at the chosen θ_c of 0.75.

els (F_{B^3} of 0.65) and edges (both 0.73) shows that a significant portion of disagreement is due to a different segmentation of blank space (i.e., background), which is irrelevant for most downstream applications. Moreover, as a comparison of the values using $\max(P_{B^3}, R_{B^3})$ highlights, nearly all disagreement is due to annotators working at different levels of granularity, and not because of vastly different segmentations. We thus conclude that our dataset presents a high-quality resource for web page segmentation, and could even be extended to provide a hierarchical ground truth segmentation in the future.

Segmentation Fusion

In order to fully utilize the wisdom of the crowd as well as to allow for easier evaluation of segmentation algorithms and training of learning-based ones, we fuse the five segmentations per web page into a single coherent ground truth as described above. We use *pixels* as the atomic elements to be in line with the annotation as it is based on screenshots.

Figure 3.16 shows that fusing only elements that the majority of annotators put together (threshold of 3) reduces the number of *pixels* by 20%, but much less so the *edges* (6-7%), *nodes* (5%), and especially *chars* (2%). The annotators thus largely agreed on which elements are in a segment. The larger reduction for *pixels* is due to few annotators working at a more coarse level, for which segments naturally contain more blank space.

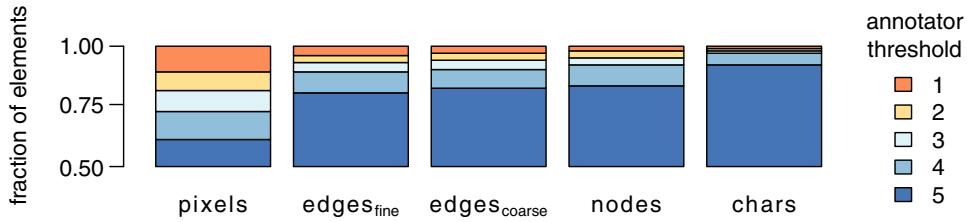


FIGURE 3.16: Fraction of atomic elements that are part of the ground truth for different annotator thresholds.

Figure 3.17 a and b exemplify the fusion. Figure 3.17 c compares the number of segments before and after the fusion for various θ_s . Roughly speaking, for $\theta_s = 0.9$ elements are put together in one segment if all annotators did so, whereas for $\theta_s = 0.1$ they are put together if any annotator did so. As is desirable, the figure shows that the distribution of the employed majority voting ($\theta_s = 0.5$) is also very similar to the original averaged distribution.

3.3.5 Algorithm Comparison Experiment Setup

Our empirical comparison of web page segmentation algorithms is based on the Webis-WebSeg-20. Importantly, the dataset contains a web archive file for each web page, which allows to re-render the web page as if viewed at the time of the archiving. For algorithms that need no complete re-rendering, the dataset also provides for each page the DOM HTML, a screenshot, and the list of DOM nodes mapped to their coordinates on the screenshot. The latter allows to convert between segment descriptions as screenshot coordinates and as sets of DOM nodes. As the ground-truth uses a flat segmentation but some algorithms produce hierarchical segmentations, we flatten such hierarchical segmentations for the evaluation.

In order to provide results that are applicable for various downstream tasks of web page segmentation, we execute all experiments for each of the five types of atomic elements defined in Section 3.3.3. Different downstream tasks of web page segmentation weigh certain errors differently. For example, although for most downstream tasks it does not matter how background space is segmented, it is important for tasks that consider the spacing between segments, like design mining. P_{B^3} and R_{B^3} can be adapted to a downstream task by calculating them specifically for the type of elements of the web page that is relevant for that task.

The following describes the segmentation algorithms that are compared in our experiments, and reports on the results of a corresponding parameter tuning for the algorithms. Table 3.9 gives an overview of the algorithms

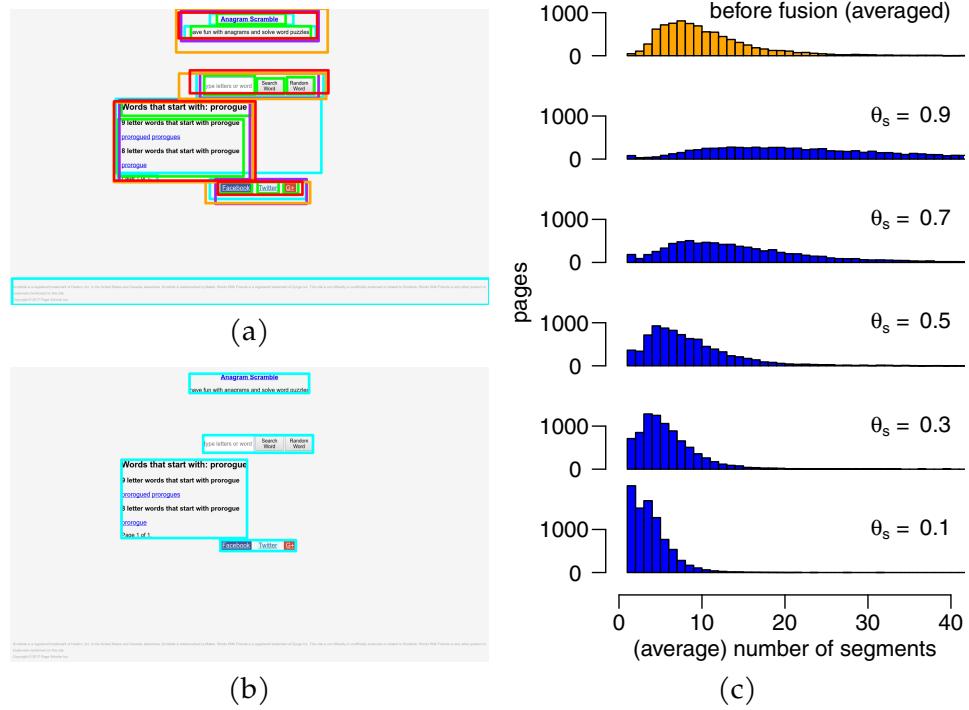


FIGURE 3.17: Segmentations for the web page in Figure 3.12a, (a) by the five annotators (one color each), and (b) after fusion with $\theta_s = 0.5$. (c) Number of pages by the number of segments before and after fusion for various values of θ_s .

in the experiments, which are chosen as representatives for different segmentation paradigms and tasks. For web page segmentation, we evaluate the classic DOM-based VIPS, the specifically heading-based HEPS, and the purely visual algorithm of Cormier et al. Inspired by the impressive recent advances in image understanding, we also evaluate the performance of one state-of-the-art algorithm of this field for the task of web page segmentation: MMDetection. Furthermore, as the tasks of web page segmentation is conceptually similar to the task of print document segmentation, we also evaluate the performance of a state-of-the-art approach for that task, the neural network of Meier et al.. Moreover, we report results for a voting-based ensemble of the algorithms. To contextualize the results, we include a naive baseline for comparison. We found that the algorithms fail for a few web pages (at most 0.2% per algorithm), for example, due to a web page's own JavaScript code interfering with the JavaScript code of the segmentation algorithm. We use the baseline's segmentation in this case as a fallback.

TABLE 3.9: Overview of the five compared segmentation algorithms with respect to the kind of input documents they were created for, the features they use, and the format of the output segmentation.

Name	Ref.	Document	Features	Output
VIPS	[43]	Web page	Tree, style, location	Rectangle tree
HEPS	[167]	Web page	Tree, style	Node set
Cormier et al.	[60]	Web page	Screenshot	Rectangle tree
MMDetection	[51]	Photo	Screenshot	Pixel masks
Meier et al.	[171]	Article page	Screenshot, text-mask	Mask

We provide all code for the evaluation in the a repository, and all generated segmentations as a new data resource.²⁹

VIPS The “VIision-based Page Segmentation algorithm” [43] is the de-facto standard for web page segmentation. Starting from one segment that covers the entire page, VIPS creates a hierarchical tree of segments based on the DOM tree of a web page. The rectangular segments are split based on their so-called degree of coherence, which is computed through heuristic rules based on the tag names, background colors, and sizes of DOM nodes, as well as visual separators: segments are split if their so-called degree of coherence is less than the permitted degree of coherence (PDoC), which is the single parameter of the algorithm. Previous implementations of VIPS rely on web rendering frameworks that are no longer maintained and render modern pages incorrectly. We thus ported one implementation³⁰ to JavaScript so that every modern browser can run it. For the experiments, we then used the reproduction mode of the Webis-Web-Archiver (cf. Section 2.2) to have a Chrome browser run VIPS on the web pages as they are re-rendered from the web archives. Though Cai et al. [43] described the degree of coherence to range from 0 to 1, the implementation we ported and thus ours alike use an integer range from 1 to 11, since the heuristic rules suggest the corresponding 11 thresholds. The VIPS algorithm failed for 14 web pages (0.2%) due to rendering errors or due to interference of the web page’s and VIPS’ JavaScript code.

In a 10-fold cross-validation, the optimal value for PDoC was consistently 6. Figure 3.18 shows the average number of segments and performance for all values from 1 to 11 over all web pages. As the top graph

²⁹Code, documentation: <https://github.com/webis-de/ecir21-an-empirical-comparison-of-web-page-segmentation-algorithms>

Provenance data: <https://doi.org/10.5281/zenodo.4146889>

³⁰Our port of https://github.com/tipopela/vips_java is included in our code repository.

shows, the number of segments stays almost the same for PDoC from 1 to 6, but increases considerably beyond that. The graphs are very similar for all types of atomic elements, with the notable exception of P_{B^3} —and thus also $F_{B^3}^*$ —for *pixels*, which is considerably worse. We discuss this observation below. Compared to the default value for PDoC of 8 for the original implementation, $F_{B^3}^*$ increases by up to 0.20, which highlights the importance of parameter tuning.

HEPS The “HEading-based Page Segmentation algorithm” [167] uses heading detection to identify segments. The authors define a heading as both visually prominent and describing the topic of a segment. HEPS does not solely rely on the HTML heading tags, as the authors found that headings are frequently defined by other means, and that heading tags are frequently used for other purposes. Instead, HEPS identifies headings and their corresponding segments through heuristic rules based on their position in the DOM tree, tag name, font size, and weight. The algorithm first identifies candidate headings using text nodes and images, and after that their corresponding blocks. It then creates a hierarchical segmentation based on the identified blocks. We use the original JavaScript implementation by the authors of the algorithm³¹ in the same manner as our reimplementation of VIPS. For consistency with the other algorithms in this comparison, we merge the extracted headings with their associated segments. The HEPS algorithm originally failed for 211 web pages (2.5%) due to rendering errors or due to interference of the web page’s and HEPS’ JavaScript code, but we were able to reduce this amount to just 5 web pages (0.06%) through slight changes in handling of arrays in the code.

Cormier et al. Cormier et al. implement a purely visual algorithm to web page segmentation that uses edge detection to find semantically significant edges, used to synthesize a coherent segmentation [60]. The algorithm takes a screenshot of the web page as input, and therefore does not require to re-render the page. It first calculates for each pixel the probability of a “locally significant edge,” which is based on how different the horizontal or vertical image gradients at the pixel are from those of the surrounding pixels. After that, the algorithm composes horizontal and vertical line segments from these edge pixels, up to a maximum length of t_l . Note that the larger t_l , the larger the “gap” that visual edges can have to still be considered one line segment. The algorithm then starts with the entire page as

³¹<https://github.com/tmanabe/HEPS>

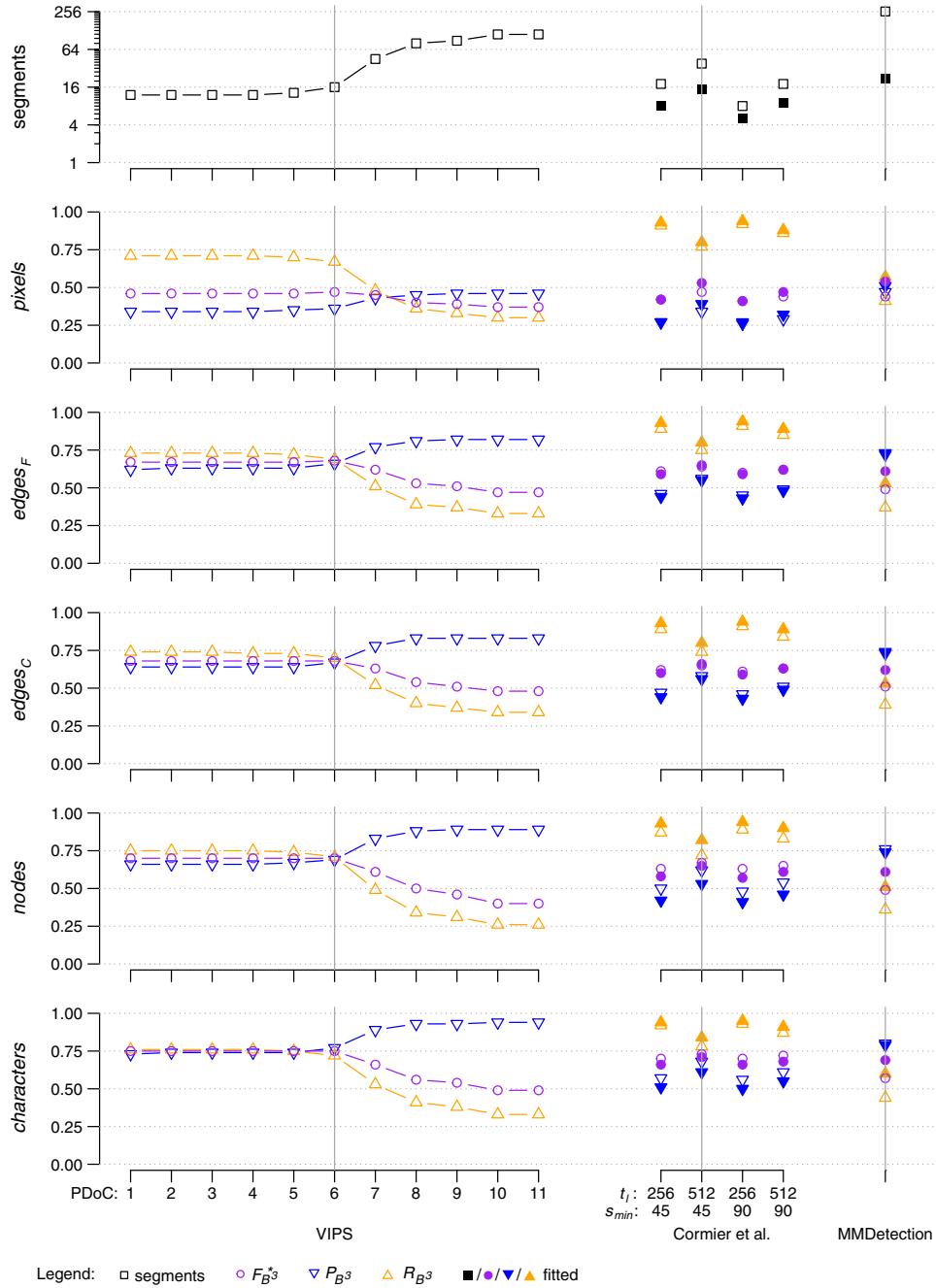


FIGURE 3.18: Number of segments (top plot), $F_{B^3}^*$, P_{B^3} , and R_{B^3} for different parameters for the algorithms of VIPS, Cormier et al., and MMDetection. Filled symbols correspond to the values after fitting the segmentation to DOM nodes. The vertical lines show the overall best-performing parameter setting for each algorithm after fitting, as measured by $F_{B^3}^*$.

one segment, and recursively splits the segments into two by choosing the vertical or horizontal line that is the most “semantically significant,” i.e., that has the most and clearest edge pixels. The algorithm stops if there are no semantically significant lines in a segment, or if a split would result in a segment with one side being less than s_{\min} long. The authors thankfully provided us with their implementation for our experiments. The algorithm is computationally expensive, and requires up to 1 hour for the larger web pages of the dataset on a modern CPU, but could likely be sped up considerably through the use of multi-threading and GPUs.

Due to the runtime requirements of the current implementation, we only tested four parameter settings that the original authors suggested to us: each combination of $t_l \in 256, 512$ and $s_{\min} \in 45, 90$. The algorithm contains another parameter t_p that is used as a threshold for determining semantically significant line segments, but we always use $t_p = 0.5$ as suggested by the authors. Figure 3.18 shows the average number of segments and performance over all web pages. For a fair comparison, we fit the visual segmentations to DOM nodes as in Section 3.3.4, which has for most cases just a minor effect on the performance, though it does increase $F_{B^3}^*$ for the best parameter setting ($t_l = 512, s_{\min} = 45$) for *pixels* by 0.06. This setting is used in our further experiments.

MMDetection The Hybrid Task Cascade models [52] from the MMDetection toolbox [51] jointly segment real-world images (photos) and detect objects in them. At the time of our experiments, this algorithm led the MSCOCO [160] detection task leaderboard³² and can thus be considered state-of-the-art for photo segmentation. The neural network model³³ features an intricate cascading structure. In spot checks, we found that the algorithm detected only segments within images that were included in the web pages. We found that this is due to a separate filtering step that classifies segments as containing real-world objects, so we disabled this step since its purpose does not exist in web page segmentation. Otherwise, the algorithm is the same as the original and no re-training is performed to investigate the similarities of photos and web pages. As segments can be arbitrarily formed in our evaluation setup, we use the corresponding instance segmentation output of the algorithm instead of the more coarse bounding boxes. Like for Cormier et al., we fit the resulting pixel mask segmentation to DOM nodes, which results in performance increases up to 0.12 in

³²<https://cocodataset.org/#detection-leaderboard>

³³We use the model with X-101-64x4d-FPN backbone and c3-c5 DCN as available and suggested at <https://github.com/open-mmlab/mmdetection/blob/master/configs/htc>

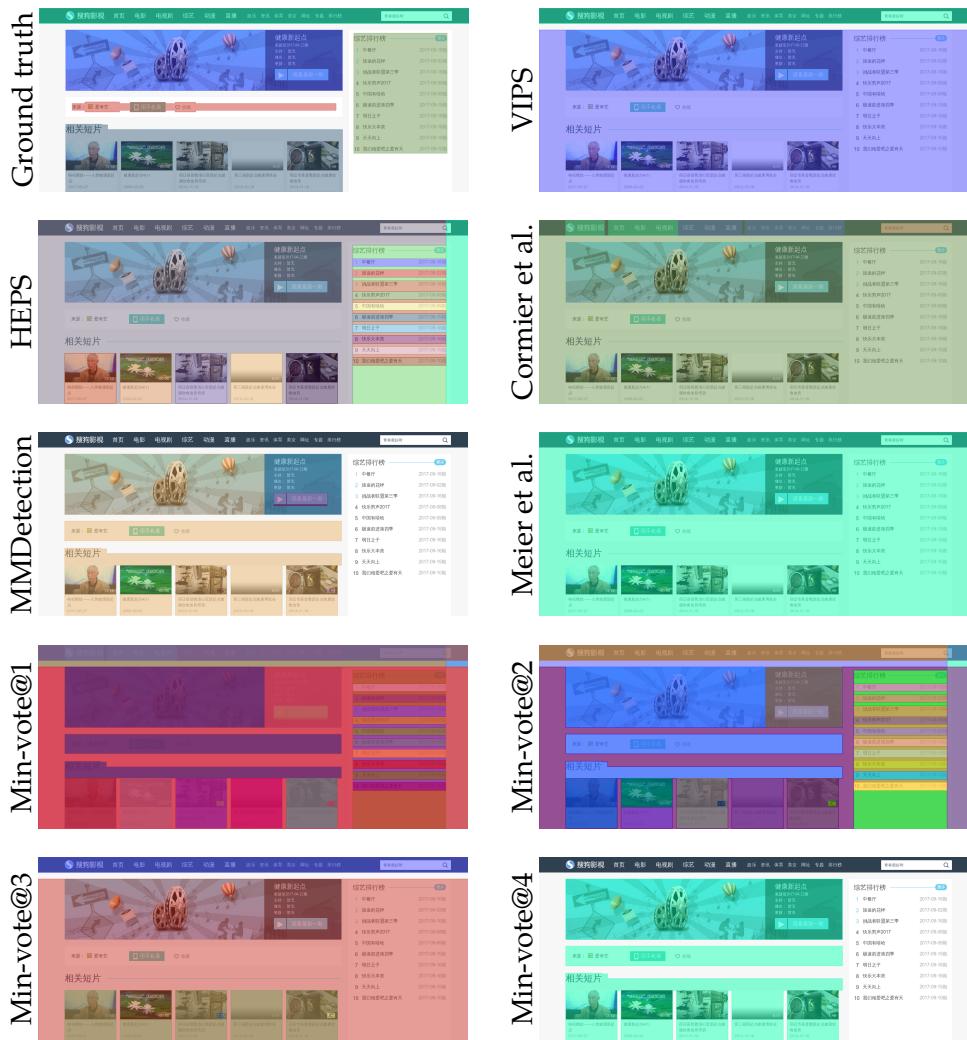


FIGURE 3.19: Segmentations of the top part of the same web page.

F_B^* ³⁴. MMDetection found no segments for 103 web pages (1%), which we treated like segmentations of one segment that contains the entire page.

Meier et al. The convolutional neural network by Meier et al. [171] is state-of-the-art in segmenting digitized newspaper pages. We reimplemented it in contact with the authors,³⁴ but instead of determining the position of text through optical character recognition (OCR) we use the positions of text nodes from the corresponding list of nodes that accompanies the Webis-WebSeg-20. As the algorithm requires the input to be always of the same

³⁴The authors reported an erratum in their publication to us, so we used the corrected kernel size of 3×3 instead of 5×5 for layers conv6-1 and conv7-1.

size, we had to crop or extend the web page screenshots to a uniform height. As a compromise between extremes, we selected a height that covers about 2/3 of pages, namely 4096 pixels. We then scaled the pages to 256x768 pixels to match the input width of the original approach. Since no pre-trained model is available, we use standard 10-fold cross-validation in the evaluation, and assure that all pages of a website are in the same fold. The training stopped when the loss did not improve for ten consecutive epochs, which led to a training of 20.8 epochs on average.

As the algorithm processes cropped web pages, its results are not fully comparable to those of the other algorithms. For this reason, we report the obtained measurements with some reservations and do not include the segmentations in the ensemble described below. The algorithm found no segments for 4 web pages (0.05%), which we treated like segmentations of one segment that contains the entire page.

Min-vote@n We also employ an ensemble of four of these algorithms, excluding the algorithm of Meier et al. as explained above. The ensemble algorithm is identical to the algorithm that was employed to fuse the human annotations to a single ground-truth in Section 3.3.4—just treating the algorithms as annotators. To filter out noise, the algorithm first removes all elements from consideration which less than n algorithms placed into segments. After that, the algorithm performs standard classic hierarchical agglomerative clustering, with the similarity of two elements being the ratio of algorithms that placed the elements in the same segment. Like in Section 3.3.4, we use a similarity threshold of $\theta_s = \frac{n-0.5}{k}$, where $k = 4$ is the number of algorithms. The algorithm thus tends to put elements in one segment if at least n algorithms did so. We report results for all plausible values for n , namely 1 to 4.

Baseline To put the performance of the algorithms into perspective, we report results for the naive approach of segmenting a web page into one single segment. This approach reaches always the maximum recall of 1 at the cost of the lowest possible precision. Both VIPS and the algorithm of Cormier et al. use this segmentation as their starting point.

3.3.6 Results of the Comparison

Table 3.10 shows the performance of each of the algorithms detailed above on the Webis-WebSeg-20 dataset. The reported values all reflect the results after tuning the respective parameters of the algorithms.

TABLE 3.10: Average number of segments per web page and evaluation results for each discussed algorithm on the Webis-WebSeg-20 dataset (Baseline, VIPS, HEPS, Cormier et al., MMDetection, Meier et al., and the Min-vote@ n ensembles): average F_1 -score (F_{B^3}), precision (P_{B^3}), recall (R_{B^3}), as well as the harmonic mean of the averaged precision and recall ($F_{B^3}^*$) for each type of atomic elements. The ground truth contains 9.1 segments on average. The highest score in each row (excluding the baseline) is highlighted in bold. The results of Meier et al. are shown in gray as its evaluation is not fully comparable.

Measure	Baseline	VIPS	HEPS	Corm.	MMD.	Meier	MV@1	MV@2	MV@3	MV@4
Segments	1.0	16.1	36.1	15.3	23.0	4.6	6.5	18.7	36.5	69.5
<i>pixels</i>	F_{B^3}	0.24	0.38	0.33	0.36	0.42	0.32	0.30	0.39	0.30
	$F_{B^3}^*$	0.28	0.47	0.44	0.53	0.54	0.50	0.35	0.50	0.45
	P_{B^3}	0.16	0.36	0.36	0.39	0.51	0.48	0.22	0.38	0.60
	R_{B^3}	1.00	0.67	0.56	0.80	0.57	0.52	0.96	0.72	0.36
<i>edges_{fine}</i>	F_{B^3}	0.44	0.59	0.48	0.51	0.53	0.41	0.50	0.56	0.39
	$F_{B^3}^*$	0.49	0.68	0.58	0.65	0.61	0.55	0.56	0.66	0.49
	P_{B^3}	0.32	0.66	0.61	0.55	0.73	0.55	0.40	0.61	0.81
	R_{B^3}	1.00	0.69	0.55	0.80	0.53	0.55	0.96	0.71	0.36
<i>edges_{coarse}</i>	F_{B^3}	0.45	0.61	0.49	0.53	0.54	0.42	0.51	0.57	0.39
	$F_{B^3}^*$	0.49	0.68	0.59	0.66	0.62	0.56	0.56	0.67	0.50
	P_{B^3}	0.32	0.67	0.62	0.56	0.74	0.55	0.40	0.63	0.82
	R_{B^3}	1.00	0.70	0.56	0.80	0.53	0.57	0.96	0.72	0.36
<i>nodes</i>	F_{B^3}	0.42	0.63	0.43	0.52	0.52	0.44	0.49	0.54	0.34
	$F_{B^3}^*$	0.46	0.70	0.54	0.65	0.61	0.56	0.55	0.65	0.44
	P_{B^3}	0.30	0.69	0.63	0.53	0.74	0.52	0.38	0.64	0.85
	R_{B^3}	1.00	0.71	0.46	0.82	0.51	0.61	0.96	0.65	0.29
<i>chars</i>	F_{B^3}	0.52	0.67	0.50	0.61	0.61	0.50	0.59	0.62	0.40
	$F_{B^3}^*$	0.57	0.75	0.60	0.71	0.69	0.61	0.64	0.71	0.50
	P_{B^3}	0.39	0.77	0.73	0.61	0.79	0.59	0.48	0.72	0.90
	R_{B^3}	1.00	0.72	0.51	0.84	0.60	0.63	0.96	0.71	0.35

The single algorithms, excluding baseline, Meier et al., and the ensemble, generate between 15.3 and 36.1 segments on average. This difference can be explained by the algorithms working at different levels of granularity. If successful, using more segments should increase the P_{B^3} . However, this is not necessarily the case: Though HEPS is clearly working at a finer level of granularity than VIPS (cf. Table 3.10 and Figure 3.19), both algorithms perform similar in terms of P_{B^3} .

The highest $F_{B^3}^*$ scores are reached for *chars* and the smallest for *pixels*. This difference is likely due to web page segmentation algorithms being developed for information extraction purposes mainly, and thus mostly optimized for text. However, for applications like design mining, even the spac-

ing between elements needs to be segmented correctly. New algorithms will be required for such and similar downstream tasks.

Conversely, the results differ only marginally between $\text{edges}_{\text{fine}}$ and $\text{edges}_{\text{coarse}}$, despite the visually very different edge detection [131]. This result is very convenient for future evaluations, as it indicates that (1) the parametrization of the edge detector does not play a major role, and (2) it is sufficient to evaluate for one parametrization of the edge detector. We recommend to employ $\text{edges}_{\text{fine}}$ in the future, as it produces fewer segments that have no edges and which are thus not considered in the evaluation.

The best-performing algorithm from the literature for most types of atomic elements is the VIPS algorithm, reaching a $F_{B^3}^*$ of up to 0.75 and convincingly beating the baseline in all cases. It thus comes closest to human annotators—and also relatively close in terms of the average number of segments, which is 9.1 for the ground-truth. Moreover, for a higher value of PDoC it can reach a very high P_{B^3} of up to 0.94 for *chars* (cf. Figure 3.18), which is close to human agreement (cf. [131]). Therefore, PDoC can indeed be used to adjust the level of segmentation granularity. Nevertheless, P_{B^3} is considerably lower at the optimal value for PDoC, which suggests that VIPS can benefit from an adaptation of PDoC to the (part of the) web page at hand. Though VIPS performs similarly well for most types of atomic elements, its precision is rather low for *pixels*. This difference is likely due to background pixels on the left and right of the actual content of the web pages: whereas VIPS includes such pixels in the segments, the human annotators did not (cf. Figure 3.19 for one example).

However, both the algorithm by Cormier et al. and MMDetection reach a similar performance to VIPS in terms of $F_{B^3}^*$, which demonstrates the viability of purely visual approaches to web page segmentation. By comparison, the algorithm by Meier et al. fails to compete with the other algorithms, even though it had a clear advantage over the other algorithms by being trained on the data. Its poor performance might be due to the required adjustment of the input screenshots.

The results for the min-vote ensembles show that even a basic voting scheme can be employed to efficiently fuse the output of different algorithms. Remarkably, Min-vote@2 reaches a $F_{B^3}^*$ scores very similar to those of VIPS. Like PDoC for VIPS, the parameter n here fulfills the role of selecting the desired level of granularity. This is especially helpful as some algorithms, like HEPS, do not have such a parameter. The ensemble therefore allows to incorporate the HEPS heuristic (and others without such a parameter) and still to select a level of granularity.

A special ensemble is that of Min-vote@4, which puts elements in one segment if and only if all four single algorithms did so. We want to highlight that P_{B^3} is about 0.9 for all atomic element types except *pixels*, which indicates that most segments of these elements are indeed separated from others by at least one of the algorithms. However, *pixels* are an exception here, which shows a deficit that needs to be addressed by future algorithms.

3.3.7 Conclusion

This Section revisits the task of web page segmentation, filling gaps that hindered the evaluation of generic web page segmentation algorithms. Unlike previous research, our evaluation framework does not focus on one of the various downstream tasks of web page segmentation. Instead it accounts for the different downstream tasks through a unified similarity measure for web page segmentations—well-founded in clustering theory—which can be used for tasks that focus on visual, structural, or textual elements. In the context of preserving web pages, it might be best to use an average of all the corresponding measures. Moreover, we show how this measure can provide the basis for annotator agreement calculation, ground truth fusion, and segmentation quality assessment. This foundation is used to construct the Webis-WebSeg-20, a dataset that comprises 42,450 segmentations from human annotators for 8,490 pages from 4,824 sites. Our evaluation framework and this dataset allows for the first time to assess web page segmentation algorithms for different downstream tasks in a coherent fashion. We report on the first of such assessments.

As we contrast and discuss the results of our evaluation for each type of atomic page elements, it becomes clear that the classical VIPS algorithm is still the overall best option, unless the downstream task requires pixel-based segments. In that case, purely visual page segmentation performs better, whereas otherwise it is a close second to VIPS. MMDetection performed especially well for being designed and trained for photographic images. Interestingly, the state-of-the-art approaches for such images as well as for newspaper page segmentation both employ deep learning, while the approaches for web page segmentation rely mostly on hand-crafted heuristics and observations. We believe that this difference mainly stems from the fact that no large-scale datasets for web page segmentation have been available in the past. With the work at hand, we lay the foundation for the development of new approaches that may improve over the long-standing, yet heretofore unknown champion, VIPS.

After web page segmentation, many downstream applications require labels for segments. A possible extension of our dataset would thus add segment labels. In the spirit of compatibility with as many downstream tasks as possible, a promising choice of segment labels is the function they fulfill on the web page [50]. Since such function labels carry a specific meaning, system developers can match such labels to the task at hand and then pick the corresponding segments, as well as evaluate with the extended dataset which algorithm performs best for segments that have the respective function. In the context of preserving digital culture, a labeling of segment functions would especially allow for more targeted indexing of the segments, allowing for a better retrieval of relevant web pages. Moreover, such annotation would also enable new analyses, for example showing how the implementation of different segment functions has changed over time.

3.4 Summary

This chapter focused on harnessing web archives for preserving digital culture, addressing three challenges in particular. First, Section 3.1) contributed an operationalization of a web page’s reproduction quality from the user’s perspective, a dataset with respective annotations, and the first approaches for a corresponding automatic quality assessment. Such an assessment allows for immediate quality control of the archiving process, enabling web archivists to take measures while web pages with low quality scores are still online. Second, Section 3.2 contributed a first analysis of unexpected content in general-purpose web archives (ads-only pages, loading indicators, page-sized pop-ups, access-blocking CAPTCHAs, and dominant error messages). Unexpected content—estimated in the section to affect roughly 10% of web pages in general crawls—constitutes a challenge in web archiving that is complementary to the quality assessment in the focus of Section 3.1). Its use in immediate quality control, however, is similar. Third, Section 3.3 contributed an in-depth investigation into the task of web page segmentation, providing a perception-based conceptualization of web page segments that is agnostic of downstream uses. Therefore, this conceptualization aligns with the goal of preservation, which is similarly agnostic of use cases. An automated web page segmentation is helpful for the preservation of digital culture, as it allows to process and index the different segments of an archived web page separately, facilitating a better retrieval of the preserved content. On the other hand, segmentation approaches harness the rich features of web page renderings, which web archives provide.

4

Harnessing Web Archives to Critically Assess Information

As discussed in Chapter 1, the web is a major information source for a large part of the human population, but to assess the provided information in a critical manner is complicated, both manually and automatically. Usually, a critical assessment requires looking at the information’s context on the same web page and other pages on the web. Though this context is highly ephemeral—e.g., a news article might be changed without leaving traces—, web archives can preserve the context and thus allow for reproducible and comprehensible assessments. In computational research, web archives can, in a similar way, provide for stable benchmark datasets, which are essential for the development of assessment technologies. Moreover, employing several snapshots of a web page, the edit history of a piece of information is an additional resource for assessment. Especially for collaboratively managed information sources like Wikipedia, the rate at which information is changed, the recency of the last change, or patterns of contributors correcting each other, are all among the valuable features that the edit history can provide for a critical assessment.

However, several challenges remain to fully harness web archives for such an assessment. This chapter focuses on two such challenges, one employing a page-specific archive—Wikipedia’s edit history—and the other employing classical web archives. As the first selected challenge, it has been unclear how to harness the potential of an article’s edit history, in particular how changes are reverted and re-reverted, to assess the quality of each edit. Specifically, *how to employ edit histories to assess information?* Section 4.1

contributes a novel pattern-based approach to distinguish different types of benign edits from vandalism, tested for the largest collaboratively edited information page of the present, Wikipedia. Applying the approach then uncovers temporal statistical patterns of vandalism that can inform on-the-fly edit quality assessments. An analysis using the language-independent approach across Wikipedia editions further reveals differences in the patterns between countries. The second selected challenge is to detect content on the web that can have undesirable effects on society, especially by misleading or polarizing through advocating extreme political views. Web archives are here essential to provide benchmark datasets. Specifically, this chapter investigates the question of *how to identify extremist (hyperpartisan) content in news archives?* Section 4.2 presents a thorough investigation on the automatic assessment of the hyperpartisan-ness of news articles in web archives. If first details an own approach employing stylistic features, comparing it to the assessment of an article’s veracity, and then the results of an international competition on the task. The results suggest that a large-scale automatic identification of hyperpartisan news on the web is possible.

4.1 Identification of Vandalism on Wikipedia

Vandalism is one of Wikipedia’s most prominent problems. From the start, the freedom that anyone can edit any article on Wikipedia has attracted vandals who damage articles instead of improving them. While the freedom to edit is a cornerstone of Wikipedia’s success story, part of Wikipedia’s community is constantly embroiled in reviewing edits to spot and revert the damage done by vandals. Without this cleanup, the quality of Wikipedia’s articles would quickly deteriorate. However, Geiger and Halfaker [87] find that only a small portion of Wikipedia’s community takes charge of reviewing, so that the amount of edits per time period cannot be handled entirely manually in a timely manner. This is why tools are employed to streamline reviewing and to automate part of it, ranging from rule-based bots up to machine learning approaches capable of detecting more subtle vandalism. In fact, a shared task organized by Potthast et al. [195] resulted in plenty of approaches.

The rigorous enforcement of Wikipedia’s codes of conduct significantly raised the bar for newcomers: Halfaker et al. [101] report that reverts—especially those done by automatic vandalism detectors—severely affect user retention. This raised bar, in turn, is considered a contributing factor to Wikipedia’s ongoing decline in terms of active editors since 2007 as pointed

out by Suh et al. [227]. Halfaker et al. [102] pass the blame to overprotective editors and detectors, suggesting that new approaches are in need to make reverters more friendly, so that new users who blunder will not be alienated while still undoing the deliberate damage caused by vandals. It is striking, though, that despite the large body of work on Wikipedia edits, reverters, and vandalism in particular, to the best of our knowledge, no research has been carried out to uncover *why* anonymous editors damage Wikipedia until now. Our contributions in this respect are threefold:

1. Ex post facto vandalism detection. Based on Wikipedia’s archived edit history, we conduct the first systematic analysis of Wikipedia article revert graphs to identify vandalism and damaging edits *after they have been undone* as ground truth for our analysis (Section 4.1.2).
2. Historic editor geolocation. Based on the archived IP addresses of Wikipedia’s anonymous editors since 2002, we geolocate 77% of these in terms of country and time zone by cross-checking geolocation databases with Regional Internet Registry data (Section 4.1.3).
3. Spatio-temporal analysis. Combining the aforementioned results, we conduct the first in-depth spatio-temporal analysis of Wikipedia’s archive, revealing a strong dependence of vandalism on time of day, day of week, country, culture, and Wikipedia language (Section 4.1.4).

The most salient insight of our archive analysis is that the relative amount of vandalism is significantly higher during the working hours of a week day (excluding summer) than otherwise, e.g., varying between one in three and one in six edits for the United States. Peaks of vandalism can be observed when people start working in the morning and after typical break times, suggesting a strong connection between labor and vandalism. Besides shedding light on human behavior in general, our results inform vandalism prevention efforts. For reproducibility sake, we provide the software underlying our analysis open source.¹

4.1.1 Related Work

Wikipedia has become a frequent subject for research across computer science (surveyed by Medelyan et al. [170] and Okoli et al. [183]), and the social sciences (surveyed by Schroeder and Taylor [212]).

¹See <http://github.com/webis-de/ICWSM-17/>.

Edits and editors Wikipedia’s collaborative writing process is of particular interest for its unique scale and since Wikipedia’s community is almost entirely self-organized. Steiner [224] develops a tool that allows for monitoring the current editing traffic on Wikipedia, separating three sources of edits: anonymous editors, registered ones, and automatic bots. Today, a little more than 15% of the edits on Wikipedia are done by bots and another 26% by anonymous users, whereas the majority of 59% of edits originates from registered users. In this regard, the question of “Who writes Wikipedia?” has been intensely debated, and it has been frequently pointed out that the majority of manual edits originates from a core group of registered “elite” editors who make up for most of the contributions [139, 185, 198], considering both edit quantity and edit quality (i.e., longevity of edited words). This may explain why research focuses almost exclusively on the portion of edits originating from registered users. In this connection, identifying and automatically detecting edit types [65, 66], and which types of edits originate from groups of registered editors who assume certain social roles within Wikipedia’s community [18, 81, 147, 244, 249], has recently attracted attention. The latter include basic roles identified via access privileges, or so-called barnstars awarded to editors by the community for outstanding contributions with respect to certain criteria, as well as custom user role ontologies. Besides roles, others attempt to quantify the extent of gender bias found within edits and among editors [17, 240]. Furthermore, Kuznetsov [153] investigates the motivations of Wikipedia editors, and Halfaker et al. [101] and Halfaker et al. [102] study the cohort of registered newcomer editors and whether they remain active when faced with backlash from the community, anonymous users, or from automatic reviewing tools.

Yasseri et al. [253] analyze temporal patterns of Wikipedia edits, contrasting all language versions of Wikipedia. The authors identify four groups of languages that exert different distributions of edit frequencies throughout the day. A clear night-and-day curve can be observed for most Wikipedias. For languages spoken in different time zones, the authors model the edit ratio distributions as mixtures of a standard distribution derived from averaging all Wikipedias edited mostly from a single time zone. However, the results allow for no insights into the nature of vandalism. In fact, all of the aforementioned studies mention vandalism only in passing, or not at all.

Vandalism Although vandalism on Wikipedia has attracted considerable attention, too, surprisingly, there is hardly any work as to why or under what circumstances editors vandalize. Geiger and Ribes [88] trace the

steps that lead to the banning of an individual vandal as an example of Wikipedia’s distributed self-preservation process; Shachaf and Hara [214] focus on user who act as trolls; and Kumar et al. [151] identify registered users engaging in vandalism using behavioral features. Otherwise, most papers on Wikipedia vandalism propose automatic vandalism detection tools: Potthast et al. [194] first developed machine learning technology for this purpose, and many of the approaches in existence today have been developed or derived from the results of two shared tasks at PAN 2010 and PAN 2011 [193, 195]. One of the currently best-performing approaches is that of Adler et al. [3], combining many of the approaches submitted to the shared tasks. From reviewing the literature on vandalism detection, none of the authors analyzed behavioral aspects, with one notable exception: West et al. [245] exploit the spatio-temporal characteristics of Wikipedia edits to train a machine learning model for vandalism detection. Similar to our methodology, they identify vandalism in Wikipedia article histories, and they employ a geolocation database to map the IP addresses of anonymous editors to their place of origin. Unlike in this work, though, their analysis encompasses only a small fraction of reverted edits from Wikipedia’s history, since they rely only on edits reverted with the administrative rollback tool. Furthermore, their geolocation does not take into account that old IP addresses may not be reliably geolocated with newer geolocation databases. Furthermore, they stop short of analyzing the spatio-temporal patterns, whereas the small scale and the noisy geolocation might have thwarted such analyses. Still, the features proposed help a machine learning algorithm to pick up vandalism.

4.1.2 Mining “Vandalism”

This section defines vandalism edits in Wikipedia and details our approach at identifying such edits as a ground truth for our analysis in its archived history: we rely on *ex post facto* evidence, namely whether an edit has been reverted manually or automatically. We loosely follow the self-reflection steps outlined by Howison et al. [112] to ensure validity.

Operationalizing “Vandalism”

Wikipedia defines vandalism somewhat vaguely as “editing (or other behavior) deliberately intended to obstruct or defeat the project’s purpose, which is to create a free encyclopedia, in a variety of languages, present-

ing the sum of all human knowledge.”² For clarification, the definition provides examples such as the removal of encyclopedic content, change beyond recognition, or change without regard to policies (neutral point of view, verifiability, and no original research), but also juvenile acts like adding obscenities, crude humor, nonsense, or removing an article’s entire content (“blanking”). Excluded from being vandalism, “even if misguided, willfully against consensus, or disruptive, [is] any good-faith effort to improve the encyclopedia,” like edit wars, where two parties fight over which version of an article is better by repeatedly reverting the opposing party’s version. Perhaps the most decisive property an edit must fulfill to be vandalism is that of being done with malicious intent—which is also the most elusive one. For example, subtle changes to an article’s point of view according to one’s own agenda may seem perfectly legitimate editing to a non-expert. This renders operationalizing vandalism difficult, as one cannot even entirely trust human judgment.

Its vague definition notwithstanding, vandalism is a real problem and remedies are sorely needed. The literature has hence adopted the term for lack of a better one to describe the efforts at identifying edits that come close to the above definition, thereby aligning terminology with that of the Wikipedia community. Vandalism has basically been operationalized in three ways (listed in ascending order of scalability and descending order of accuracy): (1) Based on external, independent review for up to thousands of edits [195]. (2) Based on internal, dependent review by analyzing explicit comments left by community members undoing vandalism (e.g., Kittur et al. [140]; Tran and Christen [231]). However, comments are often missing or may be left as false accusations (e.g., in edit wars). (3) Based on article states by considering all full page reverts (e.g., Rzeszotarski and Kittur [204]). Taken alone, this approach has a strongly oversimplified view on vandalism. In this work, we operationalize vandalism based on Approaches 2 and 3 to allow for full scale analyses of Wikipedia’s history archive. Nevertheless, we go beyond these approaches by analyzing the revert graphs of archived Wikipedia article histories in order to filter revert patterns that suggest good intentions on the part of an editor.

Identifying Past Vandalism as Ground Truth

At Wikipedia, manual or automatic undoing of vandalism edits basically happens by reinstating the latest non-vandalized revision of an article,

²Wikipedia. Wikipedia: Vandalism.

<https://en.wikipedia.org/w/index.php?title=Wikipedia:Vandalism&oldid=1055692475>

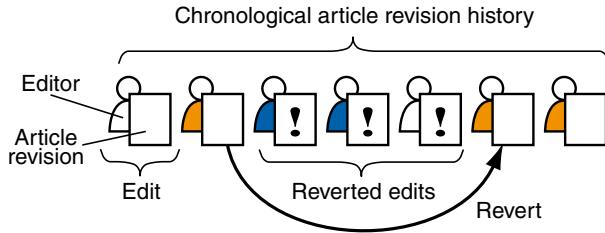


FIGURE 4.1: Illustration of a Wikipedia article revision history. Each revision is the result of an editor’s changes to its preceding revision, yielding a chronological sequence of revisions by successive editors. Shade indicates different editors, arc arrows indicate reverts, where an old article revision is reinserted as new revision, undoing all intermediate revisions.

which is directly supported from Wikipedia’s user interface. Edits that are undone this way are called *reverted*, whereas the undoing edit is called *revert* (see Figure 4.1). Reverted edits are not deleted; instead, a copy of the revision preceding the reverted edits is appended to the article’s revision history—a so-called *identity revert* or *full page revert*. We focus on full page reverts, as estimates suggest that *partial reverts* (i.e., edits that restore only some parts) cover only few cases of vandalism [83, 140].

As raw data for our analysis, we use the full page reverts from all Wikipedia article histories comprised in the May 2016 Wikipedia history dumps. The English Wikipedia history dump, 47 gigabyte compressed XML, contains 663,079,526 edits on 39,306,588 pages. We consider only the 471,070,114 edits on the 12,488,908 articles, disregarding user pages, discussion pages, etc. Due to article deletions, revision histories for many more than the 5.3 million articles that are currently online at Wikipedia are available. As a first step, we identify all full page reverts by matching the SHA-1 hashes of article wikitexts [140]: when a given SHA-1 value appears twice or more in a given article’s revision history, every reappearance after the first one constitutes a revert, and all edits between two appearances are reverted. The first row of Table 4.1 shows both the total number of reverts identified (44.9 million), and the resulting total number of reverted edits (119.7 million).

While reverted edits identified via SHA-1 matching have been used to train vandalism classifiers [232, 245], only a fraction of reverts (14.8%) *explicitly* undo vandalism, judging by the comments left by editors (when using Kittur et al. [140] approach to identify such comments). As it is not mandatory to indicate that vandalism is reverted, relying on comments alone severely underestimates the amount of vandalism on Wikipedia. However, presuming that all full page reverts are vandalism is a gross over-

TABLE 4.1: Step-by-step filtering of the English Wikipedia as per the revert patterns depicted in Figure 4.2. Counts of full page reverts and counts of reverted edits affected by corresponding full page reverts are given. Full page reverts are analyzed for indications of vandalism in edit comments as per Kittur et al. [140], and reverted edits are divided into edits originating from editors who are anonymous, registered, or bots. Reverted edits remaining after filtering are mostly vandalism, and almost unanimously damaging.

Revert filtering step	Full page reverts				Reverted edits					
	Vandalism as per Kittur		Total		Editor		Total			
	No	Yes	Absolute	Relative	Anonymous	Registered	Bot	Absolute	Relative	
Results of naive SHA-1 matching	38,244,710	6,670,575	44,915,285	100.0%	66,375,400	50,314,563	3,051,882	119,741,845	100.0%	
(a) reverts to page blank	-462,242	-4,085	-466,327	-1.0%	-19,176,154	-24,090,455	-1,629,953	-44,896,562	-37.5%	
(b) empty reverts due to renaming/removal/error	-2,085,189	-99,135	-2,184,324	-4.9%	0	0	0	0	0.0%	
Results after filtering pseudo-reverts	Σ	35,697,279	6,567,355	42,264,634	94.1%	47,199,246	26,224,108	1,421,929	74,845,283	62.5%
(c) self reverts	-3,865,372	-8,991	-3,874,363	-8.6%	-2,613,154	-2,018,218	-158,218	-4,789,590	-4.0%	
(d) revert corrections	-387,301	-62,437	-449,738	-1.0%	-862,439	-1,074,102	-27,456	-1,963,997	-1.6%	
(e) reverted reverts	-313,539	-13,133	-326,672	-0.7%	-2,719,594	4,088,871	-179,472	-6,987,937	-5.8%	
Results after filtering error-corrections	Σ	31,131,067	6,482,794	37,613,861	83.7%	41,004,059	19,042,917	1,056,783	61,103,759	51.0%
(f) interleaved reverts	-4,573,240	-401,277	-4,974,517	-11.1%	-3,606,705	-5,188,456	-295,504	-9,090,665	-7.6%	
(g) reverts reverting more than one editor	-1,776,317	-339,784	-2,116,101	-4.7%	-7,060,825	-4,860,432	-449,690	-12,370,947	-10.3%	
Results after filtering ambiguous reverts	Σ	24,781,510	5,741,733	30,523,243	68.0%	30,336,529	8,994,029	311,589	39,642,147	33.1%
(h1) reverts reverting registered editors or bots	-6,116,841	-799,928	-6,916,769	-15.4%	0	-8,994,029	-311,589	-9,305,618	-7.8%	
(h2) reverts reverting editors with IPv6 addresses	-213,963	-51,808	-265,771	-0.6%	-338,137	0	-338,137	-0.3%		
Results after all filtering steps	Σ	18,450,706	4,889,997	23,340,703	52.0%	29,998,392	0	29,998,392	25.1%	

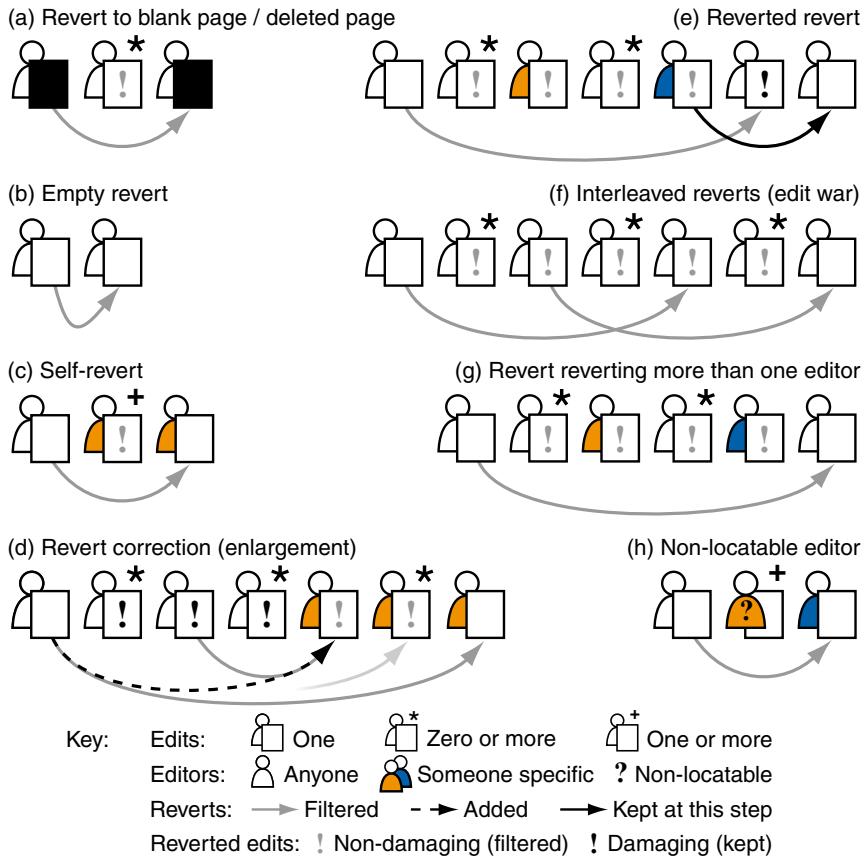


FIGURE 4.2: Revert patterns used for filtering full page revert graphs stepwise: first pseudo-reverts (a,b) are filtered, then error-corrections (c,d,e), ambiguous reverts (f,g), and finally reverts reverting edits of non-locatable editors (h). Each pattern depicts a regular expression that is matched against an article’s revision history, filtering or reinterpreting reverts accordingly.

estimation: in a manual analysis of the revert graphs of 100 randomly selected articles we found many reoccurring revert-patterns that seem natural for collaborative editing situations and that are completely harmless. In what follows, we define the harmless revert patterns we identified, and detail how filtering them from the set of full page reverts affects the amount of reverted edits that can be called vandalism with a high confidence.

Figure 4.2 lists the revert patterns used to filter reverts that are not suited to our analysis, as it is unlikely or unclear whether they indeed revert vandalism. The filter patterns (a) to (h) are organized in the order in which they are applied. Their order is important, since reverts may cross or include one another, resulting in intricate graphs that need to be carefully disentangled. Rows (a) to (h) in Table 4.1 show the amount of full page

reverts that are filtered by the corresponding pattern, and the amount of reverted edits that are hence disregarded. Related groups of patterns are detailed below.

Filtering pseudo-reverts Patterns (a) and (b) in Figure 4.2 depict pseudo-reverts, namely reverts that are not intended as such. Pattern (a) concerns removing all content from or deleting an entire article. This happens occasionally in an article’s history—sometimes as an act of vandalism—, so that pseudo-reverts may revert edits by many different editors (in Figure 4.2 denoted by a star as in a regular expression, where a white editor may be any editor). Pattern (b) captures reverts that do not change the article, which may occur when an article is renamed or due to MediaWiki errors. We filter these reverts first to not confuse other patterns; but note that reverts that undo page blanking remain untouched. As a result, about 2.7 million reverts are filtered, covering about 44.9 million unintentionally reverted edits (see Table 4.1, rows (a,b)).

Filtering error-corrections Patterns (c), (d), and (e) depict reverts which revert edits that are likely not vandalism. Pattern (c) concerns self-reverts where an editor fixes a mistake by undoing it again. Pattern (d) concerns series of reverts by the same editor, where the latest revert covers the previous reverts, implicating that the editor just corrected the revert. In such cases, we replace the series of reverts with a newly created one that corresponds to the editor’s actual intentions. In 9% of these cases, the editor enlarged her revert more than once in a row. Pattern (e) concerns longer reverts that are immediately reverted again, implying the editor of the first revert tried to damage an article by resetting it to a previous revision. We disregard the original revert and count only the one reverting the vandalism revert. Another 4.7 million reverts are filtered, covering about 13.7 million reverted edits (see Table 4.1, rows (c,d,e)).

Filtering ambiguous reverts Patterns (f) and (g) depict ambiguous reverts where it remains unclear which of the reverted edits originate from editors with damaging intent. Pattern (f) captures interleaved reverts as they usually appear in edit wars, since, as per Wikipedia’s definition of vandalism, edit wars do not necessarily happen among ill-intentioned editors. Interestingly, the majority of the about 1.5 million edit wars in the English Wikipedia have been rather short-lived, with 43% spanning only 2, and 37% spanning only 3 reverts. Pattern (g) concerns reverts that affect edits of different editors, suggesting a series of different vandalism cases. Here we

decided to err on the safe side and ignore reverts with such a strong claim. About 7.1 million reverts and about 21.5 million reverted edits are filtered (see Table 4.1, rows (f,g)).

Filtering non-locatable editors (h) Finally, in anticipation of the next step of geolocating the editors of vandalism edits, Pattern (h) filters reverts that revert edits whose editors cannot be geolocated. These are registered editors, who are responsible for less than 12% of reverted vandalism as per Kittur et al. [140], bots, and, due to lacking data in the geolocation databases, the very few editors using IPv6 addresses. About 7.2 million reverts and about 9.6 million reverted edits are filtered (see Table 4.1, rows (h1,h2)).

Altogether, 52% of all reverts, and about 75% of all reverted edits are filtered as harmless, ambiguous, or non-locatable. What remains as ground truth for our subsequent analysis are 29,998,392 edits, which represents vandalism originating from anonymous editors (see Table 4.1, last row).

Assessing the Vandalism Ground Truth

To assess our success at identifying vandalism, we perform a sanity-check on the filtered reverts and look at the effect filtering has on recall and precision. Similar to other collaboration scenarios, one would expect that most reverts affect very few edits, and few reverts affect many edits. To show that this is indeed the case, Figure 4.3a plots the number of reverts over the number of reverted edits on a double-logarithmic scale. For recall, we use the number of reverts whose comments indicate the removal of vandalism as per Kittur et al. [140] (see Table 4.1). Among all 44.9 million reverts, a total of 6,670,575 (14.9%) are vandalism reverts, which corroborates the results of Kittur et al. [140]. Applying Patterns (a)-(e) filters few reverts from this subset, where most are actually empty reverts. A large portion of reverts is filtered via Patterns (f) and (g) as ambiguous reverts: edit wars are not necessarily vandalism, and reverts reverting edits from multiple different users may include innocent editors. In these cases, we sacrifice some recall in favor of precision. Finally, 799,928 explicit vandalism reverts are filtered because they originate from registered users or bots, since they are not our focus of attention. In sum, our filtering recalls 73.3% of all explicit vandalism reverts. When disregarding those from registered users as well as pseudo-reverts, our approach recalls 84.7% of the remainder. As for precision, we manually review random samples of reverted edits by anonymous editors, 1,000 drawn before filtering and 1,000 after. In these cases, 68.7%

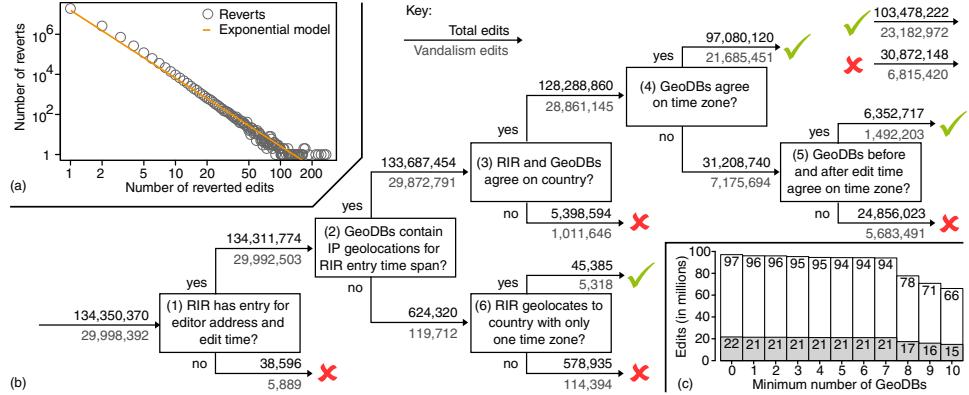


FIGURE 4.3: (a) Number of full page reverts with a specific number of edits they revert and fitted exponential model in a log-log plot. (b) Decision tree to decide whether to trust the available geolocation information for an edit (✓), or not (✗). The numbers denote the total edits and reverted edits for the English Wikipedia that went through each branch. (c) Number of total edits (white bars) and vandalism edits (gray bars) in millions from the yes-branch of Step (4) in (b) over the number of GeoDBs considered

of the reverted edits before filtering are indeed vandalism, whereas after filtering precision rises to a solid 82.8%.

These results show that identifying vandalism based on the ex post facto evidence is feasible with high precision and recall. Unlike using actual vandalism detectors, our approach incorporates the revert decisions of human editors and vandalism detection bots alike, including cases where editors err and revert their own edits. Moreover, our approach is language-independent and uses understandable rules. Of course, it cannot be used to automatically undo vandalism: rather, the reverted edits remaining after filtering form a ground truth of past vandalism on Wikipedia as per consensus of man and machine. These reverted edits, and their geolocation, serve as ground truth for our spatio-temporal analysis of anonymous vandalism on Wikipedia.

4.1.3 Geolocating Editors

For anonymous edits, the Wikipedia history dumps supply their server times and their editors' IP addresses at the time of editing. The server time, however, forecloses spatio-temporal analyses, since editors are typically far removed from the server. We hence resort to geolocation databases

(GeoDBs) to augment the dumps.³ But since IP addresses may change location over time, special care must be taken when dealing with historic IPs to ensure reliable geolocation. We show for the first time that even decade-old IP addresses can be reliably geolocated in terms of country and time zone by combining GeoDBs with Regional Internet Registry (RIR) data. Since 2003, RIRs supply daily updates of IP allocations to organizations, including their country. By combining eleven commercial GeoDBs from IPPligence and IP2Location with RIR data, cross-checking geolocations for consistency and removing all inconsistent ones, we obtain reliable geolocations for 77% of all anonymous edits on Wikipedia.⁴ The source code of our geolocation is freely available.⁵

To geolocate the editors' IP addresses, we identified relevant inconsistencies and devised a set of rules to deal with them. The resulting flow diagram of decisions is depicted in Figure 4.3b: (1) Removal of the few IP addresses that have no corresponding RIR entry.⁶ (2) Check whether the IP addresses are contained in one or more GeoDBs that fall within a "RIR span" (characterized by the time span between the RIR entry directly before and the RIR entry directly after the Wikipedia time of an edit). (3) If yes, removal of IP addresses where RIR span and GeoDBs disagree on their country. (4) If they agree on the country, check whether they agree on their time zone as well. For 94 million of the 97 million edits that pass this check (97%), at least 7 GeoDBs agree on the time zone (Figure 4.3c), making these geolocations very reliable. In case of time zone disagreements within the GeoDBs, (5) check whether the GeoDBs within the RIR span directly before and directly after an edit agree on time zones, and removal of all IP addresses where this is not the case. If yes, this corresponds to providers relocating an IP block within a multi-time-zone country, which is not recorded by RIRs. Going back to Step (2), when there is no GeoDB in the RIR span around an edit's time, (6) check whether RIR geolocates to countries that have only one time zone, and removal of IP addresses where this is not the case. This way, 103,478,222 of the 134,350,370 anonymous edits (77%) from the English Wikipedia can be reliably geolocated.

³Regarding GeoDBs, previous research suggests that country information is reliable (accuracy above 95%) and that latitude/longitude coordinates typically have a tolerance of far below 1,000km [192, 216].

⁴IPPligence Max from 2008 (Oct., Nov., Dec.), 2014 (Apr., Jul., Aug., Oct., Nov.), and 2015 (Feb., Apr.), <http://www.ippligence.com>; DB11 lite from 2016 (Jun.), <https://www.ip2location.com>; RIR data available at <http://ftp.RIR.net/pub/stats> where RIR is one of {afринic, apnic, arin, lacnic, ripe}.

⁵<https://github.com/webis-de/aitools4-aq-geolocation>

⁶A couple of old assignments are missing for historical reasons. APNIC. RIR Statistics Exchange Format. <https://perma.cc/SV84-LEBR>

TABLE 4.2: Historic geolocation success for all anonymous editors of the English Wikipedia in terms of edits and unique IP addresses whence they originated. Aside the totals, the subset of edits considered vandalism or damaging, and their corresponding IP addresses. Numbers are given for each exit node of the decision tree in Figure 4.3b, divided by whether or not the geolocation is trustworthy.

Decision Tree		Edits		Unique IP addresses	
Trusted	Exit step	Vandalism as defined here	Total	Vandal IPs	Total
<i>Entire Wikipedia:</i>		29,998,392 (22%)	134,350,370	11,990,674	34,993,205
No (✗)	Step (1)	5,889 (15%)	38,596	2,584	8,047
	Step (3)	1,011,646 (18%)	5,398,594	387,376	1,302,473
	Step (5)	5,683,491 (22%)	24,856,023	2,379,726	6,601,222
	Step (6)	114,394 (19%)	578,935	49,518	135,053
	Σ	6,815,420 (22%)	30,872,148	2,819,094	8,045,883
Yes (✓)	Step (4)	21,685,451 (22%)	97,080,120	8,586,646	25,453,545
	Step (5)	1,492,203 (23%)	6,352,717	635,490	1,712,340
	Step (6)	5,318 (11%)	45,385	2,558	12,572
	Σ	23,182,972 (22%)	103,478,222	9,224,625	27,178,053

In Steps (4) and (5), we determine the time zone based on the coordinates given by the GeoDBs using a time zone world map⁷ and cross-check it with the country stored in the GeoDBs. The GeoDB country and the time zone world map also sometimes disagree. Most of the time, the GeoDB country aligns with the GeoDB city, while coordinates may be off (compared to city coordinates in Wikipedia). To compensate for these inaccuracies, we take the nearest time zone to the coordinates within the GeoDB country as long as it is within 7.5° of the given coordinates (i.e., half the distance between meridians). A few cases with errors due to incorrect country codes (e.g., AS for Australia or RS for Serbia and Yugoslavia) or longitudes (e.g., Dar es Salaam being 39° East, not West) have been detected this way and manually fixed.

Table 4.2 shows the numbers of edits removed/kept as a result of filtering IP addresses with unreliable geolocation, and the numbers of unique IPs whence they originated. The latter decreases as expected, however, the ratio of reverted edits remains identical (22%), indicating that the geolocated edits form an unbiased sample. In sum, 23,182,972 reverted edits are subject to our subsequent analysis.

⁷Efele.net. tz_world Version 2016d. <https://perma.cc/P3MJ-DR9R>

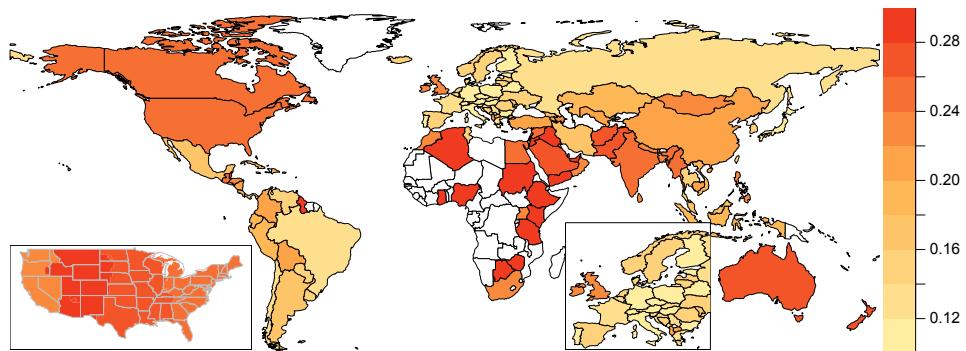


FIGURE 4.4: Ratio of vandalism to all edits in the English Wikipedia by country. Countries with less than 1,000 vandalism edits are not colored. The embedded small maps show (left) the vandalism ratio in the United States (without Alaska and Hawaii) by major time zone (from West to East: Pacific, Mountain, Central, and Eastern) with overlaid state borders and (right) Europe enlarged

4.1.4 Spatio-Temporal Analysis

To analyze spatio-temporal patterns, we calculate the ratio of vandalism edits (i.e., reverted edits as defined above) among Wikipedia edits per hour of the day and per location. The analysis is restricted to the anonymous edits that can be reliably geolocated, which includes most of the anonymous edits. Since we observed no correlation between being geolocated and being vandalism, we expect the restriction of reliable geolocation to not affect the results presented below. However, while our revert filter for vandalism detection is designed to avoid mislabeling a proper edit as vandalism, some cases of vandalism may have been missed. Still, almost all cases in which editors indicate by a comment that they are cleaning up vandalism take the form of full page reverts (also found by Kittur et al. [140]), so that it is unlikely that the vandalism ratio in anonymous edits or its spatio-temporal distribution are substantially different to what we observe. We estimate the vandalism ratio per hour of day (starting and ending at the full hour) by averaging over all days since January 1, 2006. Before 2006, in the early stages of Wikipedia, vandalism ratios are unstable and hence unreliable. About 4.3% of edits are discarded this way, but yielding an overall increase in effect sizes.

Our findings are based on visual inspection, backed by careful statistical analysis. We use Cohen’s d [57] to analyze the variances of the average vandalism ratios. While visibly different graphs usually correspond to significant differences due to sufficiently large sample sizes (millions of edits), high variances are a sign that the vandalism ratio is influenced by other

factors. To give an impression of when the vandalism ratio estimates are based on few edits, these estimates are toned down in the figures. Finally, we show the significance for all effects we analyze with Cohen's d using the Welch Two Sample t-test, with one to three asterisks (*) indicating p -values less or equal to 0.05, 0.01, and 0.001.

Figure 4.4 shows the vandalism ratio per country.⁸ The highest vandalism ratios are observed in Africa, possibly caused by difficulties with the English language, causing native English editors to consider edits from Africa vandalism more often. However, only 0.9% of the geolocated edits to the English Wikipedia come from Africa, so that we decided to leave an analysis of the reasons for future work. Both in Europe and in South America, the highest vandalism ratios are in the countries with English as the official language: Great Britain, Ireland, and Guyana, which suggests a correlation of main language and vandalism. A similar observation is made below, when we compare edits from a specific country to different language versions of Wikipedia.

Vandalism Ratios in the United States

Figure 4.5a reveals different vandalism ratio timelines for edits to the English Wikipedia from the United States, as well as the absolute number of edits and vandalism edits. While most edits are made between 14 and 17 hours, the ratio of vandalism to all edits peaks much earlier at around 9 hours with two more peaks occurring at 13 hours and at 19 hours. The lowest vandalism in both absolute and relative numbers occurs between 23 and 8 hours, which we refer to as night time for the purpose of this analysis. During the night, about one in six edits is vandalism, which changes dramatically to about one in three edits at peak times. This visually obvious difference in the vandalism ratio between night and day is reflected in the statistical analysis: the Cohen's d between the vandalism ratio averages for night and day shows a very strong statistical effect ($d = 14.7^{***}$). For reference, Figure 4.5a also plots the graph when only considering edits that are explicitly labeled as vandalism reverts by a corresponding editor comment as per Kittur et al. [140]. The two graphs resemble each other, further justifying our ex post facto vandalism detection.

The plots suggest that vandalism is connected to labor (working hours), with peaks of vandalism occurring when people start to work/study in the morning (8 to 9 hours) and after lunch (13 to 14 hours), e.g., as a way of

⁸The map uses GADM 2.8 country/state data, <https://perma.cc/K48Q-GFAZ>, and Efele 2016d timezone data, <https://perma.cc/P3MJ-DR9R>.

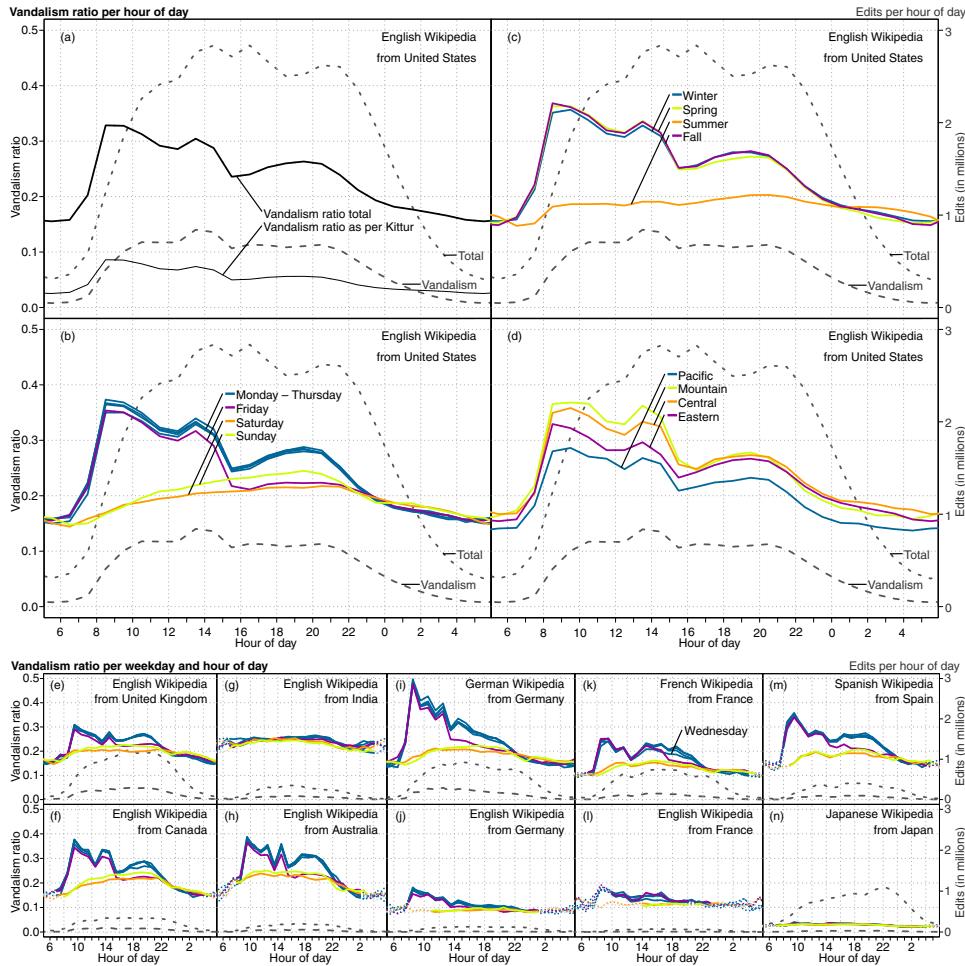


FIGURE 4.5: All plots show the *ratio of vandalism to all edits* per hour of day (left axis, solid lines), and for reference, the absolute number of edits and vandalism edits per hour of day (right axis, dashed lines), both averaged over Wikipedia's history. Plot (a) shows the overall ratio of vandalism edits on the English Wikipedia originating from the United States. Plots (b,c,d) divide the overall ratio by weekday, season, and US time zone. Plots (e-n) show vandalism ratios divided by weekday for the English Wikipedia edited from various countries, and for the German, French, Spanish, and Japanese Wikipedias when edited from Germany, France, Spain, and Japan. Ratios estimated from less than 1,000 vandalism edits are displayed with dotted lines

“fighting” stress or boredom. Running with the hypothesis of labor-related vandalism, the increase in the ratio of vandalism between 15 and about 20 hours may also be explained by people working long hours or by relieving stress after work. Alternatively, this increase may be due to an increased negativity over the course of the day, as Golder and Macy [91] found in their analysis of Twitter data. Further evidence for the labor-related vandalism hypothesis is provided by Figure 4.5b, which shows a clear difference in vandalism ratios between workdays and weekends: On workdays, the vandalism ratio is much higher than on Saturday and Sunday. On Fridays, however, the vandalism ratio graph is very similar to workdays up until about 16 hours, at which time it starts to resemble the graph of a weekend day (possibly an expression of “Thank God it’s Friday”). A statistical analysis shows a very strong effect between Monday to Friday and Saturday plus Sunday for 8 to 15 hours ($d = 1.49^{***}$), and a strong effect between Monday to Thursday and Friday to Sunday for 15 to 22 hours ($d = 0.88^{***}$). The increase of vandalism ratio on weekends has a medium effect, comparing the hour intervals ($d = 0.53^{***}$ for Saturday and $d = 0.68^{***}$ for Sunday). This small increase might again be related to the increase in negativity found by Golder and Macy [91]. As shown in Figure 4.5c, vandalism also reduces during summer. This could be due to people going on vacation or being generally more relaxed. However, the effect size between summer and the other months for the time between 8 and 22 hours is only small ($d = 0.34^{***}$), which is due to a large variance in the vandalism ratios from fall to spring. Although we have formed a number of hypotheses that may explain the variance, a thorough investigation requires correlating vandalism with other variables of interest and is hence left to future work.

We also investigated regional influences, partitioning the United States from west to east and analyzing vandalism ratio differences for each of the four parts corresponding to the well-known time zones Pacific, Mountain, Central, and Eastern.⁹ The different vandalism ratios for these time zones are visible in Figure 4.4 (bottom left), and on a per hour basis in Figure 4.5d. The graphs look very similar, the difference being only the overall vandalism ratio. While the differences between the four time zones are all significant, they seem minor compared to other influences; even between 8 and 15 hours the effect sizes are small ($d < 0.30$).

⁹While the geolocation detailed above uses the fine-grained IANA time zones, huge size differences in terms of area covered render them less suited here (e.g., Indiana alone has 8 time zones).

Vandalism Ratios across Countries

By repeating the above analyses for different countries and Wikipedia languages, we find large differences but also commonalities. For the sake of brevity, we only report the most interesting results when considering the weekday as an additional variable.¹⁰ Figures 4.5e-h show the vandalism ratio for the countries with the second to fifth-most edits to the English Wikipedia. Similar to the US, a difference between workdays and the weekend is clearly visible for the United Kingdom, Canada, and Australia, with the corresponding effect sizes d being 0.98***, 1.15***, and 0.87*** for 8 to 15 hours, and 0.81***, 0.61***, and 0.74*** for 15 to 22 hours. The effect is much smaller for India (0.12** for 8 to 15 hours and 0.32*** for 15 to 22 hours) with the possible cause that Indian citizens put much less emphasis on labor vs. leisure than people in Western countries. Also, many Western countries outsource to India, leading many to adapt their working habits accordingly, possibly further smoothing the graph.

Since Figure 4.4 suggests that the vandalism ratio to the English Wikipedia is higher in countries with English as the official language, we also analyze whether countries with edits in more than one Wikipedia variant have different vandalism ratios. Figures 4.5i-l compare the edits from Germany or France to the English Wikipedia with those to the respective “home” Wikipedias: The vandalism ratios are indeed higher in the “home” Wikipedias, especially for Germany where the English vandalism ratio is below 0.2 instead of reaching a striking 0.5 at 8 hours in the German Wikipedia (the highest ratio we observe in our analysis). A possible explanation could be that people with different background and different susceptibility to vandalism edit the different variants (i.e., the English Wikipedia may attract more educated people in non-English countries). However, despite these differences in magnitude, the graphs in Figure 4.5i,j as well as in Figure 4.5k,l still bear resemblance as to where peaks and valleys lie. Also, the relatively low vandalism ratio for Wednesday afternoons for edits from France—likely caused by the school-free afternoon at that day—is visible in the France-plots of both the French and the English Wikipedia ($d = 0.28***$ and $d = 0.17***$). We therefore see it as more likely that people just tend to vandalize the Wikipedia variant of their mother tongue more frequently as it is an easier target (e.g., usually ranked higher by search engines) and altogether follow a similar rhythm of life with vandalism ratios peaking when starting/continuing work/studies. Finally, Figures 4.5m,n show the van-

¹⁰The full tables and plots for the other countries and Wikipedias are available at <http://github.com/webis-de/ICWSM-17/>.

dalism ratio for two more Wikipedias among the top 7 with the most edits: Spanish and Japanese. While the Spanish plot follows a pattern similar to the one in the US, the vandalism ratio in Japan is really low (3% on average), with the only statistical effect being a higher vandalism rate during the day than during night (not visible in the plot but still a medium effect of $d = 0.54^{***}$ due to the low variance). Thus, while our analysis shows that time has statistically strong effects on the vandalism ratio, the example of Japan shows that cultural differences can have an even stronger effect.

4.1.5 Conclusion

Our study of reverted anonymous edits in Wikipedia’s archive reveals strong spatio-temporal effects that are apparently related to labor. At typical work/study starting/continuing hours, a larger portion of edits than otherwise are vandalism—with a remarkable peak of about 50% around 8 hours from Germans to the German Wikipedia. During weekends and vacation, the ratio of reverted edits is substantially lower than on working days. This suggests that vandalism helps people to relief from stress when starting to work, to fight boredom, or to show off in front colleagues/fellow students. In conclusion, a better understanding of vandalism and when it happens is a first step towards gaining a better grasp of the problem’s underlying causes and to answer the question “Why are people vandalizing Wikipedia?” While the term “vandalism” usually implies destruction without reason, a significant portion of vandalism on Wikipedia may not happen without reason, and the vandals may therefore be open to some form or another of nudging into the right direction. Our observations can initiate the development of smart technologies that monitor, detect, predict, and prevent potential threats to online communities or social software due to periodically increased susceptibilities to destructive behavior. For example, raising awareness at the right time may help users redeem themselves before acting out.

Future work should deepen the archive analyses and correlate vandalism to other variables of interest that may influence people’s behavior, such as the weather, global events, age (e.g., adults vs. pupils), urban vs. country life, or even political orientation. Closer to hand, our ex post facto evidence-based vandalism detection provides for a reliable way of generating training data with little noise that may be used to train vandalism detectors at scale. But future work should also broaden the scope to other social softwares. Our geolocation approach can be immediately applied in other scenarios where IP addresses are recorded and become part of a public archive. For

example, on discussion forums, behavioral differences may be observed dependent on when and from where someone participates in a thread. Altogether, managing anti-social behavior online has become an increasingly important task for social media platforms, and handling it well depends on a thorough understanding of its causes. For Wikipedia, the causes for vandalism are mostly unknown, whereas our work hints at some of them.

4.2 Identification of Hyperpartisan News

Yellow journalism has established itself in social media, nowadays often linked to phenomena like clickbait, fake news, and hyperpartisan news. Clickbait has been its first “success story”: When the viral spreading of pieces of information was first observed in social networks, some investigated how to manufacture such events for profit. Unlike for “natural” viral content, however, readers had to be directed to a web page containing the to-be-spread information alongside paid-for advertising, so that only teasers and not the information itself could be shared. Then, to maximize their virality, data-driven optimization revealed that teaser messages which induce curiosity, or any other kind of strong emotion, spread best. The many forms of such teasers that have emerged since are collectively called clickbait. New publishing houses arose around viral content, which brought clickbait into the mainstream. Traditional news publishers, struggling for their share of the attention market that is a social network, adopted clickbait into their toolbox, too, despite its violation of journalistic codes of ethics.

The content spread using clickbait used to be mostly harmless trivia—entertainment and distraction to some, and spam to others—but in the wake of the 2016 United States presidential election, “fake news” came to widespread public attention. While certainly not a new phenomenon in yellow journalism, its viral success on social media was a surprise to many. Part of this success was then attributed to so-called hyperpartisan news publishers [33], which report strongly in favor of one political position and in fierce disagreement with its opponents. Clinging to hyperpartisanship often entails stretching the truth, if not breaking it with fake news, whose highly emotional content makes them spread exceptionally fast, like clickbait.

The fast spread of hyperpartisan news, however, causes also a fast cycle of publication, re-publication, and content editing, that complicates the tracing of information after the fact. Therefore, web archives gained attention as a tool for provenance also among journalists—with the additional effect of not supporting linked hyperpartisan news publishers through ad

revenue and search engine optimization.¹¹ The work at hand focuses on the provenance aspect of web archives only.

We define hyperpartisan news detection as follows:

Given the text and markup of an online news article, decide whether the article is hyperpartisan or not.

Hypertpartisan articles mimic the form of regular news articles, but are one-sided in the sense that opposing views are either ignored or fiercely attacked. We deliberately disregard the distinction between left and right, since previous work has found that, in hyperpartisan form, both are more similar to each other in terms of style than either are to the mainstream. The challenge of this task is to unveil the mimicking and to detect the hyperpartisan language, which may be distinguishable from regular news at the levels of style, syntax, semantics, and pragmatics.

This section reports on two experiment series on the detection of hyperpartisan news. For both series we harnessed web archives to allow for reproducible setups: though we employ custom wrappers to provide that datasets in widespread formats for easier processing, the stored web archives allow to extend these formats later on as necessary. The first series contrasts style-based hyperpartisan and fake news detection, detailing (1) a large news dataset annotated by experts with respect to veracity and hyperpartisanship (Section 4.2.2), and (2) extensive experiments on discriminating fake news, hyperpartisan news, and satire based solely on writing style (Section 4.2.3). The second series reports on the conduction of an international shared task on the topic, detailing (1) the new dataset employed in the task, consisting of 1273 manually labeled articles and 754,000 distantly labeled articles (Section 4.2.4); (2) an overview of the approaches submitted and features employed by the 42 teams who submitted a valid run (Section 4.2.5); and (3) a discussion of the results achieved (accuracy of 0.822 on a balanced sample), including an ensemble that further increases the accuracy by 0.048 (Section 4.2.6).

4.2.1 Related Work

Though the task of hyperpartisan news detection is new, already much research has been published on the somewhat related task on fake news detection. Connections have been drawn especially due to observed correlations of news being hyperpartisan and fake [218]. We expect that approaches to hyperpartisan news will fall into the same three categories as

¹¹<https://iffy.news/wayback/>

for fake news detection and employ similar features (Figure 4.6): they can be knowledge-based (by relating to known facts), context-based (by analyzing news spread in social media), and style-based (by analyzing writing style, though we expect that also presentation style, as one could analyze through web archives (cf. Section 2.2), could provide for effective features).

Knowledge-based detection Methods from information retrieval have been proposed early on to determine the veracity of web documents. For example, [76] propose to identify inconsistencies by matching claims extracted from the web with those of a document in question. Similarly, [166] measure the frequency of documents that support a claim. Both approaches face the challenges of web data credibility, namely expertise, trustworthiness, quality, and reliability [90].

Other approaches rely on knowledge bases, including the semantic web and linked open data. [247] “perturb” a claim in question to query knowledge bases, using the result variations as indicator of the support a knowledge base offers for the claim. [55] use the shortest path between concepts in a knowledge graph, whereas [217] use a link prediction algorithm. However, these approaches are unsuited for new claims without corresponding entries in a knowledge base, whereas knowledge bases can be manipulated [105].

Context-based detection Here, fake news items are identified via meta information and spread patterns. For example, [161] show that author information can be a useful feature for fake news detection, and [69] attempt to determine the veracity of a claim based on the conversation it sparks on Twitter as one of the RumourEval tasks. The Facebook analysis of [173] shows that unsubstantiated claims spread as widely as well-established ones, and that user groups predisposed to conspiracy theories are more open to sharing the former. Similarly, [1], [154], [164], and [237] model the spread of (mis-)information, while [41] and [178] propose algorithms to limit its spread. The efficacy of countermeasures like debunking sites is studied by [228]. While achieving good results, context-based approaches suffer from working only *a posteriori*, requiring large amounts of data, and disregarding the actual news content.

Style-based detection Deception detection originates from forensic linguistics and builds on the Undeutsch hypothesis—a result from forensic psychology which asserts that memories of real-life, self-experienced events differ in content and quality from imagined events [234]. The hypothesis

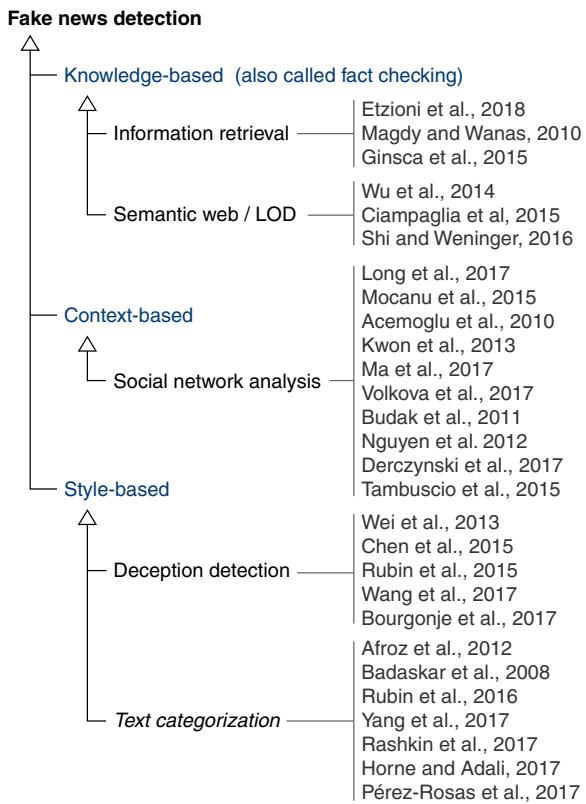


FIGURE 4.6: Taxonomy of paradigms for fake news detection alongside a selection of related work. The work at hand presents text categorization approaches.

led to the development of forensic tools to assess testimonies at the statement level. Some approaches operationalize deception detection at scale to detect uncertainty in social media posts, for example [242] and [54]. In this regard, [201] use rhetorical structure theory as a measure of story coherence and as an indicator for fake news. Recently, [241] collected a large dataset consisting of sentence-length statements along their veracity from the fact-checking site PolitiFact.com, and then used style features to detect false statements. A related task is stance detection, where the goal is to detect the relation between a claim about an article, and the article itself [38]. Most prominently, stance detection was the task of the Fake News Challenge¹² which ran in 2017 and received 50 submissions, albeit hardly any participants published their approach.

Where deception detection focuses on single statements, style-based text categorization as proposed by [19] assesses entire texts. Common applications are author profiling (age, gender, etc.) and genre classification.

¹²<http://www.fakenewschallenge.org/>

Though susceptible to authors who can modify their writing style, such obfuscations may be detectable (e.g., [4]). As an early precursor to fake news detection, [23] train models to identify news items that were automatically generated. Currently, text categorization methods for fake news detection focus mostly on satire detection (e.g., [202], [250]). [199] perform a statistical analysis of the stylistic differences between real, satire, hoax, and propaganda news. We make use of their results by incorporating the best-performing style features identified.

Finally, two preprint papers have been shared. [109] use style features for fake news detection. However, the relatively high accuracies reported must be taken with a grain of salt: their two datasets comprise only 70 news articles each, whose ground-truth is based on where an article came from, instead of resulting from a per-article expert review as in our case; their final classifier uses only 4 features (number of nouns, type-token ratio, word count, number of quotes), which can be easily manipulated; and based on their experimental setup, it cannot be ruled out that the classifier simply differentiates news portals rather than fake and real articles. We avoid this problem by testing our classifiers on articles from portals which were not represented in the training data. Similarly, [188] also report on constructing two datasets comprising around 240 and 200 news article *excerpts* (i.e., the 5-sentence lead) with a balanced distribution of fake vs. real. The former was collected via crowdsourcing, asking workers to write a fake news item based on a real news item, the latter was collected from the web. For style analysis, the former dataset may not be suitable, since the authors note themselves that “workers succeeded in mimicking the reporting style from the original news”. The latter dataset encompasses only celebrity news (i.e., yellow press), which introduces a bias. Their feature selection follows that of [202], which is covered by our experiments, but also incorporates topic features, rendering the resulting classifier not generalizable.

4.2.2 The BuzzFeed-Webis Fake News Corpus

This section introduces the BuzzFeed-Webis Fake News Corpus 2016, detailing its construction and annotation by professional journalists employed at BuzzFeed, as well as key figures and statistics.¹³ Note that the main contribution from our side has been the timely archiving and processing of the articles annotated by BuzzFeed. Though the original annotations are still available, our archived version has attracted significant attention, counting more than 1500 downloads to date. This attention highlights the importance

¹³Corpus download: <https://doi.org/10.5281/zenodo.1239675>

of web archiving for assessment tasks like hyperpartisan and fake news detection. We thank Craig Silverman, Lauren Strapagiel, Hamza Shaban, Ellie Hall, and Jeremy Singer-Vine from BuzzFeed for making their data available, enabling our research.

Corpus Construction

The corpus encompasses the output of 9 publishers on 7 workdays close to the US presidential elections 2016, namely September 19 to 23, 26, and 27. Table 4.3 gives an overview. Among the selected publishers are six prolific hyperpartisan ones (three left-wing and three right-wing), and three mainstream ones. All publishers earned Facebook’s blue checkmark , indicating authenticity and an elevated status within the network. Every post and linked news article has been fact-checked by 4 BuzzFeed journalists, including about 19% of posts forwarded from third parties. Having checked a total of 2,282 posts, 1,145 mainstream, 471 left-wing, and 666 right-wing, [218] reported key insights as a data journalism article. The annotations were published alongside the article.¹⁴ However, this data only comprises URLs to the original Facebook posts. To construct our corpus, we archived the posts, the linked articles, and attached media as well as relevant meta data to ensure long-term availability. Due to the rapid pace at which the publishers change their websites, we were able to recover only 1,627 articles, 826 mainstream, 256 left-wing, and 545 right-wing.

Manual fact-checking A binary distinction between fake and real news turned out to be infeasible, since hardly any piece of fake news is entirely false, and pieces of real news may not be flawless. Therefore, posts were rated “mostly true,” “mixture of true and false,” “mostly false,” or, if the post was opinion-driven or otherwise lacked a factual claim, “no factual content.” Four BuzzFeed journalists worked on the manual fact-checks of the news articles: to minimize costs, each article was reviewed only once and articles were assigned round robin. The ratings “mixture of true and false” and “mostly false” had to be justified, and, when in doubt about a rating, a second opinion was collected, whereas disagreements were resolved by a third one. Finally, all news rated “mostly false” underwent a final check to ensure the rating was justified, lest the respective publishers would contest it.

The journalists were given the following guidance:

¹⁴<http://github.com/BuzzFeedNews/2016-10-facebook-fact-check>

Mostly true: The post and any related link or image are based on factual information and portray it accurately. The authors may interpret the event/info in their own way, so long as they do not misrepresent events, numbers, quotes, reactions, etc., or make information up. This rating does not allow for unsupported speculation or claims.

Mixture of true and false (mix, for short): Some elements of the information are factually accurate, but some elements or claims are not. This rating should be used when speculation or unfounded claims are mixed with real events, numbers, quotes, etc., or when the headline of the link being shared makes a false claim but the text of the story is largely accurate. It should also only be used when the unsupported or false information is roughly equal to the accurate information in the post or link. Finally, use this rating for news articles that are based on unconfirmed information.

Mostly false: Most or all of the information in the post or in the link being shared is inaccurate. This should also be used when the central claim being made is false.

No factual content (n/a, for short): This rating is used for posts that are pure opinion, comics, satire, or any other posts that do not make a factual claim. This is also the category to use for posts that are of the “Like this if you think...” variety.

Limitations

Given the significant workload (i.e., costs) required to carry out the aforementioned annotations, the corpus is restricted to the given temporal period and biased toward the US culture and political landscape, comprising only English news articles from a limited number of publishers. Annotations were recorded at the article level, not at statement level. For text categorization, this is sufficient. At the time of writing, our corpus is the largest of its kind that has been annotated by professional journalists.

Corpus Statistics

Table 4.3 shows the fact-checking results and some key statistics per article. Unsurprisingly, none of the mainstream articles are mostly false, whereas 8 across all three publishers are a mixture of true and false. Disregarding non-factual articles, a little more than a quarter of all hyperpartisan left-wing articles were found faulty: 15 articles mostly false, and 51 a mixture of true and false. Publisher “The Other 98%” sticks out by achieving an almost perfect score. By contrast, almost 45% of the right-wing articles are a mixture of true and false (153) or mostly false (72). Here, publisher “Right

TABLE 4.3: The BuzzFeed-Webis Fake News Corpus 2016 at a glance.

Orientation Publisher	Fact-checking results					Key statistics per article				
	true	mix	false	n/a	Σ	Paragraphs	Links		Words	
							extern	all	quoted	all
<i>Mainstream</i>	806	8	0	12	826	20.1	2.2	3.7	18.1	692.0
ABC News	90	2	0	3	95	21.1	1.0	4.8	21.0	551.9
CNN	295	4	0	8	307	19.3	2.4	2.5	15.3	588.3
Politico	421	2	0	1	424	20.5	2.3	4.3	19.9	798.5
<i>Left-wing</i>	182	51	15	8	256	14.6	4.5	4.9	28.6	423.2
Addicting Info	95	25	8	7	135	15.9	4.4	4.5	30.5	430.5
Occupy Democrats	55	23	6	0	91	10.9	4.1	4.7	29.0	421.7
The Other 98%	32	3	1	1	30	20.2	6.4	7.2	21.2	394.5
<i>Right-wing</i>	276	153	72	44	545	14.1	2.5	3.1	24.6	397.4
Eagle Rising	107	47	25	36	214	12.9	2.6	2.8	17.3	388.3
Freedom Daily	48	24	22	4	99	14.6	2.2	2.3	23.5	419.3
Right Wing News	121	82	25	4	232	15.0	2.5	3.6	33.6	396.6
Σ	1264	212	87	64	1627	17.2	2.7	3.7	20.6	551.0

Wing News” sticks out by supplying more than half of mixtures of true and false alone, whereas mostly false articles are equally distributed.

Regarding key statistics per article, it is interesting that the articles from all mainstream publishers are on average about 20 paragraphs long with word counts ranging from 550 words on average at ABC News to 800 at Politico. Except for one publisher, left-wing articles and right-wing articles are shorter on average in terms of paragraphs as well as word count, averaging at about 420 words and 400 words, respectively. Left-wing articles quote on average about 10 words more than the mainstream, and right-wing articles 6 words more. When articles comprise links, they are usually external ones, whereas ABC News rather uses internal links, and only half of the links found at Politico articles are external. Left-wing news articles stick out by containing almost double the amount of links across publishers than mainstream and right-wing ones.

Operationalizing Fake News

In our experiments, we operationalize the category of fake news by joining the articles that were rated mostly false with those rated a mixture of true and false. Arguably, the latter may not be exactly what is deemed “fake news” (as in: a complete fabrication), however, practice shows fake news are hardly ever devoid of truth. More often, true facts are misconstrued or framed badly. In our experiments, we hence call mostly true articles real news, mostly false plus mixtures of true and false—except for satire—fake news, and disregard all articles rated non-factual.

4.2.3 Discriminating Hyperpartisan News, Fake News, and Satire by Writing Style

This section covers our methodology, including our feature set to capture writing style. For sake of reproducibility, all our code has been published.¹⁵

Moreover, we report on the results of two series of experiments that investigate style differences and similarities between hyperpartisan and mainstream news, and between fake, real, and satire news:

1. Can hyperpartisanship be distinguished from the mainstream?
2. Is style-based fake news detection feasible?
3. Can fake news be distinguished from satire?

We employ the following setup:

Style features and feature selection Our writing style model incorporates common features as well as ones specific to the news domain. The former are n-grams, n in [1, 3], of characters, stop words, and parts-of-speech. Further, we employ 10 readability scores¹⁶ and dictionary features, each indicating the frequency of words from a tailor-made dictionary in a document, using the General Inquirer Dictionaries as a basis [226]. The domain-specific features include ratios of quoted words and external links, the number of paragraphs, and their average length.

In each of our experiments, we carefully select from the aforementioned features the ones worthwhile using: all features are discarded that are hardly represented in our corpus, namely word tokens that occur in less than 2.5% of the documents, and n-gram features that occur in less than 10% of the documents. Discarding these features prevents overfitting and improves the chances that our model will generalize.

If not stated otherwise, our experiments share a common setup. In order to avoid biases from the respective training sets, we balance them using oversampling. Furthermore, we perform 3-fold cross-validation where each fold comprises one publisher from each orientation, so that the classifier does not learn a publisher's style. We use WEKA's [84] random forest implementation with default settings.

¹⁵Code download: <http://www.github.com/webis-de/ACL-18>

¹⁶Automated Readability Index, Coleman Liau Index, Flesh Kincaid Grade Level and Reading Ease, Gunning Fog Index, LIX, McAlpine EFLAW Score, RIX, SMOG Grade, Strain Index

Baselines We employ four baseline models: a topic-based bag of words model, often used in the literature, but less practical since news topics change frequently and drastically; a model using only the domain-specific news style features to check whether the differences between categories measured as corpus statistics play a significant role; and naive baselines that classify all items into one of the categories in question, relating our results to the class distributions.

Performance measures Classification performance is measured as accuracy, and class-wise precision, recall, and F_1 . We favor these measures over, e.g., areas under the ROC curve or the precision recall curve for simplicity sake. Also, the tasks we are tackling are new, so that little is known to date about user preferences. This is also why we chose the evenly-balanced F_1 .

Results: Hyperpartisanship vs. Mainstream

This experiment uncovered an odd behavior of our classifier: it would often misjudge left-wing for right-wing news, while being much better at distinguishing both combined from the mainstream. To explain this behavior, we hypothesized that *maybe* the writing style of the hyperpartisan left and right are more similar to one another than to the mainstream. To investigate this hypothesis, we devised one additional validation experiment, yielding two sources of evidence instead of just one.

Predicting orientation Table 4.4 shows the classification performance of a ternary classifier trained to discriminate left, right, and mainstream—an obvious first experiment for our dataset. Separating the left and right orientation from the mainstream does not work too well: the topic baseline outperforms the style-based models with regard to accuracy, whereas the results for class-wise precision and recall are a mixed bag. The left-wing articles are apparently significantly more difficult to be identified compared to articles from the other two orientations. When we inspected the confusion matrix (not shown), it turned out that 66% of misclassifications of left-wing articles are falsely classified as right-wing articles, whereas 60% of all misclassified right-wing articles are classified as mainstream articles. Misclassified mainstream articles spread almost evenly across the other classes.

The poor performance of the domain-specific news style features by themselves demonstrate that orientation cannot be discriminated based on the basic corpus characteristics observed with respect to paragraphs, quotations, and hyperlinks. This holds for all subsequent experiments.

TABLE 4.4: Performance of predicting orientation.

Features	Accuracy	Precision			Recall			F_1 -score		
		all	left	right main.	left	right	main.	left	right	main.
Style	0.60	0.21	0.56	0.75	0.20	0.59	0.74	0.20	0.57	0.75
Topic	0.64	0.24	0.62	0.72	0.15	0.54	0.86	0.19	0.58	0.79
News style	0.39	0.09	0.35	0.59	0.14	0.36	0.49	0.11	0.36	0.53
All-left	0.16	0.16	-	-	1.00	0.0	0.0	0.27	-	-
All-right	0.33	-	0.33	-	0.0	1.00	0.0	-	0.50	-
All-main.	0.51	-	-	0.51	0.0	0.0	1.00	-	-	0.68

TABLE 4.5: Performance of predicting hyperpartisanship.

Features	Accuracy	Precision		Recall		F_1 -score	
		all	hyp.	main.	hyp.	main.	hyp.
Style	0.75	0.69	0.86	0.89	0.62	0.78	0.72
Topic	0.71	0.66	0.79	0.83	0.60	0.74	0.68
News style	0.56	0.54	0.58	0.65	0.47	0.59	0.52
All-hyp.	0.49	0.49	-	1.00	0.0	0.66	-
All-main.	0.51	-	0.51	0.0	1.00	-	0.68

Predicting hyperpartisanship Given the apparent difficulty of telling apart individual orientations, we did not frantically add features or switch classifiers to make it work. Rather, we trained a binary classifier to discriminate hyperpartisanship in general from the mainstream. Table 4.5 shows the performance values. This time, the best classification accuracy of 0.75 at a remarkable 0.89 recall for the hyperpartisan class is achieved by the style-based classifier, outperforming the topic baseline.

Comparing Table 4.4 and Table 4.5, we were left with a riddle: all other things being equal, how could it be that hyperpartisanship in general can be much better discriminated from the mainstream than individual orientation? Attempts to answer this question gave rise to our aforementioned hypothesis that, perhaps, the writing style of hyperpartisan left and right are not altogether different, despite their opposing agendas. Or put another way, if style and topic are orthogonal concepts, then being an extremist should not exert a different style dependent on political orientation. Excited, we sought ways to *independently* disprove the hypothesis, for which we then used leave-out classification.

Validation using leave-out classification If left-wing and right-wing articles have a more similar style than either of them compared to mainstream articles, then what class would a binary classifier assign to a left-wing article, if it were trained to distinguish only the right-wing from the main-

TABLE 4.6: Ratio of left articles misclassified right when omitting left articles from training, and vice versa.

Features	Left		Right	
	Trained on:	right+main. all	left+main. all	all
Style	0.74	0.90	0.66	0.89
Topic	0.68	0.79	0.48	0.85
News style	0.52	0.61	0.47	0.66

stream, and vice versa? Table 4.6 shows the results of this experiment. As indicated by proportions well above 0.50, full style-based classifiers have a tendency of classifying left as right and right as left. The topic baseline, though, gets confused especially when omitting right articles from the training set with performance close to random. The fact that the topic baseline works better when omitting left from the training set may be explainable: leading up to the elections, the hyperpartisan left was often merely reacting to topics prompted by the hyperpartisan right, instead of bringing up their own.

With caution, we conclude that the evidence gained from our independent experimental setups supports our hypothesis that the hyperpartisan left and the hyperpartisan right have more in common in terms of writing style than any of the two have with the mainstream. Another more tangible (e.g., practical) outcome of this experiment is the finding that hyperpartisan news can apparently be discriminated well from the mainstream: in particular the high recall of 0.89 at a reasonable precision of 0.69 gives us confidence that, with some further effort, a practical classifier can be built that detects hyperpartisan news at scale and in real time, since an article’s style can be assessed immediately without referring to external information.

Results: Fake vs. Real (vs. Satire)

This series of experiments targets research questions (2) and (3). Again, we conduct two experiments, where the first is about predicting veracity, and the second about discriminating satire.

Predicting veracity When taking into account that the mainstream news publishers in our corpus did not publish any news items that are mostly false, and only very few instances that are mixtures of true and false, we may safely disregard them for the task of fake news detection. A reliable classifier for hyperpartisan news can act as a pre-filter for a subsequent,

more in-depth fake news detection approach, which may in turn be tailored to a much more narrowly defined classification task. We hence use only the left-wing articles and the right-wing articles of our corpus for our attempt at a style-based fake news classifier.

Table 4.7 shows the performance values for a generic classifier that predicts fake news across orientations, and orientation-specific classifiers that have been individually trained on articles from either orientation. Although all classifiers outperform the naive baselines of classifying everything into one of the classes in terms of precision, the slight increase comes at the cost of a large decrease in recall. While the orientation-specific classifiers are slightly better for most metrics, none of them outperform the naive baselines regarding the F_1 -score. We conclude that style-based fake news classification simply does not work in general.

Predicting satire Yet, not all fake news are the same. One should distinguish satire from the rest, which takes the form of news but lies more or less obviously to amuse its readers. Regardless the problems that spreading fake news may cause, satire should never be filtered, but be discriminated from other fakes. Table 4.8 shows the performance values of our classifier in the satire-detection setting used by [202] (the S-n-L News DB corpus), distinguishing satire from real news. This setting uses a balanced 3:1 training-to-test set split over 360 articles (180 per class). As can be seen, our style-based model significantly outperforms all baselines across the board, achieving an accuracy of 0.82, and an F score of 0.81. It clearly improves over topic classification, but does not outperform Rubin et al.’s classifier, which includes features based on topic, absurdity, grammar, and punctuation. We argue that incorporating topic into satire detection is not appropriate, since the topics of satire change along the topics of news. A classifier with topic features therefore does not generalize. Apparently, a style-based model is competitive, and we believe that satire can be detected at scale this way, so as to prevent other fake news detection technology from falsely filtering it.

4.2.4 The PAN-SemEval-Hyperpartisan-News-Detection Dataset

To study hyperpartisanship at an larger scale, we collected, archived, and now provide two further datasets with this task. One has 1,273 articles, each labeled manually, while the second, larger dataset of 754,000 articles is labeled in a semi-automated manner via distant supervision at the publisher level. These datasets are further split into public and private sets. We re-

TABLE 4.7: Performance of predicting veracity.

Features	Accuracy		Precision		Recall		F_1 -score	
	all		fake	real	fake	real	fake	real
<i>Generic classifier</i>								
Style	0.55		0.42	0.62	0.41	0.64	0.41	0.63
Topic	0.52		0.41	0.62	0.48	0.55	0.44	0.58
<i>Orientation-specific classifier</i>								
Style	0.55		0.43	0.64	0.49	0.59	0.46	0.61
Topic	0.58		0.46	0.65	0.45	0.66	0.46	0.66
All-fake	0.39		0.39	-	1.00	0.0	0.56	-
All-real	0.61		-	0.61	0.0	1.00	-	0.76

TABLE 4.8: Performance of predicting satire (sat.).

Features	Accuracy		Precision		Recall		F_1 -score	
	all		sat.	real	sat.	real	sat.	real
Style	0.82		0.84	0.80	0.78	0.85	0.81	0.82
Topic	0.77		0.78	0.75	0.74	0.79	0.76	0.77
All-sat.	0.50		0.50	-	1.00	0.0	0.67	-
All-real	0.50		-	0.50	0.00	1.00	-	0.67
Rubin et al.	n/a		0.90	n/a	0.84	n/a	0.87	n/a

leased the public set for the model training, tuning, and evaluation,¹⁷ while the unreleased private set is used to enable blind, cloud-based evaluation.

As online news articles are published mainly in the HTML format, both datasets use a unified HTML-like format (see Figure 4.7). The web archives for the larger part are available upon request as they are too large for regular hosting. We restricted the distributed markup for the article content to paragraphs (`<p>`), links (`<a>`), and quotes (`<q>`). We distinguished *internal* links to the other pages of the same domain, from which we removed the href-attribute value to avoid classifiers fitting to them; and links to *external* domains, for which we kept the attribute. An XML schema that exactly specifies the format is distributed along the datasets.

Dataset Part Annotated By Article

We gathered a crowdsourced dataset of 1,273 articles, each labeled manually by 3 annotators [236]. These articles were published by active hyperpartisan and mainstream websites and were all assured to contain political news. Annotators were asked to rate each article’s bias on the following 5-point Likert scale:

¹⁷<https://doi.org/10.5281/zenodo.1489920>

```

<article id="0182515" published-at="2007-01-22" title="They're crumbling">
<p>What a pleasant surprise to see Jacques Leslie, a journalist and real
expert on dams, with a long <a
href="http://www.nytimes.com/2007/01/22/opinion/22leslie.2.html
?ex=1327122000&en=42caf99f05e4cba8&ei=5090&partner=
rssuserland&emc=rss" type="external">op-ed</a> on the hallowed pages
of the New York Times. Leslie, author of <a href="" type="internal">Deep
Water: The Epic Struggle Over Dams, Displaced People and the
Environment</a>, highlights the threat posed by poorly maintained and
increasingly failing dams around the country:</p>
<p>Unlike, say, waterways and sanitation plants, a majority of dams - 56
percent of those inventoried - are privately owned, which is one reason dams
are among the country's most dangerous structures. Many private owners
can't afford to repair aging dams; some owners go so far as to resist paying
by tying up official repair demands in court or campaigning to weaken state
dam safety laws.</p>
<p>Kinda makes you want to find out what is upstream.</p> </article>

```

FIGURE 4.7: Example of a non-hyperpartisan article in our dataset. An archived version of the original article is available at <https://web.archive.org/web/20121006194050/> <https://grist.org/article/remember-the-dams/>.

1. No hyperpartisan content
2. Mostly unbiased, non-hyperpartisan content
3. Not Sure
4. Fair amount of hyperpartisan content
5. Extreme hyperpartisan content

We removed all articles from the dataset with low agreement score and the aggregated rating of “not sure” (see Vincent and Mestre for more details). We then binarized the labels to hyperpartisan (average rating of 4 or 5) and not (average rating of 1 or 2). The final by-article set achieved an inter-annotator agreement of 0.5 Krippendorff’s alpha. Of the remaining 1,273 articles, 645 were published as a training dataset, whereas the other 628 (50% hyperpartisan and 50% not) were kept private for the evaluation. To ensure that classifiers could not profit from overfitting to publisher style, we made sure there was no overlap between the publishers of the articles between these two sets.

Dataset Part Annotated By Publisher

To allow for methods that require huge amounts of training data, we compiled a dataset of 754,000 articles, each labeled as per the bias of their respective publisher. To create this dataset, we cross-checked two publicly

available news publisher bias lists compiled by media professionals from BuzzFeed news¹⁸ and Media Bias Fact Check.¹⁹ The former was created by BuzzFeed journalists as a basis for a news article, whereas the latter is Media Bias Fact Check’s main product. While both lists contain several hundred news publishers, they disagree only for nine, which we removed from our dataset.

We then crawled, archived, and post-processed the articles available on the publishers’ websites and Facebook feeds. We archived all articles using the Webis Web Archiver (cf. Section 2.2), that removes pop-overs and similar things preventing the article content from being loaded. After filtering out publishers that did not mainly publish political articles or had no political section to which we could restrict our crawl, we were left with 383 publishers. For each of the publishers’ websites we wrote a content-wrapper to extract the article content and relevant meta data from the HTML DOM. We then removed all articles that were too short to contain news,²⁰ that are not written in English, or that contain obvious encoding errors. The final dataset consisted of 754,000 articles, split into a public training set (600,000 articles), a public validation set (150,000 articles) and a non-public test set (4,000 articles). Like for the by-article dataset, we ensured that there is no overlap of publishers between the sets. Each set consists of 50% articles from non-hyperpartisan publishers and 50% articles from hyperpartisan publishers, the latter again being 50% from left-wing and 50% from right-wing publishers.

4.2.5 Participating Systems at SemEval-2019 Task 4

The PAN-SemEval-Hyperpartisan-News-Detection dataset constituted the core of the SemEval-2019 Task 4: Hyperpartisan News Detection, which we organized and on which we report in the following. This task attracted several teams, who tackled the task with a very diverse and interesting set of solutions. The teams employed very different sets of features, a wide variety of classifiers, and also employed the large by-publisher dataset in different ways. Around half of the submissions used hand-crafted features. In the following, we give an overview of the submitted approaches. For a more readable and condensed form, we only use the team names here, which were chosen from fictional journalistic characters or entities (see Table 4.9 for references).

¹⁸<https://github.com/BuzzFeedNews/2017-08-partisan-sites-and-facebook-pages>

¹⁹<https://mediabiasfactcheck.com>

²⁰Based on manual inspection of a hundred short articles, we set the threshold to 40 words.

Features

The teams that participated in this task employed a variety of features, including standard word n -grams (also unigrams, i.e., bag-of-words), word embeddings, stylometric features, HTML features like the target of hyperlinks, and a meta data feature in the form of the publication date.

N-Grams Most teams that used hand-crafted features also included word n -grams: Teams *Pioquinto Manterola* and *Tintin* used them as their only features. Character and part-of-speech n -grams were, for example, used by *Paparazzo*.

Word embeddings Many teams integrated word embeddings into their approach. Frequently used were Word2Vec, fastText, and GloVe. Noticeably, *Tom Jumbo Grumbo* relied exclusively on them. *Bertha von Suttner* relied on ELMo embeddings [189], which have the advantage of modeling polysemy. Where the aforementioned word embeddings all rely on neural networks, *Doris Martin* employed a document representation based on word clusters as part of their approach.

BERT [70], which jointly conditions on both left and right context in all layers, is a rather new technique that was used by several teams. *Peter Parker* directly applied a freely available pre-trained BERT model to the task, whereas *Howard Beale* and *Clint Buchanan* trained their own BERT models on the by-publisher dataset and then performed fine-tuning on the by-article dataset. Despite the fine-tuning, *Howard Beale* reported overfitting issues for this strategy. Going one step further, *Jack Ryder* and *Yeon Zi* integrated BERT in their neural network architectures.

Stylometry Many teams used stylometric features including punctuation and article structure (*Steve Martin*, *Spider Jerusalem*, *Fernando Pessa*, *Ned Leeds*, *Carl Kolchak*, *Orwellian Times*), readability scores (*Ned Leeds*, *Pistachon*, *Steve Martin*, *Orwellian Times*, *D X Beaumont*), or psycholinguistic lexicons (*Ned Leeds*, *Spider Jerusalem*, *Steve Martin*, *Pistachon*). *Borat Sagdiyev* employed a self-compiled list of trigger words that contains mostly profanities. They noticed that such words are used more often in hyperpartisan articles.

Emotionality Several teams used sentiment and emotion features, either based on libraries (*Borat Sagdiyev*, *Steve Martin*, *Carl Kolchak*) or lexicons (*Spider Jerusalem*, *D X Beaumont*). Notably, *Kermit the Frog* uses sentiment

detection only. *Vernon Fenwick* and *D X Beaumont* used subjectivity and polarity metrics as features.

Named entities *Borat Sagdiyev* used named entity types as features. In preliminary tests only the type of “nationalities or religious and political groups” was found to be predictive.

Quotations A few teams treated quotations separately. Whereas *Spider Jerusalem* and *Borat Sagdiyev* created separate features from quotations, *The Ankh Morpork Times* filtered them out for not necessarily representing the views of the author.

Hyperlinks Only few teams considered hyperlinks. Both *Borat Sagdiyev* and *Steve Martin* used external lists of partisan web pages to count how often an article links to partisan and non-partisan pages. They assume that articles tend to link other articles on the same side of the political spectrum.

Publication date Based on the conjecture that months around American elections could see more hyperpartisan activity, *Borat Sagdiyev* used the publication month and year as separate features.

Classifiers

While many different classifiers were used overall, neural networks were the most frequent, which mirrors the current trend in text classification.

The most popular type of neural networks among the participants were convolutional ones (CNNs), which employ convolving filters over neighboring words. Many teams cited the architecture by Kim [138]. *Xenophilius Lovegood* added a second layer to their CNN in order to encode more information about the articles, using both available and custom-learned embeddings. While *Pioquinto Manterola* experimented with a CNN, it suffered from overfitting and was thus not used for the final submission. *Peter Brinkmann* built a submission using available embeddings. *Brenda Starr* combined a CNN with a sentence-level bidirectional recurrent neural network and an attention mechanism to a complex architecture. A similar approach was employed by *The Ankh Morpork Times*. An ensemble of three CNN-based models was used by *Bertha von Suttner*. *Steve Martin* used a character bigram CNN as part of their approach.

Next to CNNs, long short term memory networks (LSTM) were employed by *Kit Kittredge* and *Miles Clarkson*. The latter extended the network

with an attention model. Moreover, *Joseph Rouletabille* used the hierarchical attention network of Yang et al. [252].

Besides neural networks, a wide variety of classifiers were used. A few teams opted for SVMs (e.g., *The Orwellian Times*), others for random forests (e.g., *Fernando Pessa*), linear models (e.g., *Pistachon*), the Naive Bayes model (e.g., *Carl Kolchak*), XGBOOST (*Clark Kent*), Maxent (*Doris Martin*), and rule-based models (*Harry Friberg*). *Morbo* used ULMFit [111] to adapt a language model pre-trained on Wikipedia articles to the articles and classes of this task.

Usage of the By-publisher Dataset Part

The submitted systems can also be distinguished by whether and how they used the large, distantly-supervised by-publisher dataset. Though much larger than the by-article set, its labels are noisy, whereas the opposite holds for the by-article dataset. One of the key challenges faced by many teams was how to train a powerful expressive model on the smaller dataset without overfitting. Most teams made use of the larger dataset in some form or another. A challenge faced by some of the teams was that the test split of the by-article dataset was balanced between classes, whereas the corresponding training dataset was not.

Several systems trained the whole or part of their system on the by-publisher dataset. Some extracted features like n -grams (e.g., *Sally Smedley*), word clusters (*Doris Martin*), or neural network word embeddings (e.g., *Clint Buchanan*). Others used the larger dataset to perform hyperparameter search (e.g., *Miles Clarkson*). Many teams trained their models using the by-publisher dataset only (*Pistachon*, *Joseph Rouletabille*, *Xenophilius Lovegood*, *Peter Brinkmann*, and *Kit Kittredge*).

To reduce the noise in the distantly-supervised data, some teams used only a subset of it. *Yeon Zi*, *Borat Sagdiyev* and *The Anhk Morpork Times* fitted a model on the by-article dataset and ran it on the by-publisher one: the articles of the by-publisher dataset that were misclassified by this model, were presumed to be noisy and filtered out.

Fairness and Reproducibility

In this shared task, we asked participants to submit their software instead of just its run output. The submissions were executed at our site on the test data, enabling us to keep the test data entirely secret. This has two important advantages over traditional shared task setups: first, software submission gives rise to blind evaluation; and second, it maximizes the replicability

and the reproducibility of each participant’s approach. To facilitate software submission and to render it feasible in terms of work overhead and flexibility for both participants and organizers, we employ the TIRA Integrated Research Architecture [197].

A shortcoming of traditional shared task setups is that typically the test data are shared with participants, albeit without ground truth. Although participants in shared tasks generally exercise integrity and do not analyze the test data other than running their software on it, we have experienced cases to the contrary. Such problems particularly arise in shared tasks where the stakes are higher than usual; when monetary incentives are offered or winning results in high visibility. A partial workaround is to share the test data only very close to the final submission deadline, minimizing analysis opportunities. But if sharing the test data is impossible for reasons of sensibility and proprietariness, or because the ground truth can be easily reverse-engineered, a traditional shared task cannot be held.

Another shortcoming of traditional shared tasks (and many computer science publications in general) is their lack of reproducibility. Although sharing the software underlying experiments as well as the trained models is easy, and although it would greatly aid reproducibility, this is still rare. Typically, all that remains after a shared task are the papers and datasets published. Given that shared tasks often establish a benchmark for the task in question, acting normative for future evaluations, this outcome is far from optimal and comparably wasteful. All of the above can be significantly improved upon by asking participants not to submit their software’s run output, but the software itself. However, this entails a significant work overhead for organizers, especially for larger tasks.

In order to mitigate the work overhead, we employ TIRA. In a nutshell, TIRA implements evaluation as a service in the form of a cloud-based evaluation platform. Participants deploy their software into virtual machines hosted at TIRA’s cloud, and then remotely control the machines and the software within, executing it on the test data. The test data are available only within the cloud, and made accessible on demand so that participants cannot access it directly. At execution time, the virtual machine is disconnected from the internet, copied, and only the copy gets access to the test data. Once the automatically executed software terminates, its run output is saved and the virtual machine copy is destroyed to prevent data leaks. This way, all submitted pieces of software can be archived in working condition, and be re-evaluated at a later time, even on new datasets.

4.2.6 Results of the SemEval-2019 Task 4

A total of 42 teams completed the task, representing more than twenty countries between them, including India, China, the USA, Japan, Vietnam, and many European countries. Table 4.9 shows the accuracy, precision, recall, and F_1 score for each team, sorted by accuracy. This task used accuracy as the main metric to represent a filtering scenario. The accuracy scores ranged from 0.462 up to 0.822.

The results show a range of trade-offs between precision and recall and the resulting F_1 scores. The highest F_1 was 0.821 with a precision of 0.815 and a recall of 0.828; the highest precision was 0.883 with a recall of 0.672 (F_1 : 0.763); and the highest recall was 0.971 with a relatively low precision of 0.542 (F_1 : 0.696).

Methods Used by the Top Teams

While the winning team, *Bertha von Suttner*, used deep learning (sentence-level embeddings and a convolutional neural network) the second-placed team, *Vernon Fenwick*, took a different approach and combined sentence embeddings with more domain-specific features and a linear model. Out of the top five teams, only two used “pure” deep learning models of neural networks without any domain-specific, hand-crafted features, showing no single method has a clear advantage over others.

Bertha von Suttner used a model based on ELMo embeddings [189] and trained on the by-article dataset. After minimal preprocessing, a pre-trained ELMo was applied onto each token of each sentence, and then averaged, to obtain average sentence embeddings. The sentence embeddings were later passed through a CNN, batch-normalized, followed by a dense layer and a sigmoid function to obtain the final probabilities. The final model was an ensemble of the 3 best-performing models of a 10-fold cross-validation. The authors tried to include the by-publisher dataset, but found in their preliminary tests no approach to profit from the large data.

The second and third best teams used linear models as their main predictor and embeddings as features, training on the by-article dataset only. *Vernon Fenwick* extracted sentence embeddings with the Universal Sentence Encoder (USE) [46], while *Sally Smedley* used BERT to generate contextual embeddings. Both teams also employed hand-crafted, domain-specific features. *Vernon Fenwick* extracted article-level and sentence-level polarity, bias, and subjectivity, among others, while *Sally Smedley* used the by-publisher dataset to extract key discriminative phrases, which they later looked up in the training data.

TABLE 4.9: For each team and dataset, the performance of the submission that reached the highest accuracy is shown. If a team published their code, the links (in the digital thesis version) to the respective repository. We also forked all repositories for archival at <https://github.com/hyperpartisan-news-challenge>.

Submission		By-article dataset					By-publisher dataset					
Team name	Authors	Code	Rank	Acc.	Prec.	Rec.	F ₁	Rank	Acc.	Prec.	F ₁	
Bertha von Suttner	Jiang et al.		1	0.822	0.871	0.755	0.809	8	0.643	0.616	0.762	0.681
Vernon Fenwick	Srivastava et al.		2	0.820	0.815	0.828	0.821					
Sally Smedley	Hanawa et al.		3	0.809	0.823	0.787	0.805	11	0.625	0.640	0.571	0.603
Tom Jumbo Grumbo	Yeh et al.		4	0.806	0.858	0.732	0.790	13	0.619	0.592	0.762	0.667
Dick Preston	Isbister and Johansson		5	0.803	0.793	0.818	0.806	27	0.514	0.520	0.352	0.420
Borat Sagdiyev	Palić et al.		6	0.791	0.883	0.672	0.763	19	0.592	0.644	0.412	0.502
Morbo	Isbister and Johansson		7	0.790	0.772	0.822	0.796	16	0.601	0.587	0.679	0.630
Howard Beale	Mutlu et al.		8	0.783	0.837	0.704	0.765	9	0.641	0.606	0.806	0.692
Ned Leeds	Stevanoski and Gievská		9	0.775	0.865	0.653	0.744	22	0.573	0.546	0.857	0.667
Clint Buchanan	Drissi et al.		10	0.771	0.832	0.678	0.747					
Yeon Zi	Lee et al.		11	0.758	0.744	0.787	0.765	5	0.663	0.635	0.766	0.694
Tony Vincenzo	Staykovski		12	0.750	0.764	0.723	0.743					
Paparazzo	Nguyen et al.		13	0.747	0.754	0.732	0.743	24	0.530	0.530	0.541	0.535
Steve Martin	Joo and Hwang		14	0.745	0.853	0.592	0.699	18	0.597	0.625	0.483	0.545
Eddie Brock	Šajatović et al.		15	0.744	0.782	0.675	0.725	10	0.631	0.681	0.491	0.571
Ankh Morpork Times	Almendros et al.		16	0.742	0.811	0.631	0.710	21	0.588	0.646	0.389	0.486
Spider Jerusalem	Alabdulkarim and Alhindi		17	0.742	0.814	0.627	0.709					
Carl Kolchak	Chen et al.		18	0.739	0.729	0.761	0.745					
Doris Martin	Agerri		19	0.737	0.754	0.704	0.728					
Pistachon	Saleh et al.		20	0.729	0.724	0.742	0.733	15	0.608	0.638	0.499	0.560
Joseph Rouletabille	Moreno et al.		21	0.725	0.788	0.615	0.691	2	0.680	0.640	0.827	0.721
Fernando Pessa	Cruz et al.		22	0.717	0.806	0.570	0.668	17	0.600	0.585	0.681	0.630
Pioquinto Manterola	Sengupta and Pedersen		23	0.704	0.741	0.627	0.679					
Miles Clarkson	Zhang et al.		24	0.683	0.745	0.557	0.638	6	0.652	0.612	0.832	0.705
Xenophilus Lovegood	Zehe et al.		25	0.675	0.619	0.914	0.738	4	0.663	0.632	0.781	0.699
Orwellian Times	Knauth		26	0.672	0.654	0.729	0.690	23	0.537	0.530	0.658	0.587
Tintin	Bestgen		27	0.656	0.642	0.707	0.673	1	0.706	0.742	0.632	0.683
D X Beaumont	Amason et al.		28	0.653	0.597	0.939	0.730					
Jack Ryder	Shaprin et al.		29	0.646	0.646	0.646	0.646	7	0.645	0.600	0.869	0.710
Kermit the Frog	Anthonio and Kloppenburg		30	0.621	0.582	0.860	0.694	20	0.589	0.575	0.681	0.623
Billy Batson	Kreutz et al.		31	0.615	0.568	0.962	0.714					
Peter Brinkmann	Färber et al.		32	0.602	0.560	0.955	0.706	28	0.497	0.496	0.344	0.406
Anson Bryson	Stiff and Medero		33	0.592	0.720	0.303	0.426					
Sarah Jane Smith	Chakravartula et al.		34	0.591	0.554	0.933	0.695	14	0.612	0.586	0.765	0.664
Kit Kittredge	Cramerus and Scheffler		35	0.578	0.547	0.908	0.683					
Brenda Starr	Papaadopoulou et al.		36	0.575	0.542	0.971	0.696	3	0.664	0.627	0.807	0.706
Harry Friberg	Afsarmanesh et al.		37	0.565	0.537	0.949	0.686					
Robin Scherbatsky	Marx and Akut		38	0.551	0.542	0.662	0.596	25	0.524	0.822	0.062	0.116
Clark Kent	Gupta et al.		39	0.548	0.683	0.178	0.283	26	0.519	0.565	0.170	0.261
Murphy Brown	Sen and Jiang		40	0.529	0.518	0.822	0.635	12	0.623	0.615	0.659	0.636
Peter Parker	Ning et al.		41	0.503	0.502	0.771	0.608					
John King	Bansal et al.		42	0.462	0.460	0.443	0.451					

Overall Insights

The results reveal several insights into the suitability of different features and approaches for the task of hyperpartisan news detection.

Word-embeddings have been reported to be a very efficient feature by many teams. *Tom Jumbo Grumbo* achieved an accuracy of 0.806 with GloVe embeddings and a classifier trained on the by-article dataset. The application of a pre-trained BERT model by *Peter Parker* performed very poorly (accuracy 0.503). However, the same BERT embeddings were used for great

effect by *Sally Smedley*, using techniques like word-dropout and informative phrase identification (accuracy 0.809).

Also standard word n -grams were found to be suitable for the task, though not as strong as embeddings. While n -grams were used in several well-performing approaches, *Pioquinto Manterola* reached an accuracy of 0.704 with unigrams alone.

Several teams reported an increase in accuracy through sentiment or similar features (e.g., *Borat Sagdiyev*). *Kermit the Frog* used sentiment detection alone to reach an accuracy of 0.621.

Besides textual features, a few teams also analyzed HTML and article meta-features. *Borat Sagdiyev* performed a detailed analysis in this regard, which helped them to achieve the highest precision of all teams. For example, they found that both the publication date and the number of links to known hyperpartisan pages could each improve the overall accuracy by about 0.01 to 0.02.

Of the top teams, only *Sally Smedley* used the by-publisher dataset, and only to select n -grams. Based on the reports of several teams, the utilization of this dataset thus seems more difficult than we expected. We conjecture that this is due to the mis-classification of what should be the most informative articles: non-hyperpartisan articles from mainly hyperpartisan publishers, and hyperpartisan articles from non-hyperpartisan publishers. These articles are especially suited to distinguish features that identify hyperpartisanship from features that identify publisher style. While we assumed that the advantages of big data would outweigh this drawback, the results suggest that it might be more worthwhile to put effort in larger datasets where each article is annotated separately. Still, some teams managed to use the by-publisher dataset as a large dataset of in-domain texts. For example, *Clint Buchanan* reported that pre-training embeddings on the by-publisher dataset increased the accuracy of their system on the by-article dataset.

Moreover, the ranking of teams for the two test datasets is quite different. *Bertha von Suttner*, who ranked first for by-article, reached only rank eight for the by-publisher dataset. Conversely, *Tintin*, who optimized for by-publisher, ranked first there but only 27th for the by-article dataset. This discrepancy highlights the unexpected large differences between the datasets.

Meta-Classification

Inspired by successes of meta classifiers in past SemEval tasks (e.g., Hagen et al. [100]), we enabled and encouraged participants to devise meta classifiers that learn from the classifications of the submitted approaches.

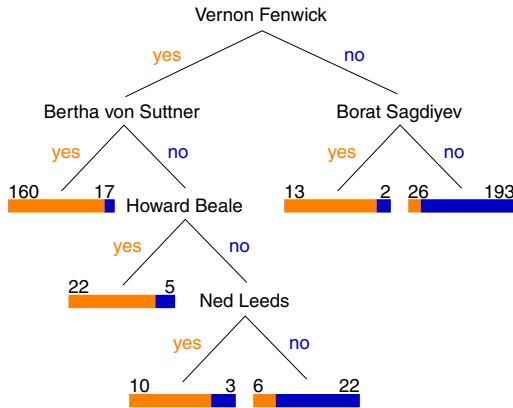


FIGURE 4.8: Meta-classification decision tree J48-M10 learned on the predictions of the submitted systems (hyperpartisan: **yes** or **no**; by-article dataset). The numbers show the training class-distribution at the leafs.

For this meta-classification task, we split the test datasets further into new training (66%) and test sets (33%). We again made sure that there are an equal amount of non-hyperpartisan and hyperpartisan articles, as well as an equal share of left-wing and right-wing articles within the hyperpartisan sets. Furthermore, we again assured that no publisher had articles in both the training and the test sets. An instance in these datasets corresponds to the classifications (hyperpartisan or not) of the best-performing software of each team (42 classifications for the by-article dataset and 30 for the by-publisher one) of one article from the original test data.

We provide two simple classification systems for baselines, majority voting and an out-of-the-box decision tree, which both outperform the best single submitted software and which were both outperformed by the meta-classifiers submitted. Majority voting refers to a system that outputs the classification (hyperpartisan or not) that the most base classifiers selected. As it does not learn a decision boundary, it is—strictly speaking—not a meta classifier. For the decision tree, we used the J48 implementation of WEKA [84]. We tested two variants: standard settings (*J48-M2*) and restricting leaf nodes to contain at least 10 articles (*J48-M10*) to force a simpler decision tree. Simpler trees often generalize better to unseen data.

Figure 4.8 shows the J48-M10 tree for the by-article dataset. For every leaf of the tree, more than 75% of the corresponding training articles are from the same class. This shows that even with as few as 5 decision nodes, the training set could be fitted reasonably well. The meta classifier was thus able to use the submitted systems as predictive and distinct features, which shows that some submitted systems performed well on some articles where

TABLE 4.10: Accuracy, precision, recall, and F_1 -measure for the by-article meta learning test dataset.

Team or system name	Acc.	Prec.	Rec.	F_1
Fernando Pessa	0.899	0.895	0.904	0.900
Spider Jerusalem	0.899	0.903	0.894	0.899
Majority Vote	0.885	0.892	0.875	0.883
J48-M10	0.880	0.916	0.837	0.874
J48-M2	0.856	0.863	0.846	0.854
Bertha von Suttner alone	0.851	0.901	0.788	0.841

other systems did not and vice versa. Even more, the 5 systems employed by the meta-classifier are all within the top 10 systems of the task, which shows that there is considerable variation even among the top performers. This is reasonable, given the variety of approaches used.

In addition to our approaches, two teams submitted their own classifiers in the short time span they had. *Fernando Pessa* used a random forest classifier trained on the single predictions as well as the average vote. *Spider Jerusalem* used a weighted majority voting algorithm, where they weighted each single prediction by the precision of the respective classifier on the training set.

Table 4.10 shows the performance of the approaches on the meta learning test dataset. Note that the best single system, *Bertha von Suttner*, reaches an increased accuracy of 0.851 on the meta learning test set. This is due to variations in the small dataset. Still, all ensemble approaches reach a higher accuracy. The majority voting approach reaches an accuracy of 0.885, and thus outperforms the J48 classifiers. This is somewhat surprising, but shows that there is a lot to gain by integrating also the systems that performed less well—team *Fernando Pessa* came to a similar insight in their paper [62]. The approaches of the two participants performed very similar, despite their methodological differences, and outperformed the majority vote. They managed to achieve an accuracy 0.048 points above *Bertha von Suttner* and therefore a considerable increase in performance.

We also repeated the experiments for the by-publisher dataset part, but could not produce decisive results there. We assume that this is due to most teams focusing on the other dataset part and both parts being more different than expected.

4.2.7 Conclusion

By harnessing web archives we show that news articles conveying a hyperpartisan world view can be distinguished from more balanced news by writing style alone. Moreover, for the first time, we found quantifiable evidence that the writing styles of news of the two opposing orientations are in fact very similar: there appears to be a common writing style of left and right extremism. We further show that satire can be distinguished well from other news, ensuring that humor will not be outcast by fake news detection technology. All of these results offer new, tangible, short-term avenues of development, lest large-scale fact-checking is still far out of reach. Employed as pre-filtering technologies to separate hyperpartisan news from mainstream news, our approach allows for directing the attention of human fact checkers to the most likely sources of fake news.

Moreover, this section reports on the setup, participation, results, and insights gained from the first task in hyperpartisan news detection, hosted as Task 4 at SemEval-2019. We detailed the construction of both a manually annotated dataset of 1,273 articles as well as a large dataset of 754,000 articles, compiled using distant supervision from a large-scale web archive. The section provides a systematic overview of the 34 papers submitted by the participants, insights gathered from single teams, by comparing their approaches, and by an ad-hoc meta classification. Through the use of TIRA [197], we were able to establish a blind evaluation setup, so that future approaches can be compared on same grounds. Moreover, through the use of TIRA we can directly evaluate the submitted approaches on new datasets for hyperpartisan news detection, provided they are formatted like the datasets presented here. Very promising results were achieved during the task, with accuracy values above 80% on a balanced test set—and even up to 90% using meta classification on all submissions. Like in many other NLP tasks, word embeddings could be used to great effect, but hand-crafted features also performed well. The differences between the two employed datasets were larger than anticipated, which suggests a focus on by-article annotations in the future. A larger dataset of this kind will probably assist in improving the accuracy of future models even beyond the already very good level.

It thus seems that hyperpartisan news detection is already sufficiently developed to take the next step and demand human-understandable explanations from the approaches. The most obvious use cases of hyperpartisan news detectors are for filtering articles, which always requires a careful handling to avoid unwarranted censorship. Especially in the current political

climate, it therefore seems necessary that hyperpartisanship detectors not only reach a high accuracy, but also reveal their reasoning. When providing reasoning, it will in turn be essential to provide the respective versions of the articles and other relevant articles as provenance, thereby further increasing the importance of web archives for dealing with hyperpartisan news.

4.3 Summary

This chapter focused on harnessing web archives for a critical assessment of information on the web, addressing two challenges in particular. First, Section 4.1 employed the page-specific archive of Wikipedia to assess the quality of 470 million edits. The proposed method used patterns of collaborative quality assurance for a large-scale assessment, enabling a spatio-temporal analysis of reverted edits that revealed temporal regularities across countries. These regularities can now be used to inform automatic quality assessment methods. Second, Section 4.2 employed classical web archives of news articles to enable technologies for the automatic detection of hyperpartisan news. The section presents two respective datasets, both of which were created using the web archive technology presented in Section 2.2. Classification results on these datasets suggest that a large-scale automatic assessment of hyperpartisanship on the web is possible.²¹

²¹Eleven of the teams who participated in the corresponding shared task also published their code: <https://github.com/hyperpartisan-news-challenge>

5

Harnessing Web Archives for Online Security and Privacy

As discussed in Chapter 1, as the most extensive public sphere of today, the web also comes with dangers to its participants. Online (or cyber) security is about mitigating direct risks and harms like identity theft and software manipulation. Online privacy is about preventing invisible and unwanted tracking and data collection. Both measures are needed to strengthen democracies as they support a free exchange of opinions [157]. In some cases, security and privacy are about an attacker not being able to predict individual behavior. For example, not being able to predict which password someone chooses, the way humans click on a button to certify that they are not bots, or the impossibility to identify someone based on a few answers. In other cases, security and privacy are about protocols and software for communication. Web archives can be harnessed in both cases, for they provide (1) the large-scale human behavioral data that some security and privacy analyses require and (2) a potentially secure and privacy-preserving way to access web content.

However, several challenges remain to fully harness web archives to tackle challenges in security and privacy. As one step forward, this chapter illustrates how web archives can be employed for online security and privacy, providing one example each: a security-related analysis of the randomness of human language in a specific situation and a discussion of private web archiving technology, which allows for privacy-preserving re-finding of information. In the context of security, widespread advice for password generation is to come up with a random new sentence, to

take the first character of each word as the password, and to use the sentence as a mnemonic for the password. However, *how to estimate the security of passwords that were derived from randomly chosen human mnemonics?* Section 5.1 quantifies the corresponding guessing resistance for the first time in terms of standard security measures, using sentences from a large-scale web archive. Employing language modeling techniques on this dataset of 3 billion sentences, the section shows that the security provided by such so-called “mnemonic” passwords is roughly that of a 12–13 sided dice per password character—much less than what the alphabet’s set of 26 characters could offer. In the context of privacy, users often have to forego some web service features to avoid privacy-intrusive tracking. Specifically, this chapter investigates the question of *how to protect privacy when re-finding online information?* Section 5.2 presents private web archives as one method to support re-finding without additional communication to the web server. The section details a possible system setup, contributes a prototypical implementation for practical research, and discusses possibilities, limitations, and further research directions.

5.1 Security Estimate for Mnemonic Passwords

Password authentication is widely accepted, has low technical requirements, and hence is expected to stay as a part of authentication systems [37, 96]. Irrespective their popularity, password authentication has always been criticised for the fact that users tend to choose weak passwords—simply to avoid the extra effort of memorizing strong passwords. To animate users to devise stronger passwords, so-called mnemonic passwords are often recommended, which shall provide both strength and memorability [248].¹ Such advice boils down to the following:

Create a sentence. Memorize it. Concatenate the first characters of each word. Use the string as password.

The strength of mnemonic passwords is based on these three assumptions: First, humans can easily remember their mnemonics, a fact that has been shown within several studies [152, 248]. Second, it is infeasible to guess a mnemonic, even if an adversary was able to generate and test millions of guesses per second. This can be assumed, if the user in fact follows the

¹Also recommended by Google or in the New York Times. Google. Strengthen your account security – Create strong passwords. <https://web.archive.org/web/20190301063516/> <https://safety.google/security/security-tips/>. New York Times. How to Devise Passwords That Drive Hackers Away. <https://perma.cc/SX3F-F6PK>

advice and creates the mnemonic himself instead of picking a famous sentence [152]. Third, and most importantly, the derived passwords inherit most of the guessing difficulty of the mnemonic, so that guessing the password remains infeasible as well. To the best of our knowledge, regarding the last point no results have been published in the relevant literature.

By harnessing the vast amount of human text stored in web archives, this work contributes various new and interesting results on this third assumption. Our approach is to generate passwords from a huge sample of human-generated sentences using a generation rule (a variant of “concatenate the first characters of each word”), estimate the resulting password distribution with language models, and calculate common strength estimates from the distribution. The contributions in detail:

- We collect one of the largest available corpora of human-chosen mnemonics.
- We extract a total of 3.1 billion web sentences from the ClueWeb12 crawl web archive² with a specialized filter algorithm, show that these sentences are more complex than mnemonics using a standard readability score, and take a sample with appropriate sentence complexity.
- We use the corpus of mnemonics to provide evidence that the distributions of the character probabilities which are used by common password strength measures are approximately the same for mnemonics and web sentences (both all and the less complex sample). This allows us to substitute web sentences for mnemonics.
- To model mnemonic password distributions we optimize language models. For this, we introduce position-dependent language models to password modeling, for which we show that they improve the estimation over regular language models.
- Using common password strength measures that cover both online and offline attack scenarios, we compare the strengths of password distributions from all and only the simpler sentences under 18 different password-generation rules.

Our approach comes along with a number of important advantages. It is fully reproducible since it uses a static web archive crawl. It exploits the knowledge of the password generation to its full effect, which makes the strength estimates more reliable compared to estimates obtained from

²<http://lemurproject.org/clueweb09/>

dictionary-based cracking attempts. It causes no privacy concerns since no private authentication data is involved. It allows to precisely compare password generation rules, such as concatenating the words' last characters instead of the first.

5.1.1 Related Work

Different to existing studies we do not analyze a password corpus, but put a well-known³ generation principle for passwords to the test.

Mnemonic password strength analyses have previously focused on cracking them by using dictionary or brute-force attacks [248] or a collection of quotes, lyrics, and similar known phrases [152]. However, these analyses are based on very small sample sizes (see Table 5.1), the results depend largely on the employed cracking dictionaries, and they leave the exact generation process to the participants. Also, the used mnemonics are not available. It is interesting to note that Kuo et al. find that, if not explicitly forbidden, users tend to choose famous sentences as mnemonics, with the—expected—negative impact on security.

Very recently, Yang et al. [251] published a strength analysis on what they call mnemonic-based strategy variants, which are variations of the "*create a sentence*" part of the mnemonic password advice. They find that the security against online attacks can be increased when suggesting to the users to use personalized mnemonics and providing them an example mnemonic and password. In contrast, we analyze the security for different variants of generating the password from the sentence. Furthermore, since we use a much larger sample of passwords, we can also estimate the strength of mnemonic passwords against offline attacks and our estimates against online attacks are more robust.

Password strength analysis in general used way larger password samples (up to 70 million [36]), but do not distinguish between mnemonic passwords and others. Especially interesting is the analysis by Bonneau, who found differences in password strength between different user groups (determined by account settings) [36]. Our current data does not provide this kind of meta information.

An overview of the cracking methods used in these analyses is presented by Dell'Amico et al. [67]. Language models, which we use for our analysis, are also used in password cracking [67, 163, 176, 221]. Presumably, these

³For example in a 2011 survey of 195 university people, about 40% had already used a mnemonic password [149].

TABLE 5.1: Number of mnemonics and passwords in the corpora of this and other studies. For the corpora of this study, the number of passwords is averaged over generation rules.

Corpus	#Mnemonics	#Passwords
Webis-Sentences-17	3,369,618,811	1,381,862,722
Webis-Simple-Sentences-17	471,085,690	234,106,405
Webis-Mnemonics-17	1,048	1,035
Obfuscated Yahoo! passwords [36]	-	70,000,000
Leaked from RockYou (e.g., [243])	-	32,000,000
University passwords [169]	-	44,000
Phished from MySpace (e.g., [67])	-	34,000
Survey by Kelley et al. [121]	-	12,000
Survey by Yang et al. [251]	5,334	6,236
Survey by Kuo et al. [152]	140	290
Creation advised by Yan et al. [248]	97	290
Received from Passware [176]	-	140
Survey by Vu et al. [239]	40	40

password crackers would also benefit from our contribution of position-dependent language models.

Extending the usual mnemonic password advice, Topkara et al. [230] suggest complex generation rules to create passwords very different to the mnemonic. This allows to produce from the same mnemonic somewhat independent passwords with different generation rules, which aims at reducing password-reuse between services. Our estimates could also be calculated for such rules.

The good memorability of human-chosen mnemonics has been shown by previous studies. For example, Yan et al. found that mnemonic passwords are about as memorable as passwords selected freely but with at least one non-letter [248]. As memorability measure, they used the time needed until the passwords—which the 290 participants had to use frequently—are memorized. Random passwords, on the other hand, took about 8 times as long to remember.

A different approach to mnemonic passwords is to generate the mnemonics for the users using either sentence templates and dictionaries [21, 119], linguistic transformations [115], or language models [89]. While this removes the problem of humans choosing weak mnemonics, it is unclear how this changes the memorability compared to human-chosen mnemonics.

5.1.2 Sentence Corpora Acquisition

The analysis of a password advice requires a huge sample of the random element of that advice. In the case of the mnemonic password advice, the random element is the mnemonic. Below we introduce the new Webis-Mnemonics-17 corpus, which now is the largest corpus of human-chosen password mnemonics, but which is still far too small for a well-founded statistical analysis. Hence, using a public web archive, this section introduces also two new corpora of web sentences: the Webis-Sentences-17 corpus, as well as a subset called the Webis-Simple-Sentences-17 corpus whose overall sentence complexity better fits that of password mnemonics. Below we demonstrate that mnemonics and web sentences, though different, are very similar in the distributions of character probabilities which are relevant for estimating the password strength. With this knowledge, we can then estimate the strength of mnemonic passwords using the web sentence corpora.

The Webis-Mnemonics-17 Corpus

With the aid of the crowd-sourcing platform Amazon Mechanical Turk, 1,117 mnemonics were collected in a short survey, each from a different worker. Figure 5.1 shows the study interface. The workers are told to chose a mnemonic and remember it (without writing or copying) while answering password-related multiple-choice questions. The study has been designed to fulfill best practices for Mechanical Turk user studies [141]. For example, encouraging a participation in good faith by disabling copy-and-paste. The workers took on average 3 minutes and 35 seconds to complete the study.⁴

Instead of trying to reproduce the memorability results of previous research, we opted for a shorter study with more participants.

In detail, the workers were asked to “create a new, meaningful, and easily memorable English sentence that no one can guess.” To resemble the advice of choosing the mnemonic related to the web page for which it is used, we randomly showed one topic suggestion (*money, shopping, mail, talking with friends*, or no suggestion) to the workers. The survey interface automatically enforced certain constraints to mirror plausible password requirements: The mnemonic must contain (1) only 7-bit ASCII characters; (2) at least 12 words; (3) at least 9 different words from an English dictionary (to ensure English mnemonics); and (4) no sequence of 6 or more words that also occurs in the Webis-Sentences-17 (detailed below, like a blacklist of known phrases).

⁴The corpus with detailed interaction logs of the workers is available at
<https://webis.de/data.html#webis-mnemonics-17>

(a)

Security Guidance for Step 1

To protect **your security**, it is important that you:

- Do not enter a sentence that is **similar to one you use or will use** as a memory aid!
- Do not enter a sentence that **contains confidential or personally identifiable** information!

I have read and understood the text above

Step 1: Create sentence

Step 2: Memorize sentence and password

Step 3: Four background questions

Step 4: Recall sentence and password

Submit

(b)

Step 1: Create sentence

Create a **new, meaningful, and easily memorizable English sentence** that no one can guess.

Your sentence should be related to **shopping**.

Enter the sentence here. **Do not write down or copy** the sentence.

Next

(c)

Step 2: Memorize sentence and password

The first letter of each word forms the password:

Sentence: *When I walked to the grocery store, there were camels flying overhead*
Password: **wiwttgstwcfo**

Memorize the sentence and password for step 4 or change the sentence

Do not write down or copy sentence or password.

Type in the password:

Next

(d)

Step 3: Four background questions

Has anyone ever tried to explain to you what a good password is?

Yes
 No

Have you ever created a password from a sentence (like above) before this study?

Yes
 No

How many passwords do you have? Please estimate.

0 - 10
 11 - 20
 More than 20

Do you use the same password for different accounts?

Yes
 No

Next

(e)

Step 4: Recall sentence and password

Recall the sentence to the best of your ability:

Recall the password to the best of your ability:

FIGURE 5.1: The HTML interface used to collect the Webis-Mnemonics-17 corpus. (a) Complete interface at the survey start. Participants have to read a security guidance. After that, the steps (b-e) are shown one at a time. (b) Participants have to enter a sentence that fulfills our requirements (automatically checked). (c) Participants see their sentence and the corresponding password and are told to memorize both. They have to type in the password. Should they try to paste the password, the pasting fails and they are told not to do so. They can go back to step 1 to choose another sentence. (d) Participants have to select one option for each question. (e) Participants are asked to recall sentence and password.

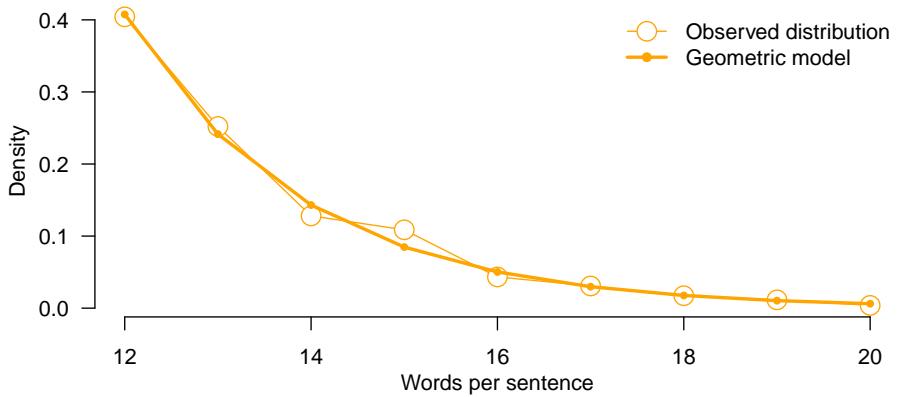


FIGURE 5.2: Distribution of sentence lengths in the Mnemonics Survey corpus and fitted geometric model.

After manual cleaning, 1 048 mnemonics remain. In detail, we rejected 17 workers that submitted grammatically incorrect mnemonics, and filtered mnemonics that were inherently meaningless (10), contained several phrases (40) or a known phrase missed by our filter (1 mnemonic), and where the interface did not record correctly (1 mnemonic). As Figure 5.2 shows, the length of the remaining mnemonics follows a geometric distribution, in concordance with password length distributions in general [163]. Table 5.2 on page 146 gives a few examples from the corpus for each suggested topic.

Despite being one of the largest available corpora of human-chosen mnemonics for password generation, the Webis-Mnemonics-17 corpus is still too small for the calculation of theoretical password strength estimates. Such strength estimates rely on the probability distribution of the passwords, which can not be estimated for corpora of such a small size: Every sentence from the Webis-Mnemonics-17 corpus leads to a different password, making it impossible to infer the probability distribution from the data. Thus, most previous work on strength estimates for mnemonic password strengths [152, 248] were restricted to reporting the percentage of cracked passwords when using cracking software, with the usual drawbacks [36]: results are hard to compare, hard to repeat, and rely on the specific cracking method. For example, because they use different cracking methods, Yan et al. and Kuo et al. come to different conclusions regarding the password strengths. In order to solve these problems, we use a web archive to collect a huge amount of sentences that are sufficiently similar to human-chosen mnemonics like those in the Webis-Mnemonics-17 corpus. We then use these web sentences in place of the mnemonics (see below).

The Webis-Sentences-17 Corpus

To analyze natural language sentences at a huge scale, we specifically designed the new Webis-Sentences-17 corpus,⁵ which is based on the ClueWeb12 web archive.⁶ The ClueWeb12 is a 27.3 TB collection of 733 million English web pages crawled in 2012. It covers authors from a wide range of age, education, and English-speaking countries. The ClueWeb12 is distributed as HTML, necessitating an automatic sentence extraction method.

Since we are interested in content sentences only, we design an automatic extraction algorithm and test it by comparing it to human extraction capabilities. For this purpose, 924 sentences were manually extracted by copy-and-pasting all fitting sentences from 100 random ClueWeb12 web pages. Out of the passwords from automatic sentence extraction, 81% match those from the human extraction.⁷ As we show below, this quality is sufficient for the purposes of the work at hand.

We use an own open source extraction method with optimized parameters:⁸ The method renders the web page text⁹ and removes non-English paragraphs [95], paragraphs with less than 400 characters, sentences with less than 50% letter-only tokens,¹⁰ and sentences without an English function word. We found that some domains use a small set of sentences frequently and filtered such sentences by removing re-occurrences within 1 000 extracted sentences. Further excluding spam pages [58] could not improve the method. We also tried the standard Boilerpipeline ArticleSentence-Extractor [144], but found that it performed worse in our tests.

Exploiting the large size of web archives, the final Webis-Sentences-17 corpus contains 3.4 billion sentences. From these, we generate per generation rule on average 1.4 billion passwords of length 8 to 20. We chose this range based on length limits in popular web pages.¹¹ Table 5.3 on page 147 gives a few examples from this corpus.

The Webis-Simple-Sentences-17 Corpus

As the Webis-Sentences-17 corpus intuitively contains more complex sentences than can be expected for mnemonics, we created the Webis-Simple-

⁵<https://webis.de/data.html#webis-sentences-17>

⁶<http://lemurproject.org/clueweb12/>

⁷Tested for the lowercase letter word initials password generation rule

⁸Source: <https://github.com/webis-de/aitools4-aq-web-page-content-extraction>

⁹Rendering by Jericho HTML: <http://jericho.htmlparser.net> v. 3.2

¹⁰Tokenization by ICU4J: <http://site.icu-project.org/home> v. 53.1

¹¹<https://web.archive.org/web/20140701104040/www.defuse.ca/password-policy-hall-of-shame.htm>

Sentences-17 sub-corpus with a sentence complexity like in the Webis-Mnemonics-17 corpus.¹² For measuring sentence complexity, we use the standard Flesch reading ease test [82] (higher F means more readable):

$$F = 206.835 - 84.6 \cdot \frac{\# \text{syllables}}{\# \text{words}} - 1.015 \cdot \frac{\# \text{words}}{\# \text{sentences}}. \quad (5.1)$$

For the Webis-Simple-Sentences-17 corpus, we sample sentences from the Webis-Sentences-17 corpus such that, for each sentence length, the syllable distribution of the sampled sentences matches the syllable distribution in the Webis-Mnemonics-17 corpus. Since this requires to compare only Flesch values for single sentences of the same length, Equation 5.1 essentially reduces to the number of syllables, where less syllables correspond to simpler sentences. For the sampling probabilities, we fit negative binomial models—which are usual for syllable counts of English sentences [97]—to the observed syllable counts of the Webis-Mnemonics-17 and Webis-Sentences-17 corpora.¹³ Figure 5.3 shows these models. When sampling sentences, the appropriate sampling probability for each sentence length and syllable count follows directly from these models. Also, the Figure shows that the web sentences are indeed significantly more complex than human-chosen mnemonics. Hence, the Webis-Simple-Sentences-17 corpus is more similar to mnemonics than the Webis-Sentences-17 corpus. The final corpus consists of 0.5 billion sentences. From these sentences, we generate on average 0.23 billion passwords of length 8 to 20 per generation rule. Table 5.4 on page 147 gives a few examples from this corpus.

Web Sentence and Mnemonic Similarity

We will now argue why password strength estimates will be approximately the same for passwords from mnemonics and from web sentences. (a) Strength estimates for password distributions depend on the distribution of password probabilities and not on the literal passwords. (b) Password probabilities can be estimated well from a sample using language models, as successfully exploited for password cracking [67, 163, 176, 221]. (c) Language models estimate password probabilities using only the conditional probabilities of the characters given their preceding characters [53]. Hence, given passwords from two different password sources in which

¹²Also available at <https://webis.de/data.html#webis-sentences-17>

¹³As the negative binomial distribution is a discrete distribution, the model for the Webis-Mnemonics-17 syllable counts is first fit to a transformed value of $(\text{syllables-per-word} - 1) \cdot 100$ and then transformed inversely

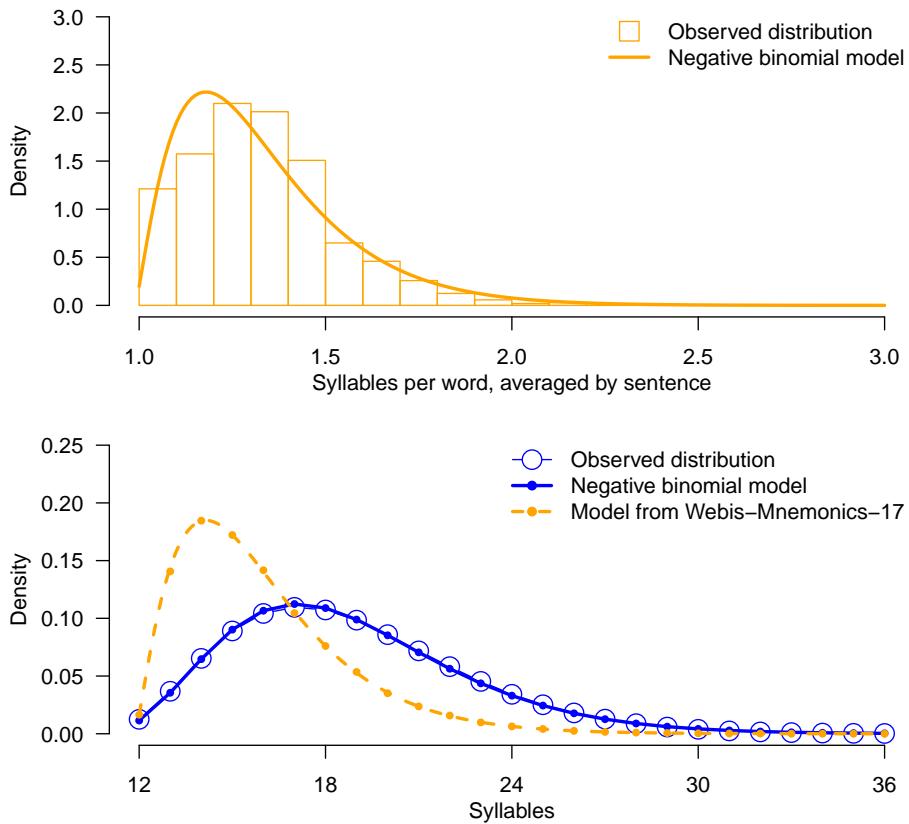


FIGURE 5.3: Distribution and fitted model of syllable counts per word for the Webis-Mnemonics-17 corpus (top) and per sentence for sentences of length 12 in the Webis-Sentences-17 corpus (bottom)

these conditional character probabilities follow approximately the same distributions, the password probabilities of these two sources will also follow approximately the same distribution (from b+c), and the strength estimates will therefore be approximately the same for both password sources (from a). It is important to note that the above reasoning does *not* require that both sources contain the same passwords.

Moreover, algorithmic successes suggest that these conditional character probabilities from mnemonics and web sentences follow approximately the same distributions: (1) Automatic language identification based on related conditional character probabilities works robustly on short texts from various sources [95]; (2) Human-chosen password phrases—a similar setting to that of mnemonics—can be cracked using language models from a few million web sentences [221].

In order to provide further evidence for the similarity, we show that, while complete passwords from mnemonics and web sentences are likely

TABLE 5.2: Example sentences from the Webis-Mnemonics-17 corpus for each of the topic suggestions from the user study.

No suggestion

- What was the color of your car when you were twenty years old?
- The order of my favorite colors followed by my cousin's pets is the password that I use.
- The five green ships docked at the west yellow arrow pointing south.
- i have an upside down kayak that floats on air without wings
- Three birds are sitting on a hibiscus tree driving their cars fast
- my very eager mother just served us pickles, never eat shredded wheat
- My parents are driving here from Michigan to visit for a week.

Your sentence should be related to mail

- beautiful mails require a touch of golden heart and brave minds that also pray
- Savings under the floorboards are safer than inside a big bank vault.
- Boy, you must be Fedex because you look like a hot mail.
- Is it all junk today, or is there anything worthwhile for a change?
- I like talking with my friends about current events and things that will happen in the near time coming.
- i can remember very well what i try to keep as a secret
- I pick up the mail at noon from the mailbox in the lobby of the building
- I want to become a successful teacher as well as a lovable mother

Your sentence should be related to shopping

- While shopping i usually purchase meaningless items that i wrap up in shinny paper.
- when I don't have money I want it, if I have money I want more.
- the cat liked to shop for cookies and bananas at the store in france
- I go shopping in the spring only when it's raining in Paris.
- There is a little girl shopping for a blue dress for her sister.
- When I go shopping, I always buy at least two bunches of bananas.
- Warehouse savings can multiply with money deposited into my account every day.
- My three sons bought the faith of the king with a robe.

Your sentence should be related to money

- Cash is king of the hill and worth every penny and cent.
- The crisp green bill did not leave the frugal boy's pocket until the day he died.
- The community i was born and raised in until I turned legal age.
- I like to bathe in a vat of crisp tens and twenties.
- Just like my inventory in Dragon Age Origins I am hella loaded
- My wife and I are often worried we will have enough money.
- She will get a new apron on her 3rd birthday next year.
- i have huge amount of money and have kept all of my money in savings banks

Your sentence should be related to talking with friends

- How do you know that carrots are good for the eye sight?
 - I told my friend a secret and told her not to tell anyone
 - Hey tell me what friends usually talk when they meet or call?
 - it is important to wash your hands through out the day to keep proper hygiene.
 - I like chat with friends because they are so funny and I am happy I have them.
 - My dear friend how are you and do you know the secret about our teacher mallika
 - Talking to friends can be fun and sometimes we learn new things.
 - My friends make me feel confident about myself and my work skills
-

TABLE 5.3: Example sentences from the Webis-Sentences-17 corpus.

-
- There are also other retail outparcel developments on the other side of the interchange as well as some industrial development in the immediate area, so the center promises to have a strong regional draw.
 - The ADA recommends that the costs associated with postexposure prophylaxis and exposure sequelae be a benefit of Workers' Compensation insurance coverage.
 - Your agents will come away with the knowledge of how service level and quality go hand-in-hand and how that affects the entire contact center.
 - This distance, the 'local loop', helps determine which of the providers in Manhattan will be the best options to provide service to your location.
 - The arena act was the product of gate keeping & was only ever important from a commercial standpoint.
 - And when it comes to painting, throw out your color charts because rural Pennsylvanians use an array of hues not found in nature or in any hardware stores looking to remain on the right side of the Better Business Bureau.
 - Nominations are called for Vice-president and two Director positions on the Board of Directors of ALIA, as incorporated under Corporations Law.
 - The lack of initiative in this case seemed puzzling due to nearly all Americans' faith at the time in the strength and reliability of the constitutional machinery of due process.
 - It will be better for you if you renounce meat & masalas.
 - Shift the focus to sharing; then owning of land loses importance and belonging to earth regains its importance.
 - At that time she was a rapid cycler with full blown manic outbreaks: rages, drinking, drugs, physical reactions, etc.
 - Rather than the usual answer --install a couple of smallish buttons that are sometimes difficult to manipulate - Cat Eye's answer to this in the Strada is to make the entire face "clickable" on this device.
-

TABLE 5.4: Example sentences from the Webis-Simple-Sentences-17 corpus.

-
- Please do not ask to return an item after 7 days of when you received the item.
 - This guide has a lot of nuggets, and I could only stop when I was finished with it.
 - She acted as a student leader during her primary school, high school, college and graduate studies.
 - As mentioned, some gyms also have a daycare program so that you can drop the kids off there while you work out.
 - How much you lose depends on the compression level, but it happens with all saves.
 - So far it looks to top the current king of the hill (Radeon 4870X2) in most but not all benchmarks.
 - Your dog will be well behaved and all your friends will want to know how you did it.
 - And if that is what we want, then talking about "attraction" and "bonding" is a good place to begin.
 - The ramps vary in size and height and you will want to look around to find the best one for your ATV needs.
 - That's blatant right there, you should have seen how wroth Bela Karolyi was about that.
 - Some of the more commonly known herbs to avoid during pregnancy include:
 - Additional cost and energy savings are realized by reducing or eliminating the need for hot water, detergent, labor costs, and capital costs.
 - You can structure it and then restructure it as per your needs.
-

TABLE 5.5: Character-wise cross entropy estimates for passwords from the Webis-Mnemonics-17 corpus of length 12.

Character set	Model corpus	Cross entropy by model order					
		0	1	2	3	4	5
ASCII	Webis-Sentences-17	4.95	4.64	4.58	4.56	4.62	4.75
	Webis-Simple-Sentences-17	4.94	4.63	4.56	4.55	4.62	4.76
	Webis-Mnemonics-17	4.59	4.51	4.54	4.55	4.55	4.55
Lowercase letters	Webis-Sentences-17	4.17	4.11	4.08	4.06	4.07	4.14
	Webis-Simple-Sentences-17	4.16	4.09	4.06	4.04	4.06	4.14
	Webis-Mnemonics-17	4.14	4.10	4.18	4.20	4.20	4.20

different, they are composed from a very similar set of common substrings. This suggests that the difference between mnemonics and web sentences is more of a topical than a linguistic kind, and has therefore not much impact on the strength estimates, if at all. To show that both kind of sentences are composed from a very similar set of common substrings, we compare the cross-entropy—a standard similarity measure of distributions—of different sentence corpora to the Webis-Mnemonics-17 corpus using language models with specific model orders. A model of order o only considers substrings up to $o + 1$ characters. As Table 5.5 shows, the cross entropy from the web sentences corpora to the mnemonic corpus gets about as low as the cross entropy between different subsets of the mnemonic corpus. Therefore, the substrings up to length 4 or 5 in passwords from the web sentences corpora are very similar to those in human-chosen mnemonics.

5.1.3 Password Strength Estimation

Password strength is measured on the password distribution, which is unknown for mnemonic passwords but which can be estimated from huge password samples using language models.

For a formal discussion, this section uses the following notations. X is a random variable distributed over a set of n passwords $\{x_1, \dots, x_n\}$ according to the password distribution \mathcal{X} . We use $p_i = \Pr[X = x_i]$ to denote the probability that a password X drawn from \mathcal{X} is equal to x_i . We enumerate passwords in descending order of their associated probability in \mathcal{X} , that means $p_1 \geq \dots \geq p_n$. Furthermore, $x_i^1 \cdots x_i^{\ell_i}$ denote the ℓ_i characters of password x_i and X^j denotes the random variable of the j -th character of a password. Finally, L denotes a random variable distributed according to the password lengths in \mathcal{X} .

Language Models

Even password corpora several orders of magnitude larger than the Webis-Sentences-17 corpus presented here would not suffice to calculate reliable maximum-likelihood estimates for the probabilities of very rare passwords. The maximum-likelihood estimate of a password probability is the number of its occurrences divided by the size of the entire password sample. However, even in the Webis-Sentences-17 corpus, the often-used Good-Turing method¹⁴ [86] estimates that about 75% of the probability mass of the unknown distribution corresponds to passwords that never occur in the corpus. The maximum-likelihood estimate is thus unsuitable for passwords.

The most widespread language models for passwords, often referred to as Markov chains or n-gram models, employ the chain-rule of probability to describe a password probability by its length and character probabilities.¹⁵ Let the probability of password x_i be

$$p_i = \Pr[L = \ell_i] \cdot \prod_{j=1}^{\ell_i} p_{i,j}, \quad \text{where}$$

$$p_{i,j} = \Pr[X^j = x_i^j \mid X^1 = x_i^1 \dots X^{j-1} = x_i^{j-1}, L = \ell_i]. \quad (5.2)$$

Instead of the exact probabilities in Equation 5.2, language models approximate the character probabilities by conditioning on only the o preceding characters [53], and thus require much less passwords. Therefore, they reduce the model complexity by assuming that

$$x_i^{j-o} \dots x_i^j = x_k^{t-o} \dots x_k^t \quad \rightarrow \quad p_{i,j} = p_{k,t}, \quad (5.3)$$

which leads to robust models used successfully in various natural language tasks [53]. Applying Equation 5.3 to Equation 5.2,

$$p_{i,j} \approx \Pr[X^j = x_i^j \mid X^{j-o} \dots X^{j-1} = x_i^{j-o} \dots x_i^{j-1}, L = \ell_i],$$

where a start-of-password symbol is used for characters preceding x_i^1 :

$$\Pr[X^j = \text{start-of-password symbol}] = \begin{cases} 1 & \text{if } j \leq 0 \\ 0 & \text{if } j > 0 \end{cases}$$

¹⁴The estimate is calculated as the number of passwords occurring only once divided by the number of different passwords in the corpus

¹⁵An alternative is to introduce an end-of-password symbol that is treated like a normal character by the language model [53]. Results are usually similar for both methods [163].

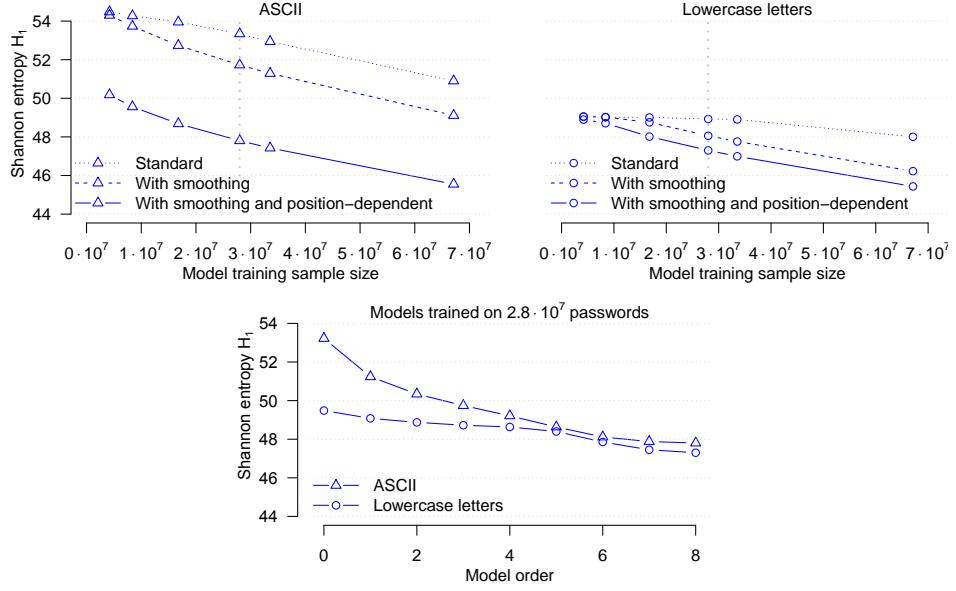


FIGURE 5.4: Effect of the sample size and model order in model training on estimated cross entropy for passwords of length 12 using the ASCII (triangles) and lowercase letters (circles) character sets. The top plots show the effect of the sample size for different model settings and optimal order. The bottom plot shows the effect of the model order for the selected sample size.

Empirical Language Model Optimization

Language models have several parameter, which are commonly optimized for a given task using the cross entropy on an independent password sample [53]. The cross entropy is

$$H_1(\mathcal{X}, \mathcal{X}') = - \sum_{i=1}^n p_i \cdot \log p'_i,$$

where p_i and p'_i are the probabilities of x_i under \mathcal{X} and \mathcal{X}' respectively. In our case, \mathcal{X} is the correct password distribution (approximated by the independent password sample) and \mathcal{X}' is the distribution as estimated by the language model. Note that, when the language model is perfect, that means $\mathcal{X} = \mathcal{X}'$, the cross entropy is minimal and equal to $H_1(\mathcal{X})$. Conversely, because a lower cross entropy corresponds to a better language model, it is safe to optimize language models for cross entropy.

Model order The model order o governs the strength of the assumption in Equation 5.3. For example, $o = \ell_i$ gives the unreliable maximum-likelihood

estimate of password probabilities. On the other hand, $\alpha = 0$ assumes that character probabilities are independent of preceding characters, which leads to robust but heavily biased estimates.¹⁶ In general, the best value for α depends on the amount of passwords in the sample.

Smoothing Smoothing methods use prior-assumptions to improve the unreliable probability estimates for rarely occurring sequences [53]. We use the interpolated Witten-Bell smoothing method [53, 246] for our experiments, which is suggested for character-based models.¹⁷ This method blends unreliable higher-order estimates with more-reliable lower-order ones.

Position-dependency For the special case of password distributions, we propose to use position-dependent language models. Position-dependent models account for the different character distributions at the start, middle, and end of sentences.¹⁸ This is done by estimating the conditional character probabilities for each character position in a password separately. Formally, this corresponds to adding the requirement $j = t$ to Equation 5.3. To the best of our knowledge, we are the first to apply position-dependent models to passwords. As the results below show, position-dependent models are superior for estimating mnemonic password distributions.

Since the different sentence corpora and generation rules lead to password corpora of different sizes, we optimize language models for two scenarios: using all available passwords (for the best strength estimates) and using only a sample of a specific size that is reached by most password corpora (for a fair strength comparison). In order to ensure a safe optimization without overfitting to the data, we create the language models¹⁹ to generate language models and a custom implementation based on KenLM²⁰ from passwords from 19 of the 20 ClueWeb12 parts and evaluate them on the last part that contains mostly web pages from different domains. Therefore, a smaller entropy estimate directly corresponds to a better model. Figure 5.4 (left, center) shows how the entropy estimates decrease with increasing sample size. To ensure a fair comparison between generation rules for

¹⁶This and similar choices between too complex ($\alpha = \ell_i$) and too simple ($\alpha = 0$) are known as bias-variance trade-off in machine learning [35].

¹⁷www.speech.sri.com/projects/srilm/manpages/srilm-faq.7.html

¹⁸For example, in web sentences of length 8, a total of 21% of the first words start with “t”, but only 8% of the last words do so, too.

¹⁹We use SRILM v. 1.7.1 (www.speech.sri.com/projects/srilm/)

²⁰kheafield.com/code/kenlm/) to get probabilities

which we have different sample sizes, we use only $2.8 \cdot 10^7$ passwords per password length and rule when comparing rules. We chose this size so that it is reached for most generation rules.

Furthermore, Figure 5.4 shows that smoothed position-dependent models of the highest order perform best, and we therefore use these models in our experiments. As the Figure demonstrates, position-dependent models are especially advantageous for ASCII passwords, probably due to the included punctuation that occurs mostly as last characters.

Password Distribution Strength Measures

A password-generation rule is stronger when the passwords it generates are more difficult to guess. However, this difficulty depends largely on the knowledge of the guesser. We employ the common Kerckhoffs' principle [122]: since we cannot estimate the knowledge of the adversary, we use the worst-case scenario that she knows the full distribution. Even if the adversary would not know the generation rule, related results suggest that users employ only very few different rules [251]. The adversary tries to guess by choosing one password, verifying it, and repeating to choose and verify until the correct one is found. Since she knows the full password distribution, she guesses passwords ordered by their probability.

We follow related work on password security and distinguish two scenarios: online, where adversaries have a small number of guesses (i.e., until blocked), and offline, where they are limited only by their time [36].

For all the measures detailed below, a higher value corresponds to a stronger password distribution.

Min-entropy The min-entropy models an adversary that takes a single guess for a password [36]. The min-entropy H_∞ is a widespread measure to assess distributions, not only of passwords. It is defined by

$$H_\infty(\mathcal{X}) = -\log p_1$$

Failure probability The failure probability is a measure for the online scenario. The failure probability λ_β reflects the average probability of not guessing a password with β guesses [39].

$$\lambda_\beta(\mathcal{X}) = 1 - \sum_{i=1}^{\beta} p_i$$

We report on $\beta = 10$ and $\beta = 100$ (like [36, 42]).

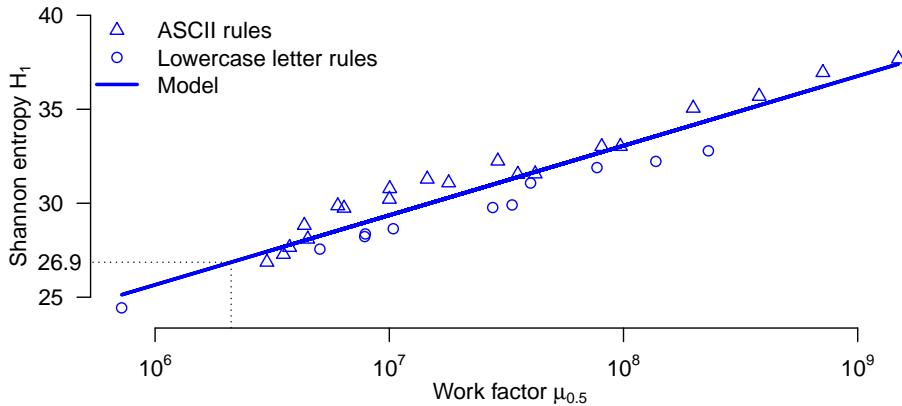


FIGURE 5.5: Scatter plot of strength estimates for different password generation rules and sentence corpora by work-factor (logarithmic scale) and Shannon entropy for passwords of length 8. All language models are trained on $2.8 \cdot 10^7$ passwords. The dotted line shows an estimated $\mu_{0.5}$ for real-world passwords [36] and the corresponding H_1 according to the model.

Work-factor The α -work-factor is a measure for the offline scenario. It models adversaries that guess until they have guessed a fraction α of passwords. The α -work-factor μ_α gives the expected number of guesses [191].

$$\mu_\alpha(\mathcal{X}) = \min \{ \beta \mid 1 - \lambda_\beta(\mathcal{X}) \geq \alpha \}$$

We report on $\alpha = 0.5$ (like [36, 39]).

Shannon entropy The Shannon entropy H_1 measures the bits needed to encode events from a distribution. Unlike the other strength measures, H_1 considers the full distribution. For a uniform distribution, $H_1 = H_\infty$, and $H_1 > H_\infty$ otherwise.

$$H_1(\mathcal{X}) = - \sum_{i=1}^n p_i \cdot \log p_i \quad (5.4)$$

Shannon entropy is usually approximated by the cross entropy on a held-out password sample.

The computational cost of the work-factor $\mu_{0.5}$ makes it infeasible for passwords of length 10 or longer, but we find that it strongly correlates with H_1 in our case (Figure 5.5, Pearson's $r = 0.71$). H_1 has been criticized as a strength measure for password distributions as it does not clearly model the offline scenario [39, 191]. However, due the strong correlation, we see it as a meaningful measure specifically for mnemonic passwords.

5.1.4 Experiments

This section analyzes the strength of mnemonic password distributions. It addresses the following research questions:

- Which of the password generation rules generates the strongest password distribution?
- What effect does sentence complexity have on password distribution strength?
- Does password distribution strength increase linearly with password length?
- Security-wise, how far are mnemonic passwords from uniformly sampled character strings?
- How strong are mnemonic passwords compared to other password approaches?

Estimates by Generation Rules

This experiment compares the strength of password distributions from 18 generation rules in terms of common strength measures. A password generation rule is an algorithm which a human can apply to transform a short text into a password. For this evaluation, we selected rules that vary by the employed character set, replacement rules, and the chosen words from the sentence and characters from the words. The selected rules follow the standard rule of word initials (no replacement, every word, first character) [152, 248] with some variations to test the effect of such variations on the reached security level. If not said otherwise, other experiments use this standard rule. Our implementation of the generation rules is available open source.²¹

Character set The generated passwords consist of either lowercase letters (26 characters) or 7-bit visible ASCII characters (94 characters). Each sentence is processed by a Unicode compatibility and canonical decomposition and stripped of diacritical marks. For lowercase passwords, all letters are converted to lowercase. Then, remaining unfitting characters are removed. Punctuation is treated as an own “word” for ASCII passwords.²² While a larger character set theoretically leads to stronger passwords, especially

²¹<https://github.com/webis-de/password-generation-rules>

²²We use the ICU4J BreakIterator: site.icu-project.org v. 53.1

users of on-screen keyboards are tempted to use only lowercase letters as switching to uppercase or special characters is an extra effort.

Replacement Sometimes the mnemonic password advice includes the replacement of words by similar-sounding characters. To analyze this advice, we include deterministically replacing word prefixes (like “towards” → “2wards”) as a variant.²³

Word We use either every word or every second word in the sentence for generating the password. Theoretically, omitting words increases the difficulty of guessing the next character.

Character position Besides concatenating the first characters, we analyze using the last or both characters as variants. For one-character words, all three variants use this character once.

Table 5.6 shows the estimated strength measures for passwords of length 12 from the 18 employed generation rules. The discussion below focuses on the results for the Webis-Sentences-17 corpus. While mnemonic password distributions in the real world contain passwords from different lengths, we restrict the analysis here to passwords from one length in order to make the comparison easier to understand, as it removes the influence of the length distribution. Especially generation rules that use two characters per word have very different length distributions. Strength estimates based on a natural distribution of password lengths are discussed from Section 5.1.4 onwards.

For a fair comparison, we use the same number of passwords for all estimates, and mark estimates for rules for which our data has less passwords in gray. These estimates in gray are less reliable and biased to higher values for H_1 .

For the online scenario measures min-entropy H_∞ and failure probability λ_β , comparable strengths are achieved by all generation rules but those that use multiple characters and every word, which are considerably weaker. For H_∞ , a further factor is the character set where ASCII has about 1 bit advantage. For λ_{100} , generation rules that use every second word are stronger than other rules.

²³The employed replacements are based on a list of “pronunciation rules” with the two additional rules of “to” → “2” and “for” → “4”: Coding Horror. ASCII Pronunciation Rules for Programmers. <https://perma.cc/GBC3-K4SJ>

TABLE 5.6: Min-entropy (H_∞), failure probability (λ_β), and Shannon entropy (H_1) for different password-generation rules, sorted by H_1 . The values are for passwords of length 12 from the Webis-Sentences-17 (WS) and Webis-Simple-Sentences-17 (WWS) corpora with models from at most $2.8 \cdot 10^7$ passwords. Values from fewer than $2.8 \cdot 10^7$ passwords are shown gray.

Password generation rule				H_∞				λ_{10}				λ_{100}				H_1	
Character set	Replacement	Word	Char. pos.	WS	WWS	WS	WWS	WS	WWS	WS	WWS	WS	WWS	WS	WWS	WS	WWS
ASCII	✓	every 2nd	1st	13.8	13.2	0.99958	0.99940	0.99827	0.99753	56.7	55.8						
ASCII	-	every 2nd	1st	13.8	13.2	0.99959	0.99940	0.99827	0.99752	53.6	52.9						
ASCII	✓	every	1st	13.8	12.5	0.99949	0.99925	0.99760	0.99689	49.9	48.0						
ASCII	✓	every 2nd	last	13.8	13.3	0.99960	0.99939	0.99825	0.99745	49.8	50.0						
Lowercase letters	-	every 2nd	1st	13.1	12.8	0.99956	0.99938	0.99840	0.99739	48.5	48.1						
ASCII	-	every	1st	13.8	12.4	0.99948	0.99925	0.99759	0.99688	47.8	46.2						
ASCII	-	every 2nd	last	13.9	13.3	0.99960	0.99939	0.99824	0.99744	47.6	47.8						
Lowercase letters	-	every	1st	11.4	12.8	0.99912	0.99928	0.99738	0.99739	47.3	45.7						
ASCII	✓	every 2nd	1st+last	12.4	12.8	0.99940	0.99925	0.99775	0.99724	46.5	44.9						
Lowercase letters	-	every 2nd	last	13.1	12.8	0.99955	0.99938	0.99833	0.99735	44.6	44.7						
ASCII	-	every 2nd	1st+last	13.1	12.5	0.99948	0.99929	0.99767	0.99725	44.6	43.0						
ASCII	✓	every	last	14.0	12.4	0.99951	0.99925	0.99759	0.99690	43.4	42.6						
Lowercase letters	-	every	last	11.4	12.8	0.99912	0.99928	0.99734	0.99738	42.7	41.8						
Lowercase letters	-	every 2nd	1st+last	12.0	13.3	0.99933	0.99941	0.99803	0.99785	42.6	41.2						
ASCII	-	every	last	14.0	12.5	0.99950	0.99925	0.99757	0.99689	42.0	41.3						
Lowercase letters	-	every	1st+last	10.3	9.6	0.99708	0.99650	0.99225	0.99098	36.8	35.3						
ASCII	✓	every	1st+last	10.8	9.7	0.99772	0.99715	0.99108	0.98826	35.8	36.1						
ASCII	-	every	1st+last	8.5	11.5	0.99400	0.99775	0.98634	0.98936	34.8	35.2						

For the offline scenario measure H_1 , passwords from ASCII achieve a similar strength to passwords with only lowercase letters when every word is used, but better strength when every second word is used. In total, using every second word and only the first character with the ASCII character set leads to the strongest of the tested password distributions. Also, word prefix replacements can increase the entropy by 2–3 bit. Moreover, using the first character of a word is preferable.

The strongest distribution is arguably using the ASCII character set, every second word, and only the first characters, which achieves best or nearly-best values for all measures. Word prefix replacement considerably increase the strength for H_1 , but not for the online scenario. However, both using only every second word and word prefix replacements come with additional memorization and processing costs, a discussion of which lies outside the scope of this publication.

Estimates by Sentence Complexity

Table 5.6 also shows that strength estimates for the Webis-Simple-Sentences-17 corpus are most times a bit weaker, but still very similar, to those from the Webis-Sentences-17 corpus for all distributions with sufficient training passwords. The maximum difference for one generation rule between the corpora are 1.6 bit for H_∞ , 0.00026 for λ_{10} , 0.00071 for λ_{100} , and 1.9 bit for H_1 . This corresponds to a large difference for H_∞ and a still noticeable difference for H_1 , but smaller than one could expect.

Therefore, mnemonics with lower complexity do indeed lead to passwords that are easier to guess. This is likely due to the reduced vocabulary of the mnemonics, which is biased towards words with less syllables. The effect of mnemonic complexity is especially strong for the min-entropy H_∞ , which considers the most probable password only. A possible explanation is that the most probable password stems from simple sentences, even for the Webis-Sentences-17 corpus. Then, the probability of this password increases naturally when more complex sentences are filtered out. On the other hand, the effect of mnemonic complexity is still noticeable for the Shannon entropy H_1 , which considers the entire password distribution. Therefore, reducing the complexity skews the entire password distribution farther away from the uniform distribution. However, the effect is much weaker than for min-entropy. An estimate of the effect could be the maximum difference in Table 5.6 between Webis-Sentences-17 and Webis-Simple-Sentences-17 for generation rules with sufficient training passwords, divided by the password size: 0.16 bit per character.

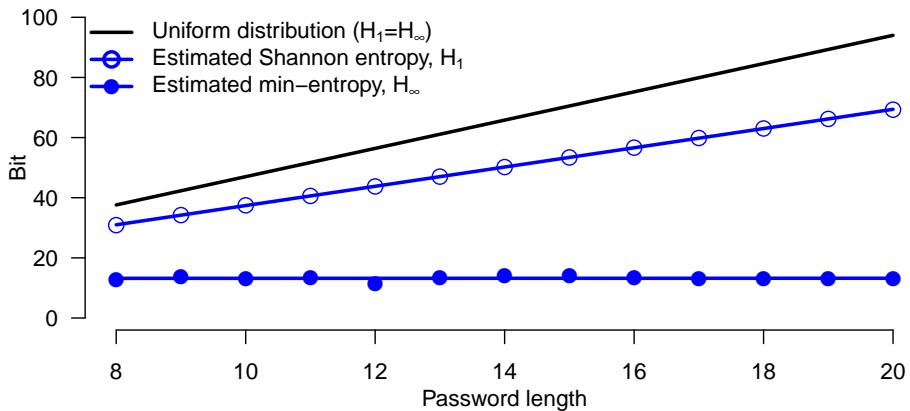


FIGURE 5.6: Shannon entropy and min-entropy estimates compared to the optimal uniform password distribution by password length. Passwords are from the Webis-Sentences-17 corpus using lowercase letters and the first character.

Estimates by Password Length

This section analyzes how the strength of password distributions increases with password length. The number of possible passwords increases exponentially with the password length, theoretically leading to stronger password distributions. Using the Webis-Sentences-17 corpus, we analyzed all rules to very similar results. As an example, Figure 5.6 shows the result for the standard generation rule using lowercase letters only.

Figure 5.6 shows that the resistance against offline attacks (H_1) increases as expected with password length, but that the resistance against online attacks (H_∞) stays rather constant.²⁴ We also found λ_{10} and λ_{100} to be rather constant.

The approximately constant resistance against online attacks shown in Figure 5.6 suggests that, for each length, there are a few sentences with a high probability irrespective the length. Only after these high-probability sentences, a spreading of the probability mass over the possible sentences occurs. This spreading is shown by the steady increase of the Shannon entropy. Unfortunately, the Webis-Mnemonics-17 corpus is far too small to reproduce this effect on human-chosen mnemonics. It thus remains unclear to which extent this effect also appears for human-chosen mnemonics. However, based on our analysis it is reasonable to assume that the resistance against online attacks of mnemonic passwords grows way less with password length than one would expect.

²⁴ H_∞ varies between 11.3 and 14.2 bit without a clear direction.

The linear increase of the Shannon entropy with password length leads to a simple model for estimating the entropy of password distributions with several lengths. In detail, one can rewrite Equation 5.4 (Shannon entropy) as

$$H_1(\mathcal{X}) = \sum_{\ell=\ell_{\min}}^{\ell_{\max}} \Pr[L = \ell] \cdot (H_1(\mathcal{X}_\ell) - \log \Pr[L = \ell]), \quad (5.5)$$

where $H_1(\mathcal{X}_\ell)$ is the entropy estimate for length ℓ . Moreover, for the probability of a password-length, $\Pr[L = \ell]$, one can use the geometric model of lengths from the Mnemonic Survey corpus (Figure 5.2).²⁵ Due to the geometric model and only a linear increase of the entropy by length, Equation 5.5 converges as ℓ_{\max} increases. We report the converged values in the following. The remaining parameter is the minimum password length ℓ_{\min} , which one can increase to increase the password distribution strength, as it is best practice for password-based authentication in general [79]. Using the mean from the fitted geometric distribution, the average password length is $\ell_{\min} + 1.4$ for passwords that take every word and $\ell_{\min} + 0.7$ for passwords that take every second word, while the mode is ℓ_{\min} in both cases. Note that this consideration makes the simplifying assumption that the parameter of the geometric distribution does not depend on ℓ_{\min} .

Table 5.7 shows the minimum-length based entropy estimates for a selection of the strongest generation rules. This table aims at replacing for mnemonic passwords the “rules of thumb” that exist for the entropy of generic passwords (e.g., [42]). Unlike these rules of thumb, which were shown to not correlate with the password distribution strength against offline attacks [243], we have shown that our entropy estimates do correlate with it (cf. Figure 5.5). As the Table shows, when considering that rules using only every second word lead to shorter passwords on average, these rules lose much of their advantage, and are even weaker for lowercase letter passwords.

Comparison to Uniform Distribution

The uniform password distribution is the strongest among all distributions with the same number of elements, but mnemonic password distributions fall short of it for three reasons: (1) some characters occur more frequently than others, (2) characters in a password are not independent of each other,

²⁵When only every second word is used, the length distribution can be adjusted accordingly. However, an adjustment is not as straight-forward when two characters per word are used, due to one-character words. As this variant gave very weak distributions, we do not consider it here.

TABLE 5.7: Estimated entropy by generation rule and minimum password length for passwords from the Webis-Sentences-17 corpus.

Character set Replacement Word Character position	Lowercase letters -	ASCII every 1st	Lowercase letters every 2nd 1st	ASCII every 2nd 1st	ASCII every 2nd 1st
ℓ_{\min}	Shannon entropy H_1				
8	38.0	37.9	34.8	37.2	39.0
9	41.2	41.3	38.0	40.9	42.9
10	44.4	44.8	41.2	44.6	46.7
11	47.6	48.3	44.4	48.3	50.6
12	50.8	51.8	47.6	52.0	54.5
13	54.0	55.2	50.8	55.7	58.4
14	57.2	58.7	54.0	59.4	62.3
15	60.4	62.2	57.2	63.1	66.2
16	63.6	65.7	60.4	66.8	70.1
17	66.8	69.1	63.6	70.5	74.0
18	70.1	72.6	66.8	74.2	77.9
19	73.3	76.1	70.0	77.9	81.8
20	76.5	79.6	73.2	81.6	85.7
21	79.7	83.0	76.4	85.3	89.5
22	82.9	86.5	79.6	89.0	93.4
23	86.1	90.0	82.8	92.7	97.3
24	89.3	93.5	86.0	96.4	101.2
25	92.5	97.0	89.2	100.1	105.1
26	95.7	100.4	92.4	103.8	109.0
27	98.9	103.9	95.6	107.5	112.9
28	102.1	107.4	98.8	111.2	116.8
29	105.3	110.9	102.0	114.9	120.7
30	108.5	114.3	105.2	118.6	124.6

and (3) the character distributions depend on the position in the password. Table 5.8 illustrates exploiting these 3 effects step by step for the standard mnemonic passwords. In addition to the Shannon entropy H_1 , the table also shows the perplexity $Ppl. = \log(H_1)$ which gives the number of elements in a uniform distribution with the same entropy.

According to the results shown in Table 5.8, both password distributions provide in an offline scenario about the same level of security as a uniform distribution over 12 to 13 characters. The biggest effect is in both cases that the characters are not uniformly distributed. On the other hand, exploiting

TABLE 5.8: Character-wise entropy (H_1) and perplexity (Ppl.) estimates for passwords by model. Passwords are of length 12 from the Webis-Sentences-17 corpus using the first character of every word. The uniform model represents the optimal distribution over 26/94 characters.

Model	Lowercase letters		ASCII	
	H_1	Ppl.	H_1	Ppl.
Uniform	4.70	26.0	6.55	94.0
Order 0	4.15	17.8	5.09	34.1
Order 8	3.71	13.1	3.98	15.8
Order 8, position-dependent	3.65	12.6	3.70	13.0

the differences in the character distributions by position (using position-dependent models) is especially valuable for ASCII passwords, where it can nearly reduce their strength to the strength of lowercase letter passwords. Like discussed, ASCII passwords are only stronger than lowercase letter passwords for specific generation rules.

Comparison to Dictionary Passwords

While the discussion in the following paragraphs exemplifies how our strength estimates can be used to compare the strength of different password generation methods, it does not incorporate other important factors for password usage like memorability, typing convenience, or susceptibility to typing errors. Unfortunately, we are not aware of any such comparison.

A second prominent suggestion for password generation is to pick several words uniformly at random from a large dictionary.²⁶ We use the 7776 words Diceware dictionary²⁷ as an example.

A computation of the strength of such dictionary-based passwords is straight-forward. The min-entropy and Shannon entropy are both equal to

$$H_\infty = H_1 = n \cdot \log 7776 ,$$

whereas the failure probability is calculated by

$$\lambda_\beta = 1 - \frac{\beta}{7776^n} .$$

²⁶Popularized in web comics like this one: XKCD. Password Strength. <https://perma.cc/RS5F-5LF4>

²⁷A. G. Reinhold. The Diceware Passphrase Home Page. <https://perma.cc/UNR8-BP6Z> (list: <https://perma.cc/5SD8-7R62>)

The maximum length of a word in the dictionary is 6 characters. On average, the created passwords have a length of $n \cdot 4.24 + (n - 1)$, where $(n - 1)$ is then number of space characters as illustrated on the Diceware homepage.

The comparison with Diceware passwords highlights the relative weakness of mnemonic passwords against online attacks: already the two-word Diceware distribution achieves a failure probability λ_{100} of 0.999998 and is thus considerably stronger in this scenario than every rule we considered and requires on average only 9.5 characters.

However, mnemonic passwords provide a better security in the offline scenario for the same password length. For example, 3 Diceware words (average password length of 14.7) achieve 38.8 bit of Shannon entropy, which is already reached by lowercase letter mnemonic passwords of minimum length 9 (Table 5.7, average length of 10.4).

Comparison to Real-world Password Distributions

This section compares the strength estimates for mnemonic passwords with estimates for real-world password distributions from the literature.

The currently largest-scale password strength analysis of real-world passwords is the one of 70 million anonymized Yahoo! passwords by Bonneau [36], which results in the following estimates: Min-entropy $H_\infty \approx 6.5$, failure probability with 10 guesses $\lambda_{10} \approx 0.98178$, and a work factor $\mu_{0.5} \approx 2,111,739$.²⁸ While no estimate for the Shannon Entropy H_1 is provided, we can apply the log-linear relationship of $\mu_{0.5}$ and H_1 that we observed for mnemonic passwords, which suggests an H_1 of ~27 Bit (cf. Figure 5.5). Bonneau also compares the Yahoo! estimates to estimates from the password lists leaked from the RockYou and Battlefield Heroes websites. He finds that the corresponding two password distributions are even weaker against offline attacks. Also, only the Battlefield Heroes passwords are stronger against online attacks ($H_\infty \approx 7.7$, $\lambda_{10} \approx 0.98878$).

Comparing these estimates for real-world password distribution with our estimates for mnemonic password, we see that mnemonic passwords are considerably stronger against both online and offline attacks. For online attacks, our estimates for the standard lowercase letters word initial rule are for H_∞ between 11.4 and 12.8, and for λ_{10} between 0.99912 and 0.99928 (Table 5.6)—reducing the corresponding success probability $(1 - \lambda_{10})$ com-

²⁸The paper provides normalized estimates, which all use a common scale. The estimates we report are un-normalized.

pared to the Battlefield Heroes passwords by 92–94%.²⁹ For offline attacks, we can extend Table 5.7 for smaller ℓ_{\min} , suggesting that a higher H_1 as for real-world passwords is reached by mnemonic passwords from the standard rule with a minimum length ℓ_{\min} of 5. While the length distribution of the Yahoo! passwords is unknown, the current minimum password length for new Yahoo! accounts is 8 and thus considerably larger. Therefore, we can conclude that mnemonic passwords are stronger against both online and offline attacks compared to the passwords in use today.

5.1.5 Conclusion

This work analyzes the strength of passwords generated according to the mnemonic password advice on a huge corpus of 3 billion human-written sentences, extracted from a public web archive. Our detailed analysis considers sentence complexity and 18 different password generation rules. To this end, we show that the necessary similarity of human-chosen mnemonics and web sentences exists. Furthermore, we contribute one of the currently biggest corpora of human-chosen mnemonics. Additionally, this work is the first to apply position-dependent language models to passwords, which improve on regular language models for modeling mnemonic passwords.

Our analysis addressed several questions on the strength of mnemonic passwords.

Of the 18 tested password generation rules, the strongest password distribution is generated by using the ASCII character set, concatenating the first character of every second word, where common word prefix replacements are used to add more special characters to the passwords. Both using only every second word and word prefix replacements have only an effect in offline attack scenarios, where adversaries are not limited by a number of guesses but by the time they want to invest.

The sentence complexity of the used mnemonics has a major effect when the adversary can perform only a few guesses, and a relatively weak effect for offline attacks.

We showed that an attacker can use knowledge on the generation process of mnemonic passwords to drastically increase his success chances, reducing the strength of mnemonic passwords against offline attacks to that of passwords from a uniform distribution over only 12 to 13 characters.

²⁹When known phrases are allowed as mnemonics, related results suggest a similar strength against online attacks as the Battlefield Heroes passwords have [251], which highlights the importance of developing password-blacklists to keep users from choosing such easy-to-guess phrases.

We analyzed the effect of password length on the strength estimates, and found that—as one would expect—the strength of mnemonic passwords against offline attacks grows linearly with the password length. On the other hand, if the adversary can only perform a few guesses, our results suggest that longer passwords provide no further advantage.

Using statistical modeling, Table 5.7 provides detailed estimates of the strength of mnemonic passwords against offline attacks for different minimum password lengths and password generation rules. This table aims to replace for mnemonic passwords the inaccurate “rule of thumb” for strength calculation that was used previously. With this table, we compare mnemonic passwords to a password generation approach that performs repeated uniform sampling from a dictionary and found that mnemonic passwords are weaker against online, but stronger against offline attacks.

The analysis of the password generation rules is limited to the strength of the corresponding password distributions and ignores that the different rules are associated with different costs for the human. For example, the best generation rule requires the human to memorize a twice as long sentence. Furthermore, already having a certain generation rule in mind will likely have an influence on mnemonic choice. For instance, if the human wants to use a rule that incorporates word prefix replacements, he may limit the considered mnemonics to such where he can actually perform a replacement operation. A more detailed study on memorability and mnemonic choice would be needed to improve this discussion.

Furthermore, this analysis is restricted to English mnemonics only, raising the question whether our results also transfer to other languages.

An interesting avenue for further research could be to use search algorithms to find the best password generation rule for a given sentence distribution. The 18 rules that we analyzed cover only a very small part of the parameter space for such rules. Investigations in this direction would require lowering the computational cost of evaluating a rule. Moreover, an analysis of the costs of generation rule parameters like suggested above could also be integrated into the cost function of the search algorithm.

Finally, the shown use of web data as a substitute for human behavior raises the question which other forms of behavior could be substituted by web data. The analysis in this work focused merely on lexical properties of textual paragraphs. However, also the text content, images, or page interactivity (how the page reacts to certain input) provides data on both the page’s author and typical visitors. As illustrated here, web archives are key for such endeavors, as they provide such data at a large scale, allowing to catch generic behavioral patterns like in this work.

5.2 WASP for Privacy-Aware Information Re-Finding

Lifelogging³⁰ has become a common practice, as a result of the omnipresence of smartphones, smart watches and fitness trackers, and emerging technologies such as smart glasses, wearable technologies and sensor-enabled smart homes. Isn't it surprising that keeping track of one's online activities is comparably underdeveloped? To the contrary, people are increasingly seeing being tracked online as something to avoid. Significant amount of work has been invested into understanding personal information management [117] and developing tools to support it, including the winner of the SIGIR 2014 Test of Time Award "Stuff I've Seen" by Dumais et al. [73]. With a bit of irony however, neither Stuff I've Seen nor follow-up Phlat [64] are available today, even if the key insights gained have likely informed the development of Windows desktop search and intelligent assistant Cortana. Likewise, Spotlight on MacOS supports search over local documents and other digital assets. Both are integrated with the web browsers from Microsoft and Apple, respectively, to index browsing history. Meanwhile, the history tabs of modern Web browsers provide access to the history of the currently open browser as well as pages recently visited on other devices. However, current browsers do not align and integrate the browsing histories across devices, nor, apparently, do the aforementioned tools index the content of web pages visited, but only their titles and URLs. In fact, the possibility to track (let alone search) one's browsing history using off-the-shelf tools is still fairly limited. As a result, several web services offer functionality to re-find information they provided to the same user in the past. However, such functionality requires the web server to collect user data, which is seen as problematic by the public, as indicated, for example, by the recent European GDPR privacy protection law.

In this context, it is not surprising that personal information access was one of the major topics discussed at the Third Strategic Workshop on Information Retrieval in Lorne (SWIRL 2018) [63]. The attendees noted that this problem, open for so long, has not been addressed adequately, and, worse, that it is an ever more daunting challenge to help people re-find and revisit their online information and prior information interactions with these sources; as this information today resides in multiple devices and a large variety of information services, that each construct their own data silos and search APIs (if such access is offered at all). Specifically, the report mentions the high cost of entry for scientists as a major obstacle, where "there is substantial engineering required for a minimal working system: to fetch

³⁰<https://en.wikipedia.org/wiki/Lifelog>

data from different silos, parse different data formats, and monitor user activity.”

We propose to take a pragmatic “shortcut” and to establish empirically how far that workaround can bring us. Increasingly, access to our digital information takes place through the web browser as *the* interface. Therefore, we set out to develop WASP (Web Archiving and Search, Personalized), a prototype system for *personal* and *private* web archiving and search. WASP saves one’s personal web browsing history using state-of-the-art web archiving technology and offers a powerful retrieval interface over that history. This browser-focused setup enables the user to recall information they personally gathered without the need to deal with the large variety of information sources, nor the need to enable privacy-averse tracking mechanisms in the employed web services. Even if we do not cover the full range of digital objects that may accrue on a person’s desktop and mobile devices, high-quality archival of web pages visited may capture a large fraction of the information we interact with.

In addition to a detailed technical description of WASP, this work reports on the observations that we made and the challenges for personal web archiving and search that we identified through our extensive use of the WASP prototype—which we provide both open source and as an executable Docker container so that others can use it within their research or personal lifelogging setup.^{31,32}

5.2.1 Related Work

WASP is directly related to prior work on desktop search, including the already mentioned Stuff I’ve Seen [73]. However, apart from not indexing all documents that may exist on a desktop, the intended usage differs slightly as well: WASP aims to track everything a user has seen, as they saw it, and in that sense provides some notion of versioning. While not yet implemented, a future version should explore the functionality once implemented in diffIE, i.e., to rank pages that evolved differently from static ones, and this way provide immediate insight in changes of the web over time [229].

WASP is also related to search tools for web archives, such as ArchiveSpark [108]. However, due to handling a single user’s view of the online world only, the system aspects to be addressed include less emphasis on scalability. Developments in the UI/UX of web archive search are, however, likely transferable, in both directions—as argued in this thesis, what

³¹<https://github.com/webis-de/wasp>

³²<https://hub.docker.com/r/webis/wasp/>

we learn from observing interactions with personal web archives may very well carry over to the large web archives of interest to Digital Humanities researchers [30].

We find that a new blend of techniques that have been proposed previously will be necessary to design the right user experience, and we realize that we have only scratched the surface so far. For example, searching the social web is different from searching the web, as shown convincingly in [11]. We also highlight the immediate relevance of research into focused retrieval carried out in context of INEX. The question of how to determine a retrieval unit has clearly not been solved, yet, and the usage scenario of personalized web archive search that we envision has increased the urgency to revisit that line of research.

5.2.2 The WASP Prototype

The WASP³³ prototype integrates existing archiving, indexing, and reproduction technology for the first time into a single application that allows for privacy-aware information re-finding. Figure 5.7 illustrates how the user's browser interacts through WASP with the web under the three usage scenarios detailed below: web page archival, search, and reproduction.

Archiving Proxy and Indexing

After starting WASP, the user has to reconfigure their browser to accept WASP as forward proxy and to trust its certificate. WASP then archives all HTTP(S) requests and responses from and to the browser in the standard Web archiving format (WARC) (Figure 5.7 (a)). This is achieved using the Internet Archive's warcprox software,³⁴ whose WARC contain all the information necessary to reproduce an archived browsing session at a later time.

In order to enable searching the archived content, we devised a software component that monitors WARC files and automatically indexes HTML responses and their corresponding requests in an ElasticSearch index.³⁵ In detail, we use the Lemur project's WARC parser³⁶ and Apache's HttpClient library³⁷ to read HTTP messages as they are appended to the WARC files. The title and text of the HTTP responses that have the MIME type HTML are extracted from responses using the Jericho HTML Parser library.³⁸ The

³³WASP is short for Web Archiving and Search, Personalized

³⁴<https://github.com/internetarchive/warcprox>

³⁵<https://www.elastic.co/>

³⁶Lemur Project. Working with WARC Files. <https://perma.cc/UV7A-NY5L>

³⁷<https://hc.apache.org/httpcomponents-client-ga/>

³⁸<http://jericho.htmlparser.net/docs/index.html>

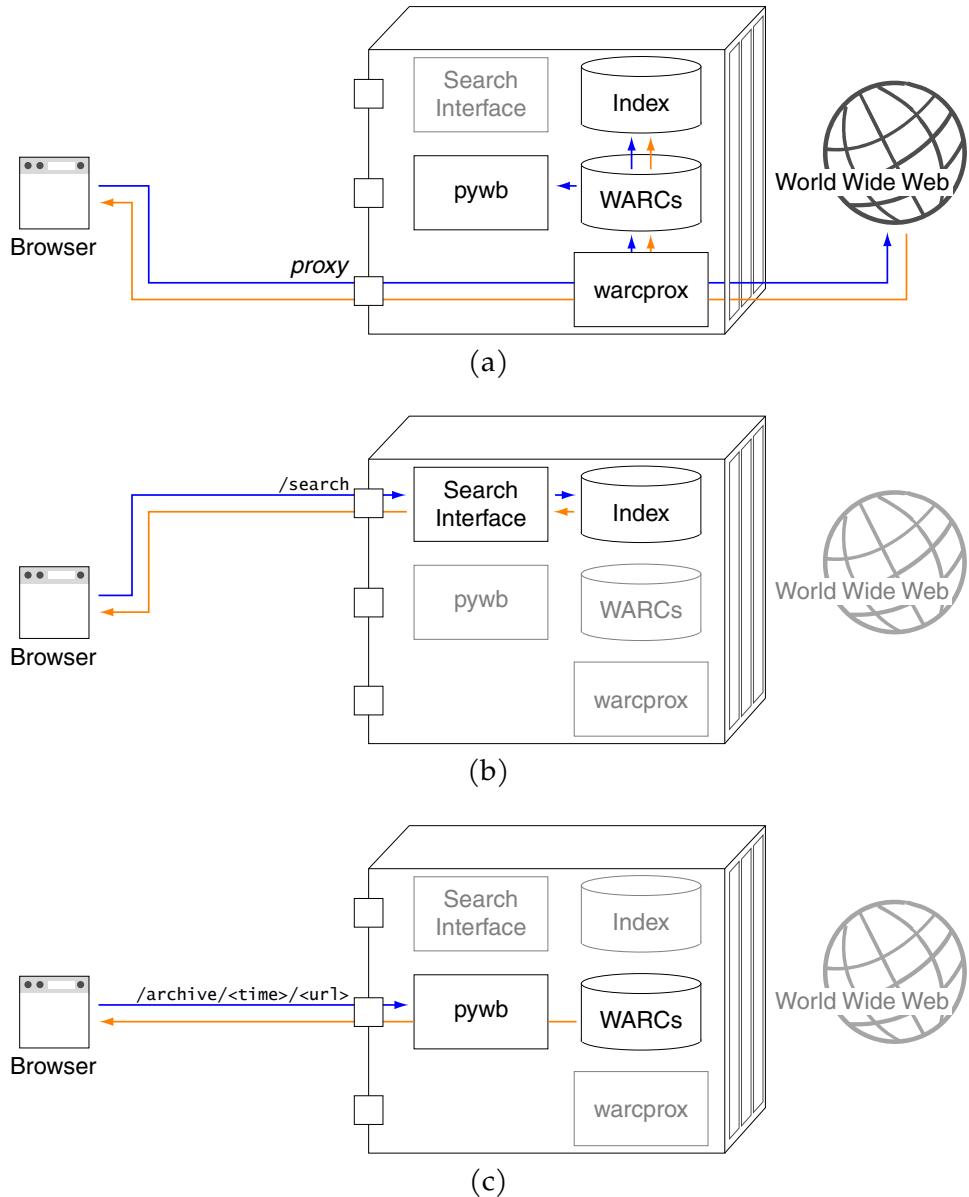


FIGURE 5.7: Architecture of the prototype: (a) during regular browsing, the container works as a forward proxy that stores all requests (\rightarrow) and responses (\leftarrow) in web archive files (WARCs) and indexes them; (b) when browsing to `localhost:<search-port>/search`, the browser shows our search interface (Figure 5.8), where results link to (c) the reproduction server, which serves content from the WARC that (fuzzy) matches a specific time and URL.

title and text of the HTTP response is indexed along with the corresponding HTTP request's time and URL. Later page revisits (identified by `warcprox` through hash value matching on HTTP responses) are added to the response's record in the index. When the index is queried, the aggregation of requests avoids duplicate results in case a page is visited more than once.

Even if web pages vanish or change, WASP can reproduce the content the user saw in the past using the Web archiving toolkit `pywb`.³⁹ Like our automatic indexing setup described above, `pywb` monitors and indexes changes to the web archives. While the ElasticSearch index is tailored toward search within the HTML content of the archived web pages, the `pywb` index is tailored towards retrieving the HTTP response corresponding to a given HTTP request, enabling efficient reproduction of pages from the personal archive.

Search Interface

Access to the archived web pages is provided using the ElasticSearch index detailed above (Figure 5.7 (b)). Under a configurable port, WASP provides the user with a basic search engine. Figure 5.8 shows a screenshot of the interface. Unlike regular web search engines, WASP's interface provides controls to specify the time the user recall visiting the desired web page ①, ②, ③⁴⁰ in addition to the familiar query box ④. Web pages are retrieved by matching query words against the title and contents of web pages visited in the specified time interval. ElasticSearch's highlight feature is used to generate query-related snippets for the results ⑨.

A difference to a regular search engine results page is that in WASP, each result item consists of two hyperlinks: one resolving the URL to the live web ⑥ as usual, and another one pointing to the archived version of the web page. This latter hyperlink refers to the port of the WASP container's reproduction proxy and the access time and URL of the web page that should be reproduced. In case several non-identical versions of the same page are found in the requested interval, the prototype displays all of them as separate results. However, we expect that more mature personal web archiving and search systems will rather condense the different versions of a web page, especially when the context of the query terms is similar in the versions. The resulting user experience offers key advantages with respect to search users' privacy: search activities remain local to WASP, and the user is left in control whether to visit the live web page (without leaking their

³⁹<https://github.com/webrecorder/pywb>

⁴⁰Date and time picker widget: <https://eonasdan.github.io/bootstrap-datetimepicker/>

Web Archive Search Personalized

From:	beginning	month ago	week ago	day ago	2018-04-17 15:04	
Until:	month ago	week ago	day ago	now	now	
wasp						Go!

(5) Page 1 for '**wasp**' from 2018-04-17 15:04 until now

(6) [How to Draw a Wasp \[archive\] \[live\]](#)

(7) <http://how2drawanimals.com/8-animals/183-draw-wasp.html>

(8) @2018-04-18 10:38

Page 1 of 5 [How to Draw a **Wasp**] Please PAUSE the "How to Draw a **Wasp**" video after each ... [Draw **Wasp** 1] Step 1: Draw a circle as a guide for the front part of the wasp's body. ...

(9) [Draw **Wasp** 2] Step 2: Draw a couple of arcs on the sides of the circle to complete the guide for the ... [Draw **Wasp** 3] Step 3: Draw another circle on the left side as a guide for the top part of the head ... [Draw **Wasp** 4] Step 4: Draw another arc below the circle as a guide for the bottom part of the head

Wasp - Wikipedia [archive] [live]

<https://en.wikipedia.org/wiki/Wasp> @2018-04-18 09:59

The term **wasp** is sometimes used more narrowly for the Vespidae, which includes the common **wasp** or yellow ... **wasp**.^[12] The largest social **wasp** is the Asian giant hornet, at up to 5 centimetres (2.0 in) in length ... Retrieved 18 June 2015. 2. ^ a b "**Wasp**". ... **Wasp**. Gollancz Science Fiction. ... OCLC 67375475. 89. ^ USS **Wasp** Veterans (15 June 1999). USS **Wasp**. Turner Publishing. pp. 8–11.

wasp - Wiktionary [archive] [live]

<https://en.wiktionary.org/wiki/wasp> @2018-04-18 09:18

) * common **wasp** (*Vespa vulgaris*) * cuckoo **wasp** (*Chrysididae*) * digger **wasp** (*Sphecidae*) ... **wasp** (*Vespa germanica*) * gold **wasp** (*Chrysididae*) * ichneumon **wasp** (*Ichneumonidae*) * ...

FIGURE 5.8: Search interface for WASP: (1) shortcuts for frequently used time settings; (2) selected query time interval; (3) date and time picker for exact time specification; (4) query box; (5) description of current result page; (6) title of result with links to archived and live version; (7) URL of the result; (8) archive time of the result; (9) snippet for the result.

preferences to another search engine), or to be satisfied with the archived result.

Reproduction Server

When using a personal Web archive in a re-finding scenario, WASP fulfills the need of users to access information from their browsing history using pywb; a state-of-the-art web page reproduction software which uses intricate URL rewriting and code injection to serve the archived web pages like they were originally received (Figure 5.7 (c)). Through the use of specific URLs, pywb can serve multiple versions of the same web page. WASP's search



FIGURE 5.9: Screenshot of a web page reproduced from the archive. pywb is configured to insert a small black banner at the bottom right of the browser viewport to remind users that they are viewing an archived page.

interface uses this feature to refer the user to exactly that version of the web page that corresponds to the clicked result link. In order to avoid confusion on the user's side as to whether or not they are browsing within the archive, a small black banner is inserted and fixed to the bottom right corner of the browser viewport for all pages that are reproduced from the archive (cf. Figure 5.9).

5.2.3 Qualitative Evaluation

Given that the WASP prototype became operational only recently, the ongoing evaluation of its archiving and retrieval quality is still in its infancy. Nevertheless, since we have been using the prototype, this section reports on insights gathered so far, namely the results of an error analysis regarding archiving quality, and an outline of evaluation methodology regarding retrieval quality.

Archiving Quality: Error Analysis

When revisiting an archived web page, one naturally expects the version reproduced from the archive to look and behave exactly the same as the live version did at the time of archiving. Yet, technological difficulties may prevent the faithful reproduction of an archived web page. Since it is usually impractical for WASP to take web server snapshots, WASP will only capture a page's client side. Therefore, only a subset of the potential server interactions end up being represented in the archive and available for the reproduction: the scrolling, clicking, form submissions, video and audio stream playback, etc. that the user performed on the live web page. If user interactions on the archived web page trigger unseen requests to the web server, reproducing the archived web page will either do nothing, show an error, or stop working.

However, even in the case the user repeats the same basic interactions on the archived page that they performed on the live page, only about half of web pages can be reproduced flawlessly [127]. These reproduction errors mostly stem from randomized requests. Indeed, in about two-third of flawed reproductions, the errors are on the level of missing advertisements or similar. While pywb replaces the JavaScript random number generator by a deterministic one, this only affects the archived page and does not fully solve the problem: different timings in the network communications lead to a varying execution order and thus a different order of pop-requests from the "random" number sequence. To greater effect, pywb employs a fuzzy matching of GET parameters that ignores some of the parameters that it assumes to have random values (e.g., session ids), be it by the parameter name or by a hash-like appearance of the parameter value. While it is unclear how many false positives this process introduces, it naturally can't find all random parameters as there exists no standard whatsoever in this regard.

Another interesting problem for web archiving we noticed are push notifications: while they are properly recorded, it remains a difficult choice if and when to trigger them during the reproduction of a web page. Should the trigger time be based on the time spent on the page or based on other events?

Finally, we found that differences between browsers can also affect the reproduction quality. Though this had only minor effects on our experience with WASP so far, the ongoing development of the web technology stack may render old web pages in the archive less reproducible in the long run. For an example, consider the ongoing demise of Flash as a major container for dynamic content. In this regard, old versions of browsers and even old

versions of operating systems may need to be kept, which is a definite requirement for web archiving in general, and also possible based on WASP's use of Docker containers, though not necessarily important for our usage scenario of personal web archiving.

Retrieval Quality Evaluation: An Outline

In principle, it should be easier to re-find something in a personal web archive than using some commercial search engine on the live web. Since a personal archive will typically be many orders of magnitude smaller, not as many candidate results for simple queries exist as on the live web. Ideally, compared to finding a needle in the huge haystack of the web, with a tailored search interface for one's smaller personal archive, the ratio of needles to hay is much higher in a re-finding scenario than in general web search. Still, since WASP is a prototype that was created very recently, we can only provide anecdotes of retrieval problems and sketch how we want to evaluate whether WASP actually helps to re-find needles.

The main evaluation scenarios we envision is re-finding something a user recalls having seen earlier on the web. Such re-finding intents will be different from the frequent re-visit patterns users show on the web [2] since their purpose is not to visit some favorite page but to check some information seen before. In this regard, we do not believe that, at the time of visiting a web page the first time around, users will have enough foresight and presence of mind to anticipate its future uses and hence place a bookmark, rendering a search in their personal archive indispensable.

We used WASP for one week in an informal self-experiment to figure out what problems arise and what should thus be integrated in a formal evaluation. The most obvious problem that differs from the general web search scenario is that of dealing with several versions of the same web page. During our short-term usage of WASP, we found that most retrieved web pages are actually relevant, but that the result lists are cluttered with different versions of the same web page that were—with respect to our information needs—practically identical; as predicted by a recent retrievability study of Web archive search [206]. A probably even more difficult problem, but one that our scenario shares with general web search, arises from the fact that nowadays web pages request a large part of their content dynamically and only if necessary. A good example of this is the Twitter timeline: while scrolling through the timeline, more tweets are requested from the server. Since WASP is currently limited to indexing HTML responses, it catches



(a)

4 signs of progress on climate change

[Here are all the promising developments I'll be talking about at the big climate conference in Paris this week.](#)

gatesnotes.com

Archived Version

(b)

FIGURE 5.10: Example of dynamic HTML content in WASP: (a) original tweet as it appeared while scrolling down the Twitter timeline (b) Twitter card as it was requested for display, archived, and indexed.

only some parts of the tweets (see Figure 5.10), which turn out to be HTML templates requested via Ajax for integration into the Twitter page.

Based on these observations, we propose the following evaluation setup for personal web archives. Since re-finding in personal web archives has not been part of any evaluation campaign so far, a respective set of topics and user interactions has to be built up front. Besides monitoring user queries against WASP's search functionality for users who agree to share parts of their browsing and search activity, one will periodically trigger active users of WASP with a re-finding game similar to PageHunt [162]. The user will be shown the screenshot of a page they have seen, or only parts thereof (e.g., only the color scheme of the layout), or will be asked to re-find a piece of information they have seen a given period of time ago (e.g., three days

ago, two weeks ago, etc.). Their task will be to come up with a sequence of queries (and clicks) such that in the end the prescribed web page appears in the top- k ranks of WASP’s retrieval component. In such cases, the desired item will be known for evaluation purposes and the re-finding task can have several difficulty levels (showing full information vs. only color scheme, target information at top of a page or only requested upon interaction, etc.). To measure retrieval success, the length of real and the comparably artificial re-finding query and click sequences can be measured as well as the specificity of the queries contrasted by the size of the personal collection. But of course, the overall interesting measure will be for how many real re-finding tasks the users are able to pull out the desired result from their personal archive—their needle stack.

5.2.4 Discussion and Lessons Learned

Our primary goal with WASP was to develop a vertical prototype of a web archiving and retrieval framework, which archives every web page and every request made by a web page, and then indexes everything archived. Based on first practical experiences with using WASP for our own respective web traffic, however, there are still many things to be sorted out before we can claim a flawless retrieval experience. Unsurprisingly, the devil is in the details, but somewhat surprisingly, we will be forced to revisit the basic notions of what is a web page, what needs to be archived, and what needs to be indexed. This section discusses lessons learned, outlining a number of exciting future directions for research and development on web archiving and retrieval in general, and for WASP in particular.

Which pages to archive?

Although WASP currently follows “archive first, ask questions later,” users of a personal archiving system likely do not wish for all their traffic to be archived, even if stored within their personal data space. Without specific measures, sensitive data will end up in the archive, e.g., banking pages, health-related browsing, as well as browsing sessions with privacy-mode enabled (where users expect all traces of their activities to be purged after the browser is closed); users may not expect for such data to emerge in search results, weeks, months, or even years later. Furthermore, just as some users regularly clean or clear their browsing history, they will wish to clean or clear their archive. Similarly, it will be necessary to protect the personal archive from unauthorized access, analyze known and new attack

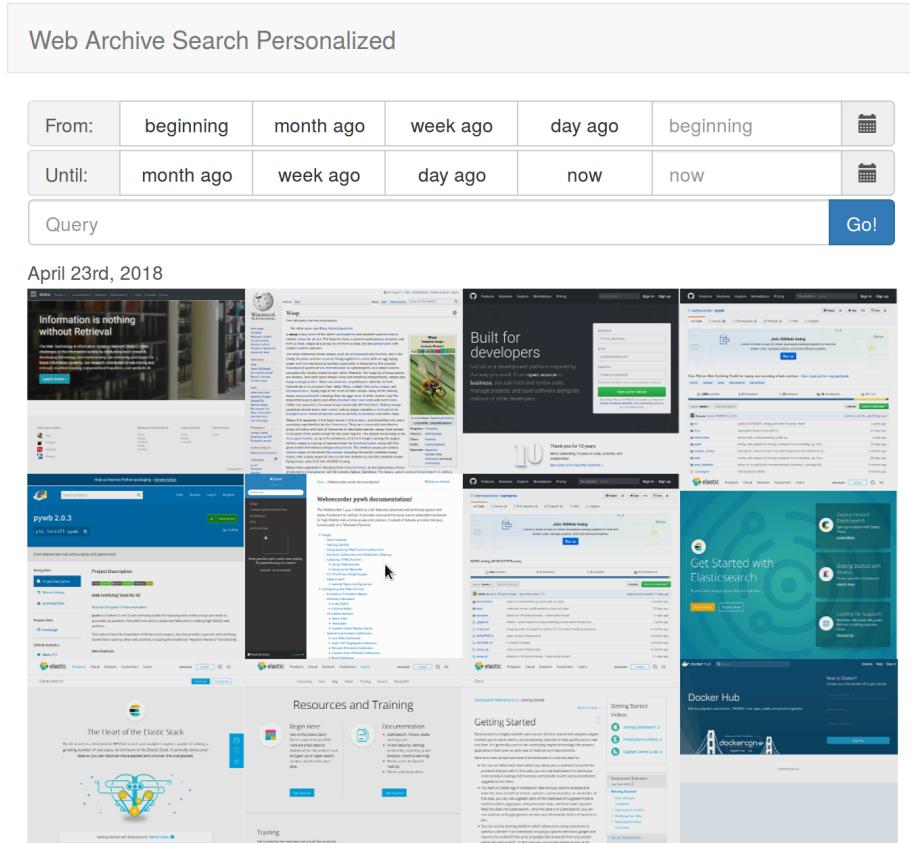


FIGURE 5.11: Mockup illustration for the screenshot mode; the screenshot in the 2nd row and 2nd column is highlighted by mouse-over.

vectors on the archiving setup, and formalize the security implications that stem from the use of such a system.

Based on these deliberations, it is clear that the user must be given fine-grained control over what sites or pages are archived, allowing for personal adjustments and policies. The recorded archive needs to be browseable, so that individual entries can be selected for removal. For more convenient browsing (both for cleaning and general re-finding), we suggest a screenshot-based interface as shown in Figure 5.11. At present, users can already influence which pages should not be archived using proxy-switching plugins available for all modern browsers that seamlessly integrate with WASP's proxy-based architecture (e.g., cf. Figure 5.12). Of course, specifying wildcard expressions hardly qualifies as a user-friendly interface for non-computer scientists, so that a better interface will be required in practice (e.g., using classification techniques similar to Eickhoff et al. [74]).



FIGURE 5.12: Firefox toolbar indicating archiving is activated. The context-menu of this icon allows to turn off the proxy-usage, thereby implementing a “pause-archiving” button.

Under some circumstances personal archiving systems could act on their own behalf to allow for an improved experience of the archived page, by archiving content the users did not request themselves. This possibility leads to several new research questions. For example, should all videos on a visited page be requested and archived, so that the user can watch them later on from their archive? Or in general, should the system predict and simulate interactions that the user may later want to do on the archived page to archive the corresponding resources while they are still available? Moreover, should the system perform such a simulation multiple times in order to detect the randomness in the web page’s requests and consider this information in the reproduction? Additionally, automatic requests based on usual human behavior could provide an additional layer of privacy protection, making it more difficult for unwanted tracking software to devise an individual user profile.

Which pages to index?

While a comprehensive archive is necessary for a high-quality reproduction of web pages, not everything that the browser receives is actually of interest to the user. From our own web browsing habits, we can informally tell that many pages opened are not relevant for future retrieval, because they are dismissed upon first glance (e.g., pop-ups) or not even looked-at at all.

Besides accidental page visits, another example of irrelevant pages may be found in more complex web applications. Take web-based RSS feed readers the likes of Feedly as an example: there is no need to index every page and every state of every page of the feed reader. Rather, the feed items to which the user pays attention are of interest for indexing, since only they are the ones the user may eventually remember and wish to revisit. In this regard, two cases can be distinguished, namely the case where feed items are displayed only partially, so that the user has to click on a link pointing to an external web page to consume a piece of content, and the case where feed items are displayed in full on the feed reader’s page. The former case is straightforward, since a click indicates user attention, so that the

feed reader’s page can be entirely ignored. In the latter case, however, every feed item the user reads should be indexed, whereas the ones the user skips should not so as not to pollute the user’s personal search results.

More generally, all kinds of portal pages and doorway pages, ranging from newspaper front pages via social network pages to search results pages are candidates for omission. Analyzing the user’s browsing behavior gives evidence which page they sufficiently scrutinized for it to be indexed. If a user spends time reading the headlines and excerpts of a front page, this would suggest to index that page, but may be difficult to discern in practice. Otherwise, a user’s behavior may be used as implicit relevance feedback to be incorporated into tailored retrieval models.

What is the document unit for indexing?

In its present form, WASP archives everything that runs under a given URL—including GET parameters, but excluding fragment identifiers—as one unit. Just like in regular search, not every piece of content is relevant for indexing. Main content extraction is an obvious solution to this problem, but the current state-of-the-art frequently fails on pages where many small pieces of content can be found. Furthermore, many websites today spread one coherent piece of content over many sub-pages (so-called pagination). For instance, news publishers often employ pagination, forcing readers to switch pages (possibly to serve extra display ads or improve engagement metrics that determine the value of the display ads shown on the publisher’s site). For archive retrieval purposes, however, pagination can be detrimental, penalizing the relevance of a paginated news article to a query, since only parts of the article are scored at a time.

On the other hand, physical pages are also not necessarily atomic: many web pages built with modern web design tools are single-page applications, where different pieces of content are shown upon user request under the same URL. For instance, a blog platform may show each blog post requested by a user simply by loading it in the background using a JavaScript-based AJAX request, and replacing the currently shown post with a new one. In this case, the perfect web archive search would identify the single posts and index them separately, injecting code upon reproduction that replaces the displayed post with the desired one. Currently, we are technologically far from such a feature. In a different case, like the Twitter timeline, a web page consists of several (possibly independent) content segments. Again, each such segment should be indexed separately for an appropriate relevance computation. To meet this challenge, web pages should be segmented into

coherent units of content that belong together on a page, and each segment identified should be treated as a document unit. However, just like with most of the aforementioned problems, page segmentation, too, is still in its infancy—though progress is achievable, as shown in Section 3.3.

For an optimization, the click behavior and dwell times on certain pages may be the best features to determine what parts should be indexed, whether pages should be merged into one, or one divided into many. Furthermore, such information on user behavior would be very useful for ranking results in the personal search. Currently, however, such behavioral data is probably not even available to commercial search engines.

5.2.5 Conclusion

This work introduces WASP, a prototypical implementation of a personal and private web archive and search system, provides a first qualitative evaluation of such a system, and outlines future steps in this regard, as well as discusses the challenges that such systems face. WASP combines state-of-the-art archiving and retrieval technology to which it adds an intuitive and tailored search interface. Generally, the use case for personal web archive search is more the one of a re-finding engine, with immediate privacy-related benefits. We identify current limitations in archiving technology for this use case and discuss how the evaluation of a search engine has to be adapted for search in personal web archives (e.g., to several versions of a single web page when it is revisited). In the same context, we discuss what content should be archived and what content should be indexed, highlighting privacy issues (e.g., archiving in incognito mode) and advantages (re-finding information using only local data).

5.3 Summary

This chapter focused on harnessing web archives for online security and privacy, illustrating in two studies how web archives can be employed in this respect in both research and practical applications. First, Section 5.1 harnessed large-scale human data of web archives, specifically sentences written, for an in-depth security analysis of widespread password generation advice. This analysis led to the first empirical security estimate of passwords generated according to this advice, allowing for the first time to directly compare this advice with others under different parameters and attack scenarios. Second, Section 5.2 harnessed web archives as a privacy-preserving way to re-find web content by automatically collecting, archiv-

ing, and indexing web content through a proxy-based system. Specifically, the section detailed a possible system setup, contributed a prototypical implementation for practical research, and discussed possibilities, limitations, and further research directions.

6

Conclusion

This chapter concludes this thesis. Section 6.1 reviews its main contributions in light of the research questions and societal challenges discussed in Chapter 1. Section 6.2 then discusses remaining issues in this context, highlighting three especially promising avenues for future research.

6.1 Main Contributions and Implications

This thesis contributed several advancements in both the archiving of web content and the use of such archives to tackle different societal challenges.

Central to both the web archiving process and analyses built on web archives is the so-called web archive ecosystem, which is by itself centered around the WARC file format for web archives. Chapter 2 details our main contribution to the web archive ecosystem, the Webis Web Archiver. The Webis Web Archiver allows for the first time to create a web analysis setup that is authentic through the use of an up-to-date browser and reproducible through web archiving and reproduction technology, combined with browser automation. The later chapters of this thesis showcase different applications of the archiver, for example, to investigate web archive reproduction quality (Section 3.1), to evaluate web page analysis algorithms (here web page segmentation algorithms; Section 3.3), or to create datasets that are then open for extension with other aspects of a web page (here of (hyperpartisan) news; Section 4.2). Moreover, the WASP software for private web archiving detailed in Section 5.2 re-uses parts of the Webis Web Archiver. As these examples illustrate, the Webis Web Archiver presents

a versatile contribution to the web archive ecosystem that supports several use cases in research alone.

Chapters 3 to 5 then tackled issues in the context of three different societal challenges: the preservation of digital culture (Chapter 3), the critical assessment of information (Chapter 4), and online security and privacy (Chapter 5). The thesis adopts the term “societal challenge” from the European Union’s Horizon 2020 program, meaning issues that concern most if not all members of an (at least) nationwide society (cf. Chapter 1). All approaches this thesis presents for tackling the issues rely on web archives.

Chapter 3 focused on harnessing web archives for preserving digital culture. Web archives are the intuitive approach to this challenge. However, issues arose regarding the qualitative assessment of reproductions from archives to guarantee accurate preservation (how to measure web archive reproduction quality?), regarding the detection of unexpected and thus likely mixed-up content in archives (how frequent are different kinds of unexpected content in web pages?), and regarding the unitization of web pages for long-term storage and retrieval (how to define and identify segments of a web page?). The chapter approached each of these issues in turn, contributing both improvements to the accurate preservation of digital culture and means to assess the current state-of-the-art. Section 3.1 contributed an operationalization of a web page’s reproduction quality from the user’s perspective, a dataset with respective annotations, and the first approaches for a corresponding automatic quality assessment. Such an assessment allows for immediate quality control of the archiving process, enabling web archivists to take measures while web pages with low quality scores are still online. Section 3.2 contributed a first analysis of unexpected content in general-purpose web archives. Unexpected content is estimated in the section to affect roughly 10% of web pages in general crawls, suggesting that the detection of such content is needed during archiving. Section 3.3 contributed an in-depth investigation into the task of web page segmentation, suggesting a perception-based approach that allows to index and retrieve digital culture artifacts in human-recognizable units.

Chapter 4 focused on harnessing web archives for a critical assessment of information on the web. Web archives are crucial here, as they preserve the context of information for both provenance and the information’s direct assessment. However, the application of web archives to this end has rarely been demonstrated in practice. The chapter thus showcased both the use of temporal provenance information (how to employ edit histories to assess information?) and the large data volume (how to identify extremist (hyper-partisan) content in news archives?) that web archives provide. In doing

so, the chapter contributed novel approaches to assess online information. Section 4.1 employed the page-specific archive of Wikipedia to assess the quality of 470 million edits using patterns of collaborative quality assurance. This approach enabled a spatio-temporal analysis of reverted edits, which revealed regularities across countries that on-the-fly quality assessment methods can now employ. Section 4.2 employed classical web archives of news articles to enable technologies for the automatic detection of hyperpartisan news. To this end, the section presents two respective web archive datasets. Classification results on these datasets, including those from an international competition on which the section reports, suggest that a large-scale automatic assessment of hyperpartisanship on the web is possible using state-of-the-art natural language processing techniques.

Chapter 5 focused on harnessing web archives for online security and privacy. Web archives are likely not the first approach that comes to one's mind for this challenge. However, as the chapter showed, web archives can be harnessed both for the large-scale human behavioral data that some security and privacy analyses require (here: how to estimate the security of passwords from human sentences?) and as a secure and privacy-preserving way to access web content (here: how to protect privacy when re-finding information?). Section 5.1 harnessed large-scale human data of web archives for an in-depth security analysis of widespread password generation advice: come up with a random sentence and concatenate the first letters of each of its words as the password. The presented analysis based on language models from web archives led to the first empirical security estimate of the corresponding passwords, enabling its analysis under different parameters and attack scenarios. Section 5.2 harnessed web archives in a prototypical system setup that allows for re-finding web content. Using a private web archive, no communication to the original web server is necessary for re-finding, effectively preserving the user's privacy.

Moreover, the approaches introduced in Chapters 3 to 5 represent contributions to the generic data analysis pipeline, allowing others to integrate them into their research. Namely, in (1) the acquisition, preservation, and annotation of web data, e.g., for high-fidelity archives (Section 3.1) or when focusing on web page segments (Section 3.3); (2) the creation of models from the web data, exploiting the temporal information, fidelity, and volume provided by web archives, e.g., for collaborative processes (here for Wikipedia; Section 4.1) or language use (here for mnemonic passwords; Section 5.1); and (3) the analysis of the models to gain insights, e.g., in distributions (here of unexpected content; Section 3.2) or classifier effectiveness (here—among others—for classifying hyperpartisanship of news

articles; Section 4.2). This thesis thus hopes to enable and promote through its example the use of web archives as key components in future data analysis pipelines—both in the context of societal challenges and beyond.

6.2 Open Problems and Future Work

Naturally, many issues remain in the context of harnessing web archives for tackling societal challenges. The following highlights three directions that seem especially promising.

Improving web archive quality Chapter 3 presented multiple contributions towards higher-fidelity web archives. However, given that web pages are becoming increasingly interactive and thus more difficult to preserve, the achieved quality is insufficient for the reliable long-term preservation of digital cultural heritage. The visual approach pursued here is merely the first step, which allows judging deviations in the reproduction—and these may even be inevitable—in human terms. Still, the approach presented here is insufficient even for a visual judgement as it considers the web page as a whole. Integrating web page segmentation into the quality assessment will, on the other hand, allow pinpointing which deviations are responsible for a perceived quality decrease (e.g., “the video element is missing” as opposed to “due to the missing video element, the elements below are now at a higher up on the screen”). Beyond the visual approach, however, it will be necessary to assess the preservation of interactions with the web page. The introduction of the Webis Web Archiver in Chapter 2.2 opened the door for such investigations. Nevertheless, many questions remain on how to select interactions for preservation and to ensure the preservation’s faithfulness.

Improving automated web page comprehension The analyses of Chapters 4 and 5 heavily relied on the archived web data but did not exploit the full potential of web archives. For example, to extract the news article contents from the web archives in Section 4.2, we coded nearly 400 wrappers—one for each news publisher in the dataset. The tailored wrappers have been necessary as the HTML source code for different publishers varies considerably. However, despite the differences in the source code, humans find it very easy to identify the news article on a web page, even for news publishers unknown to them. Rendering the web pages from web archives should thus allow automating wrapper creation reliably, often referred to as “wrapper induction.” However, performing wrapper induction at a human

performance level likely requires comprehending the web page in a more human way, using segment labels such as “main article” or “comment section,” and identifying relations between such segments. Moreover, such a human level page comprehension would enable a great variety of web data analyses—especially ones that combine data from different segments like our current image retrieval for arguments task¹—and thus to further tackle the societal challenges harnessing web archives as a data source.

Simulating web users to analyze online information intake The World Wide Web is today’s most extensive public sphere with a huge impact on our lives and society. Though information retrieval technology shows us what information is potentially available to people, social scientists often wonder what people actually see. To answer this question and tackle the associated societal challenge, scientists today recruit volunteers to share their browsing experience, for example, to assess biases in search engines² or to understand the prevalence of political opinions in “black box” services like Instagram.³ The non-profit organization Algorithm Watch⁴ was created to support such inquiries as well as to observe and influence political decisions in this regard. However, the volunteer-based analyses suffer from a high cost to reach an acceptable scale. Moreover, to not intrude into the privacy of the volunteers, they have to be conducted with great care. Simulated users can solve these problems, allowing for large-scale and continuous studies to prepare or complement the high-fidelity ones based on volunteers. Web archives, collected through the simulated users in a setup similar to WASP (Section 5.2), then provide the raw data for reproducible and extendable analyses on what people see.

More than two millennia ago, Socrates criticized the written word for telling people many things without teaching them [190]. Similarly, web archives do not prevent misinterpretation and wrong conclusions, be they deliberate or not. Still, as this thesis hopefully illustrates, web archives can be a key ingredient for tackling the challenges our societies face now and in the future by preserving the web of today and the past.

¹<https://webis.de/events#touche-2022>

²The Guardian. Wanted: browsers to help uncover the truth about online search result bias. <https://perma.cc/C6D7-84YJ>

³Süddeutsche Zeitung. #wahlfilter: Wie verzerrt ist Instagram? (possible translations: “#electionfilter” or “#choicefilter: How biased is Instagram?”). <https://perma.cc/USP6-W57C>

⁴<https://algorithmwatch.org>

Bibliography

- [1] Daron Acemoglu, Asuman Ozdaglar, and Ali ParandehGheibi. Spread of (Mis)Information in Social Networks. *Games and Economic Behavior*, 70(2):194–227, 2010. doi: 10.1016/j.geb.2010.01.005.
- [2] Eytan Adar, Jaime Teevan, and Susan T. Dumais. Large Scale Analysis of Web Revisitation Patterns. In Mary Czerwinski, Arnold M. Lund, and Desney S. Tan, editors, *26th Conference on Human Factors in Computing Systems (CHI 2008)*, pages 1197–1206. ACM, 2008. doi: 10.1145/1357054.1357241.
- [3] B. Thomas Adler, Luca de Alfaro, Santiago M. Mola-Velasco, Paolo Rosso, and Andrew G. West. Wikipedia Vandalism Detection: Combining Natural Language, Metadata, and Reputation Features. In Alexander F. Gelbukh, editor, *12th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2011)*, pages 277–288. Springer, 2011.
- [4] Sadia Afroz, Michael Brennan, and Rachel Greenstadt. Detecting Hoaxes, Frauds, and Deception in Writing Style Online. In *33rd Symposium on Security and Privacy (S&P 2012)*, pages 461–475. IEEE Computer Society, May 2012. doi: 10.1109/SP.2012.34.
- [5] Nazanin Afsarmanesh, Jussi Karlgren, Peter Sumbler, and Nina Viereckel. Team Harry Friberg at SemEval-2019 Task 4: Identifying Hyperpartisan News through Editorially Defined Metatopics. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [6] Rodrigo Agerri. Doris Martin at SemEval-2019 Task 4: Hyperpartisan News Detection with Generic Semi-supervised Featuresl. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [7] M. Elgin Akpinar and Yeliz Yesilada. Vision Based Page Segmentation Algorithm: Extended and Perceived Success. In *Workshops of the 13th International Conference on Web Engineering (ICWE 2013)*, pages 238–252. Springer, 2013. doi: 10.1007/978-3-319-04244-2_22.

- [8] Amal Alabdulkarim and Tariq Alhindi. Spider-Jerusalem at SemEval-2019 Task 4: Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [9] Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. Cardiff University at SemEval-2019 Task 4: Linguistic Features for Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [10] Yasmin Alnoamany, Ahmed Alsum, Michele C. Weigle, and Michael L. Nelson. Who and What Links to the Internet Archive. In Trond Aalberg, Christos Papatheodorou, Milena Dobreva, Giannis Tsakonas, and Charles J. Farrugia, editors, *17th International Conference on Theory and Practice of Digital Libraries (TPDL 2013)*, pages 346–357. Springer, September 2013.
- [11] Omar Alonso, Vasileios Kandylas, Serge-Eric Tremblay, Jake M. Hofman, and Siddhartha Sen. What’s Happening and What Happened: Searching the Social Web. In *9th Annual Web Science Conference (WebSci 2017)*, pages 191–200. ACM, 2017. ISBN 978-1-4503-4896-6. doi: 10.1145/3091478.3091484.
- [12] Milad Alshomary, Michael Völske, Tristan Licht, Henning Wachsmuth, Benno Stein, Matthias Hagen, and Martin Potthast. Wikipedia Text Reuse: Within and Without. In Leif Azzopardi, Benno Stein, Norbert Fuhr, Philipp Mayr, Claudia Hauff, and Djoerd Hiemstra, editors, *Advances in Information Retrieval. 41st European Conference on IR Research (ECIR 2019)*, volume 11437 of *Lecture Notes in Computer Science*, pages 747–754, Berlin Heidelberg New York, April 2019. Springer. doi: 10.1007/978-3-030-15712-8_49.
- [13] Evan Amason, Jake Palanker, Mary Clare Shen, and Julie Medero. Harvey Mudd College at SemEval-2019 Task 4: The D.X. Beaumont Hyperpartisan News Detector. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [14] Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A Comparison of Extrinsic Clustering Evaluation Metrics Based on Formal Constraints. *Information Retrieval*, 12(4):461–486, 2009. doi: 10.1007/s10791-008-9066-8.

- [15] Judith Jeyafreeda Andrew, Stephane Ferrari, Fabrice Maurel, Gael Dias, and Emmanuel Giguet. Web Page Segmentation for Non Visual Skimming. In *33rd Pacific Asia Conference on Language, Information and Computation (PACLIC 2019)*, pages 423–431, 2019.
- [16] Talita Anthonio and Lennart Kloppenburg. Team Kermit-the-frog at SemEval-2019 Task 4: Bias Detection Through Sentiment Analysis and Simple Linguistic Features. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [17] Judd Antin, Raymond Yee, Coye Cheshire, and Oded Nov. Gender Differences in Wikipedia Editing. In Felipe Ortega and Andrea Forte, editors, *7th International Symposium on Wikis and Open Collaboration (WikiSym 2011)*, pages 11–14. ACM, 2011.
- [18] Ofer Arazy, Felipe Ortega, Oded Nov, M. Lisa Yeo, and Adam Balila. Functional Roles and Career Paths in Wikipedia. In Dan Cosley, Andrea Forte, Luigina Ciolfi, and David McDonald, editors, *18th Conference on Computer Supported Cooperative Work & Social Computing (CSCW 2015)*, pages 1092–1105. ACM, 2015.
- [19] Shlomo Argamon-Engelson, Moshe Koppel, and Galit Avneri. Style-based text categorization: What newspaper am I reading. In *AAAI Workshop on Text Categorization*, pages 1–4. AAAI Press, 1998.
- [20] Javier Arias, Koen Deschacht, and Marie-Francine Moens. Language independent content extraction from web pages. In *9th Dutch-Belgian information retrieval workshop*, pages 50–55. University of Twente; Enschede, The Netherlands, 2009.
- [21] Mikhail J. Atallah, Craig J. McDonough, Victor Raskin, and Sergei Nirenburg. Natural Language Processing for Information Assurance and Security: An Overview and Implementations. In Mary Ellen Zurko and Steven J. Greenwald, editors, *Workshop on New Security Paradigms (NSPW 2000)*, pages 51–65. ACM, 2000. doi: 10.1145/366173.366190.
- [22] Leif Azzopardi. Simulation of Interaction: A Tutorial on Modelling and Simulating User Interaction and Search Behaviour. In *39th International Conference on Research and Development in Information Retrieval (SIGIR 2016)*, pages 1227–1230. ACM, 2016.

- [23] Sameer Badaskar, Sachin Agarwal, and Shilpa Arora. Identifying Real or Fake Articles: Towards better Language Modeling. In *3rd International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pages 817–822. Association for Computer Linguistics, 2008. URL <http://aclweb.org/anthology/I/I08/I08-2115.pdf>.
- [24] Jefferson Bailey, Abigail Grotke, Edward McCain, Christie Moffatt, and Nicholas Taylor. Web Archiving in the United States: a 2016 Survey. *National Digital Stewardship Alliance*, 2017.
- [25] Shumeet Baluja. Browsing on Small Screens: Recasting Web-page Segmentation into an Efficient Machine Learning Framework. In *15th International Conference on World Wide Web (WWW 2006)*, pages 33–42. ACM, 2006. doi: 10.1145/1135777.1135788.
- [26] Vangelis Banos and Yannis Manolopoulos. A Quantitative Approach to Evaluate Website Archivability Using the CLEAR+ Method. *International Journal on Digital Libraries (IJD L)*, 17(2):119–141, June 2016.
- [27] Vangelis Banos, Yunhyong Kim, Seamus Ross, and Yannis Manolopoulos. CLEAR: a Credible Method to Evaluate Website Archivability. In José Luis Borbinha, Michael Nelson, and Steve Knight, editors, *10th International Conference on Preservation of Digital Objects (iPRES 2013)*, 2013.
- [28] Ziv Bar-Yossef, Andrei Z. Broder, Ravi Kumar, and Andrew Tomkins. Sic Transit Gloria Telae: Towards an Understanding of the Web’s Decay. In *13th International Conference on World Wide Web (WWW)*, pages 328–337. ACM, 2004. doi: 10.1145/988672.988716.
- [29] Marco Baroni, Francis Chantree, Adam Kilgarriff, and Serge Sharoff. Cleaneval: a Competition for Cleaning Web Pages. In *6th International Conference on Language Resources and Evaluation (LREC 2008)*, May 2008.
- [30] A. Ben-David and H. Huurde man. Web Archive Search as Research: Methodological and Theoretical Implications. *Alexandria*, 25(1-2):93–111, 2014. doi: 10.7227/ALX.0022.
- [31] Yves Bestgen. Tintin at SemEval-2019 Task 4: Detecting Hyperpartisan News Article with only Simple Tokens. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.

- [32] Janek Bevendorff, Benno Stein, Matthias Hagen, and Martin Potthast. Elastic ChatNoir: Search Engine for the ClueWeb and the Common Crawl. In Leif Azzopardi, Allan Hanbury, Gabriella Pasi, and Benjamin Piwowarski, editors, *Advances in Information Retrieval. 40th European Conference on IR Research (ECIR 2018)*, Lecture Notes in Computer Science, Berlin Heidelberg New York, March 2018. Springer.
- [33] Shweta Bhatt, Sagar Joglekar, Shehar Bano, and Nishanth Sastry. Illuminating the ecosystem of partisan websites. In *27th International Conference on World Wide Web (WWW 2018 Companion)*, page 545–554. ACM, 2018. doi: 10.1145/3184558.3188725.
- [34] Lidong Bing, Rui Guo, Wai Lam, Zheng-Yu Niu, and Haifeng Wang. Web Page Segmentation with Structured Prediction and its Application in Web Page Classification. In *37th International Conference on Research and Development in Information Retrieval (SIGIR 2014)*, pages 767–776. ACM, 2014. doi: 10.1145/2600428.2609630.
- [35] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006. ISBN 978-0-387-31073-2.
- [36] Joseph Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *33rd Symposium on Security and Privacy (S&P 2012)*, pages 538–552. IEEE Computer Society, 2012. doi: 10.1109/SP.2012.49.
- [37] Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. Passwords and the Evolution of Imperfect Authentication. *Communications of the ACM*, 58(7):78–87, 2015. doi: 10.1145/2699390.
- [38] Peter Bourgonje, Julián Moreno Schneider, and Georg Rehm. From Clickbait to Fake News Detection: An Approach based on Detecting the Stance of Headlines to Articles. In Octavian Popescu and Carlo Strapparava, editors, *Natural Language Processing meets Journalism (NLPmJ 2017)*, pages 84–89. Association for Computational Linguistics, 2017. doi: 10.18653/v1/w17-4215.
- [39] Serdar Boztas. Entropies, Guessing and Cryptography. Technical report, Royal Melbourne Institute of Technology, Dept. of Mathematics Melbourne, 1999.

- [40] Justin Brunelle, Mat Kelly, Hany SalahEldeen, Michele C. Weigle, and Michael L. Nelson. Not All Mementos Are Created Equal: Measuring the Impact of Missing Resources. *International Journal of Digital Libraries (IJDL)*, May 2015.
- [41] Ceren Budak, Divyakant Agrawal, and Amr El Abbadi. Limiting the Spread of Misinformation in Social Networks. In *20th International Conference on World Wide Web (WWW 2011)*, pages 665–674. ACM, 2011. ISBN 978-1-4503-0632-4. doi: 10.1145/1963405.1963499.
- [42] William E. Burr, Donna F. Dodson, Elaine M. Newton, Ray A. Perlner, W. Timothy Polk, Sarbari Gupta, and Emad A. Nabbus. NIST Special Publication 800-63-2: Electronic Authentication Guideline. Technical report, National Institute of Standards and Technology, 2013.
- [43] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. Extracting Content Structure for Web Pages Based on Visual Representation. In *5th Asian-Pacific Web Conference (APWeb 2003)*, pages 406–417. Springer, 2003. doi: 10.1007/3-540-36901-5_42.
- [44] John Canny. A Computational Approach to Edge Detection. *Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, June 1986. ISSN 0162-8828. doi: 10.1109/TPAMI.1986.4767851.
- [45] Jiaxin Cao, Bo Mao, and Junzhou Luo. A Segmentation Method for Web Page Analysis Using Shrinking and Dividing. *International Journal of Parallel, Emergent and Distributed Systems (IJPEDS)*, 25(2):93–104, 2010. doi: 10.1080/17445760802429585.
- [46] Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal Sentence Encoder, 2018. URL <http://arxiv.org/abs/1803.11175>.
- [47] Deepayan Chakrabarti, Ravi Kumar, and Kunal Punera. A Graph-theoretic Approach to Webpage Segmentation. In *17th International Conference on World Wide Web (WWW 2008)*, pages 377–386. ACM, 2008. doi: 10.1145/1367497.1367549.
- [48] Nikhil Chakravartula, Vijayasardhi Indurthi, and Bakhtiyar Syed. Fermi at SemEval-2019 Task 4: The sarah-jane-smith Hyperpartisan News Detector. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.

- [49] Celena Chen, Celine Park, Jason Dwyer, and Julie Medero. Harvey Mudd College at SemEval-2019 Task 4: The Carl Kolchak Hyperpartisan News Detector. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [50] Jinlin Chen, Baoyao Zhou, Jin Shi, Hongjiang Zhang, and Qiu Fengwu. Function-based Object Model Towards Website Adaptation. In *10th International Conference on World Wide Web (WWW 2001)*, pages 587–596. ACM, 2001. ISBN 1-58113-348-0. doi: 10.1145/371920.372161.
- [51] Kai Chen, Jiaqi Wang an Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDection: Open mmlab detection toolbox and benchmark. *CoRR*, abs/1906.07155, 2019. URL <http://arxiv.org/abs/1906.07155>.
- [52] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. *CoRR*, abs/1901.07518, 2019. URL <http://arxiv.org/abs/1901.07518>.
- [53] Stanley F. Chen and Joshua Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. *Computer Speech & Language*, 13(4):359–393, 1999. doi: 10.1006/csla.1999.0128.
- [54] Yimin Chen, Niall J. Conroy, and Victoria L. Rubin. News in an Online World: The Need for an "Automatic Crap Detector". In *78th ASIS&T Annual Meeting (ASIST 2015)*, pages 81:1–81:4. American Society for Information Science, 2015. ISBN 0-87715-547-X. URL <http://dl.acm.org/citation.cfm?id=2857070.2857151>.
- [55] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. Computational Fact Checking from Knowledge Networks. *PLOS ONE*, 10(6):1–13, 2015. URL <https://doi.org/10.1371/journal.pone.0128193>.
- [56] Christian Clausner, Apostolos Antonacopoulos, and Stefan Pletschacher. ICDAR2017 competition on recognition of documents with complex layouts - RDCL2017. In *14th International Conference on*

- Document Analysis and Recognition (ICDAR 2017)*, pages 1404–1410. IEEE Computer Society, 2017. doi: 10.1109/ICDAR.2017.229.
- [57] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Routledge, New York, 2nd edition, 1988. ISBN 9780203771587. doi: 10.4324/9780203771587.
 - [58] Gordon V. Cormack, Mark D. Smucker, and Charles L. A. Clarke. Efficient and Effective Spam Filtering and Re-ranking for Large Web Datasets. *Information Retrieval*, 14(5):441–465, 2011.
 - [59] Michael Cormier, Karyn Moffatt, Robin Cohen, and Richard Mann. Purely Vision-based Segmentation of Web Pages for Assistive Technology. *Computer Vision and Image Understanding*, 148:46–66, 2016. doi: 10.1016/j.cviu.2016.02.007.
 - [60] Michael Cormier, Richard Mann, Karyn Moffatt, and Robin Cohen. Towards an Improved Vision-Based Web Page Segmentation Algorithm. In *14th Conference on Computer and Robot Vision (CRV 2017)*, pages 345–352. IEEE Computer Society, 2017. doi: 10.1109/CRV.2017.38.
 - [61] Rebekah Cramerus and Tatjana Scheffler. Team Kit Kittredge at SemEval-2019 Task 4. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
 - [62] André Cruz, Gil Rocha, Rui Sousa-Silva, and Henrique Lopes Car-doso. Team Fernando-Pessa at SemEval-2019 Task 4: Back to Basics in Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
 - [63] J. Shane Culpepper, Fernando Diaz, and Mark D. Smucker. Research Frontiers in Information Retrieval: Report from the Third Strategic Workshop on Information Retrieval in Lorne (SWIRL 2018). *SIGIR Forum*, 52(1):34–90, 2018. doi: 10.1145/3274784.3274788.
 - [64] Edward Cutrell, Daniel Robbins, Susan T. Dumais, and Raman Sarin. Fast, Flexible Filtering with Phlat. In Rebecca E. Grinter, Tom Rodden, Paul M. Aoki, Edward Cutrell, Robin Jeffries, and Gary M. Olson, editors, *24th Conference on Human Factors in Computing Systems (CHI 2006)*, pages 261–270. ACM, 2006. ISBN 1-59593-372-7. doi: 10.1145/1124772.1124812.

- [65] Johannes Daxenberger and Iryna Gurevych. A Corpus-Based Study of Edit Categories in Featured and Non-Featured Wikipedia Articles. In Martin Kay and Christian Boitet, editors, *24th International Conference on Computational Linguistics (COLING 2012)*, pages 711–726. Association for Computational Linguistics, 2012.
- [66] Johannes Daxenberger and Iryna Gurevych. Automatically Classifying Edit Categories in Wikipedia Revisions. In *2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 578–589. Association for Computational Linguistics, 2013.
- [67] Matteo Dell’Amico, Pietro Michiardi, and Yves Roudier. Password Strength: An Empirical Analysis. In *29th Conference on Information Communications (INFOCOM 2010)*, pages 983–991. IEEE Computer Society, 2010. doi: 10.1109/INFCOM.2010.5461951.
- [68] Yuntian Deng, Anssi Kanervisto, and Alexander M. Rush. What You Get Is What You See: A Visual Markup Decompiler. *CoRR*, abs/1609.04938, 2016.
- [69] Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. SemEval-2017 Task 8: RumourEval: Determining Rumour Veracity and Support for Rumours. In Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel M. Cer, and David Jurgens, editors, *11th International Workshop on Semantic Evaluation (SemEval 2017)*, pages 69–76. Association for Computational Linguistics, 2017. doi: 10.18653/v1/S17-2006.
- [70] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, (NAACL-HLT 2019)*. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423.
- [71] Djellel Difallah, Elena Filatova, and Panos Ipeirotis. Demographics and Dynamics of Mechanical Turk Workers. In *11th International Conference on Web Search and Data Mining (WSDM 2018)*, pages 135–143. ACM, 2018. ISBN 978-1-4503-5581-0. doi: 10.1145/3159652.3159661.

- [72] Mehdi Drissi, Pedro Sandoval Segura, Vivaswat Ojha, and Julie Medero. Harvey Mudd College at SemEval-2019 Task 4: The Clint Buchanan Hyperpartisan News Detector. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [73] Susan T. Dumais, Edward Cutrell, JJ Cadiz, Gavin Jancke, Raman Sarin, and Daniel C. Robbins. Stuff I'Ve Seen: A System for Personal Information Retrieval and Re-use. In Charles L. A. Clarke, Gordon V. Cormack, Jamie Callan, David Hawking, and Alan F. Smeaton, editors, *26th Annual International Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 72–79. ACM, 2003. ISBN 1-58113-646-3. doi: 10.1145/860435.860451.
- [74] Carsten Eickhoff, Kevyn Collins-Thompson, Paul N. Bennett, and Susan T. Dumais. Designing Human-Readable User Profiles for Search Evaluation. In Pavel Serdyukov, Pavel Braslavski, Sergei O. Kuznetsov, Jaap Kamps, Stefan M. Rüger, Eugene Agichtein, Ilya Segalovich, and Emine Yilmaz, editors, *35th European Conference on IR Research (ECIR 2013)*, pages 701–705, 2013. doi: 10.1007/978-3-642-36973-5_64.
- [75] D. W. Embley, Y. Jiang, and Y.-K. Ng. Record-boundary Discovery in Web Documents. In *International Conference on Management of Data (SIGMOD 1999)*, pages 467–478. ACM, 1999. ISBN 1-58113-084-8. doi: 10.1145/304182.304223.
- [76] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. Open Information Extraction from the Web. *Communications of the ACM*, 51(12):68–74, December 2008. ISSN 0001-0782. doi: 10.1145/1409360.1409378.
- [77] Muhammad Faheem. Intelligent Crawling of Web Applications for Web Archiving. In *21st International Conference on World Wide Web (WWW)*, pages 127–132. ACM, 2012.
- [78] Michael Färber, Agon Qurdina, and Lule Ahmed. Team Peter Brinkmann at SemEval-2019 Task 4: Detecting Biased News Articles Using Convolutional Neural Networks. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.

- [79] David C. Feldmeier and Philip R. Karn. UNIX Password Security - Ten Years Later. In Gilles Brassard, editor, *9th Annual International Cryptology Conference (CRYPTO 1989)*, pages 44–63. Springer, 1989. doi: 10.1007/0-387-34805-0_6.
- [80] David Fernandes, Edleno Silva de Moura, Altigran Soares da Silva, Berthier A. Ribeiro-Neto, and Edisson Braga Araújo. A Site Oriented Method for Segmenting Web Pages. In *34th International Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 215–224. ACM, 2011. doi: 10.1145/2009916.2009949.
- [81] Oliver Ferschke, Diyi Yang, and Carolyn Rosè. A Lightly Supervised Approach to Role Identification in Wikipedia Talk Page Discussions. In *9th International Conference on Web and Social Media (ICWSM 2015)*. AAAI Press, 2015.
- [82] Rudolph Flesch. A New Readability Yardstick. *Journal of Applied Psychology*, 32(3):221, 1948.
- [83] Fabian Flöck, Denny Vrandečić, and Elena Simperl. Revisiting Reverts: Accurate Revert Detection in Wikipedia. In Ethan V. Munson and Markus Strohmaier, editors, *23rd Conference on Hypertext and Social Media (HT 2012)*, pages 3–12. ACM, 2012.
- [84] Eibe Frank, Mark A. Hall, and Ian H. Witten. *Data Mining: Practical Machine Learning Tools and Techniques*, chapter The WEKA Workbench. Morgan Kaufmann, 4th edition, 2016. ISBN 0128042915, 9780128042915.
- [85] Norbert Fuhr, Anastasia Giachanou, Gregory Grefenstette, Iryna Gurevych, Andreas Hanselowski, Kalervo Jarvelin, Rosie Jones, Yiqun Liu, Josiane Mothe, Wolfgang Nejdl, Isabella Peters, and Benno Stein. An Information Nutritional Label for Online Documents. *SIGIR Forum*, 51(3):44–66, December 2017. ISSN 0163-5840. doi: 10.1145/3190580.3190588. URL <https://sigir.org/forum/issues/december-2017/>.
- [86] William A. Gale. Good-Turing Smoothing Without Tears. *Journal of Quantitative Linguistics*, 2, 1995.
- [87] R. Stuart Geiger and Aaron Halfaker. When the Levee Breaks: Without Bots, What Happens to Wikipedia’s Quality Control Processes?

- In Ademar Aguiar and Dirk Riehle, editors, *9th International Symposium on Wikis and Open Collaboration (WikiSym 2013)*, pages 6:1–6:6. ACM, 2013.
- [88] R. Stuart Geiger and David Ribes. The Work of Sustaining Order in Wikipedia: The Banning of a Vandal. In Kori Inkpen, Carl Gutwin, and John C. Tang, editors, *13th Conference on Computer Supported Cooperative Work (CSCW 2010)*, pages 117–126. ACM, 2010.
 - [89] Marjan Ghazvininejad and Kevin Knight. How to Memorize a Random 60-Bit String. In Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar, editors, *2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2015)*, pages 1569–1575, Denver, Colorado, 2015. Association for Computational Linguistics. doi: 10.3115/v1/n15-1180.
 - [90] Alexandru L. Ginsca, Adrian Popescu, and Mihai Lupu. Credibility in Information Retrieval. *Foundations and Trends in Information Retrieval*, 9(5):355–475, December 2015. ISSN 1554-0669. doi: 10.1561/1500000046.
 - [91] Scott A. Golder and Michael W. Macy. Diurnal and Seasonal Mood Vary with Work, Sleep, and Daylength Across Diverse Cultures. *Science*, 333(6051):1878–1881, 2011.
 - [92] E. Bruce Goldstein. *Sensation and Perception*. Cengage Learning, 8 edition, 2009. ISBN 9780495601494.
 - [93] Tim Gollub, Martin Potthast, and Benno Stein. Shaping the Information Nutrition Label. In Dyaa Albakour, David Corney, Julio Gonzalo, Miguel Martinez, Barbara Poblete, and Andreas Valochas, editors, *2nd International Workshop on Recent Trends in News Information Retrieval (NewsIR 2018) at ECIR*, volume 2079 of *CEUR Workshop Proceedings*, pages 9–11, March 2018. URL <http://ceur-ws.org/Vol-2079/>.
 - [94] Daniel Gomes, João Miranda, and Miguel Costa. A Survey on Web Archiving Initiatives. In Stefan Gradmann, Francesca Borri, Carlo Meghini, and Heiko Schuldt, editors, *15th International Conference on Theory and Practice of Digital Libraries (TPDL 2011)*, pages 408–420. Springer, 2011.
 - [95] Thomas Gottron and Nedim Lipka. A Comparison of Language Identification Approaches on Short, Query-Style Texts. In Cathal Gurrin,

- Yulan He, Gabriella Kazai, Udo Kruschwitz, Suzanne Little, Thomas Roelleke, Stefan M. Rüger, and Keith van Rijsbergen, editors, *Advances in Information Retrieval. 32nd European Conference on Information Retrieval (ECIR 2010)*, volume 5993 of *Lecture Notes in Computer Science*, pages 611–614. Springer, Heidelberg, March 2010. ISBN 978-3-642-12274-3. doi: 10.1007/978-3-642-12275-0_59.
- [96] Eric Grosse and Mayank Upadhyay. Authentication at Scale. *Security and Privacy*, 11(1):15–22, 2013. doi: 10.1109/MSP.2012.162.
- [97] Peter Grzybek. History and Methodology of Word Length Studies. In *Contributions to the Science of Text and Language: Word Length Studies and Related Issues*. Springer, 2007. doi: 10.1007/978-1-4020-4068-9_2.
- [98] Viresh Gupta, Baani Leen Kaur Jolly, Ramneek Kaur, and Tanmoy Chakraborty. Clark Kent at SemEval-2019 Task 4: Stylometric Insights into Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [99] Karl Gyllstrom, Carsten Eickhoff, Arjen P. de Vries, and Marie-Francine Moens. The Downside of Markup: Examining the Harmful Effects of CSS and Javascript on Indexing Today’s Web. In *21st International Conference on Information and Knowledge Management (CIKM 2012)*, pages 1990–1994. ACM, 2012.
- [100] Matthias Hagen, Martin Potthast, Michel Büchner, and Benno Stein. Webis: An Ensemble for Twitter Sentiment Detection. In Daniel Cer, David Jurgens, Preslav Nakov, and Torsten Zesch, editors, *9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 582–589. Association for Computational Linguistics, June 2015. URL <http://alt.qcri.org/semeval2015/cdrom/pdf/SemEval097.pdf>.
- [101] Aaron Halfaker, Aniket Kittur, and John Riedl. Don’t Bite the Newbies: How Reverts Affect the Quantity and Quality of Wikipedia Work. In Felipe Ortega and Andrea Forte, editors, *7th International Symposium on Wikis and Open Collaboration (WikiSym 2011)*, pages 163–172. ACM, 2011.
- [102] Aaron Halfaker, R. Stuart Geiger, Jonathan Morgan, and John Riedl. The Rise and Decline of an Open Collaboration System: How Wikipedia’s Reaction to Sudden Popularity is Causing its Decline. *American Behavioral Scientist*, 57(5):664–688, 2013.

- [103] Kazuaki Hanawa, Shota Sasaki, Hiroki Ouchi, Jun Suzuki, and Kentaro Inui. The Sally Smedley Hyperpartisan News Detector at SemEval-2019 Task 4. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [104] Gen Hattori, Keiichiro Hoashi, Kazunori Matsumoto, and Fumiaki Sugaya. Robust Web Page Segmentation for Mobile Terminal Using Content-distances and Page Layout Information. In *16th International Conference on World Wide Web (WWW 2007)*, pages 361–370. ACM, 2007. doi: 10.1145/1242572.1242622.
- [105] Stefan Heindorf, Martin Potthast, Benno Stein, and Gregor Engels. Vandalism Detection in Wikidata. In Snehasis Mukhopadhyay, ChengXiang Zhai, Elisa Bertino, Fabio Crestani, Javed Mostafa, Jie Tang, Luo Si, Xiaofang Zhou, Yi Chang, Yunyao Li, and Parikshit Sondhi, editors, *25th ACM International Conference on Information and Knowledge Management (CIKM 2016)*, pages 327–336. ACM, October 2016. ISBN 978-1-4503-4073-1. doi: 10.1145/2983323.2983740.
- [106] Michael Herczeg and Michael Koch. Allgegenwärtige mensch-computer-interaktion - dienste für alle nutzbar und beherrschbar machen. *Inform. Spektrum*, 38(4):290–295, 2015. doi: 10.1007/s00287-015-0901-1.
- [107] Helen Hockx-Yu, Stephen Johnson, Lewis Crawford, and Roger Coram. Capturing and Replaying Streaming Media in a Web Archive—A British Library Case Study. In Andreas Rauber, Panos Constantopoulos, Max Kaiser, and Rebecca Guenther, editors, *7th International Conference on Digital Preservation (iPRES 2010)*, 2010.
- [108] Helge Holzmann, Vinay Goel, and Avishek Anand. ArchiveSpark: Efficient Web Archive Access, Extraction and Derivation. In *16th Joint Conference on Digital Libraries (JCDL 2016)*, pages 83–92. ACM, 2016. ISBN 978-1-4503-4229-2. doi: 10.1145/2910896.2910902.
- [109] Benjamin D. Horne and Sibel Adali. This Just In: Fake News Packs a Lot in Title, Uses Simpler, Repetitive Content in Text Body, More Similar to Satire than Real News. *CoRR*, abs/1703.09398, 2017. URL <http://arxiv.org/abs/1703.09398>.
- [110] Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard H. Hovy. Learning Whom to Trust with MACE. In Lucy Vanderwende,

- Hal Daumé III, and Katrin Kirchhoff, editors, *10th Conference of the North American Chapter of the Association of Computational Linguistics: Human Language Technologies (NAACL 2013)*, pages 1120–1130. Association for Computational Linguistics, June 2013.
- [111] Jeremy Howard and Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification. In Iryna Gurevych and Yusuke Miyao, editors, *56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 328–339. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-1031.
- [112] James Howison, Andrea Wiggins, and Kevin Crowston. Validity issues in the use of social network analysis with digital trace data. *Journal of the Association for Information Systems*, 12(12):767, 2011.
- [113] Tim Isbister and Fredrik Johansson. Dick-Preston and Morbo at SemEval-2019 Task 4: Transfer Learning for Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [114] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988. ISBN 0-13-022278-X.
- [115] Sundararaman Jeyaraman and Umut Topkara. Have the Cake and Eat It Too - Infusing Usability into Text-password Based Authentication Systems. In *21st Annual Computer Security Applications Conference (ACSAC 2005)*, pages 473–482. IEEE Computer Society, 2005. ISBN 0-7695-2461-3. doi: 10.1109/CSAC.2005.28.
- [116] Ye Jiang, Johann Petrak, Xingyi Song, Kalina Bontcheva, and Diana Maynard. Team Bertha von Suttner at SemEval-2019 Task 4: Hyperpartisan News Detection using ELMo Sentence Representation Convolutional Network. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [117] William Jones. *Keeping Found Things Found: The Study and Practice of Personal Information Management*. Morgan Kaufmann, 2010. ISBN 978-0-12-370866-3. doi: 10.1016/B978-0-12-370866-3.X5001-2.
- [118] Youngjun Joo and Inchon Hwang. Steve Martin at SemEval-2019 Task 4: Ensemble Learning Model for Detecting Hyperpartisan News. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.

- [119] Kevin A Juang, Sanjay Ranganayakulu, and Joel S Greenstein. Using System-generated Mnemonics to Improve the Usability and Security of Password Authentication. In *56th Human Factors and Ergonomics Society Annual Meeting (HFES 2012)*, pages 506–510. SAGE, 2012. doi: 10.1177/1071181312561105.
- [120] Andrej Karpathy. CS231n Convolutional Neural Networks for Visual Recognition, 2017. <http://cs231n.github.io/convolutional-networks>.
- [121] Patrick Gage Kelley, Saranga Komanduri, Michelle L. Mazurek, Richard Shay, Timothy Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Julio Lopez. Guess Again (and Again and Again): Measuring Password Strength by Simulating Password-Cracking Algorithms. In *33rd Symposium on Security and Privacy (S&P 2012)*, pages 523–537. IEEE Computer Society, 2012.
- [122] Auguste Kerckhoffs. La Cryptographie Militaire. *JSM*, 9:161–191, February 1883.
- [123] Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein. Spatio-temporal Analysis of Reverted Wikipedia Edits. In *Eleventh International AAAI Conference on Web and Social Media (ICWSM 2017)*, pages 122–131, May 2017. URL <https://www.aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/viewFile/15699/14802>.
- [124] Johannes Kiesel, Benno Stein, and Stefan Lucks. A Large-scale Analysis of the Mnemonic Password Advice. In *24th Annual Network and Distributed System Security Symposium (NDSS 2017)*. Association for Computational Linguistics, February 2017. doi: 10.14722/ndss.2017.23077.
- [125] Johannes Kiesel, Arefeh Bahrami, Benno Stein, Avishek Anand, and Matthias Hagen. Toward Voice Query Clarification. In *41st International ACM Conference on Research and Development in Information Retrieval (SIGIR 2018)*, pages 1257–1260. ACM, July 2018. doi: 10.1145/3209978.3210160. URL <https://dl.acm.org/doi/10.1145/3209978.3210160>.
- [126] Johannes Kiesel, Arjen P. de Vries, Matthias Hagen, Benno Stein, and Martin Potthast. WASP: Web Archiving and Search Personalized. In Omar Alonso and Gianmaria Silvello, editors, *1st International Conference on Design of Experimental Search & Information Retrieval Systems*

- (*DESires 2018*), volume 2167 of *CEUR Workshop Proceedings*, pages 16–21, August 2018. URL <http://ceur-ws.org/Vol-2167/>.
- [127] Johannes Kiesel, Florian Kneist, Milad Alshomary, Benno Stein, Matthias Hagen, and Martin Potthast. Reproducible Web Corpora: Interactive Archiving with Automatic Quality Assessment. *Journal of Data and Information Quality (JDIQ)*, 10(4):17:1–17:25, October 2018. doi: 10.1145/3239574. URL <https://dl.acm.org/doi/10.1145/3239574>.
 - [128] Johannes Kiesel, Arefeh Bahrami, Benno Stein, Avishek Anand, and Matthias Hagen. Clarifying False Memories in Voice-based Search. In Martin Halvey, Ian Ruthven, Leif Azzopardi, Vanessa Murdock, Pernilla Qvarfordt, and Hideo Joho, editors, *2019 Conference on Human Information Interaction & Retrieval (CHIIR 2019)*, pages 331–335. ACM, March 2019. doi: 10.1145/3295750.3298961. URL <https://dl.acm.org/authorize?N686797>.
 - [129] Johannes Kiesel, Fabienne Hubricht, Benno Stein, and Martin Potthast. A Dataset for Content Error Detection in Web Archives. In Maria Bonn, Stephen J. Downie, Alain Martaus, and Dan Wu, editors, *18th ACM/IEEE Joint Conference on Digital Libraries (JCDL 2019)*, pages 349–350. ACM, June 2019. doi: 10.1109/JCDL.2019.00065. URL <https://dl.acm.org/doi/abs/10.1109/JCDL.2019.00065>.
 - [130] Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. SemEval-2019 Task 4: Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*, pages 829–839. Association for Computational Linguistics, June 2019. doi: 10.18653/v1/S19-2145. URL <https://www.aclweb.org/anthology/S19-2145>.
 - [131] Johannes Kiesel, Florian Kneist, Lars Meyer, Kristof Komlossy, Benno Stein, and Martin Potthast. Web Page Segmentation Revisited: Evaluation Framework and Dataset. In Mathieu d’Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux, editors, *29th ACM International Conference on Information and Knowledge Management (CIKM 2020)*, pages 3047–3054. ACM, October 2020. doi: 10.1145/3340531.3412782. URL <https://dl.acm.org/doi/10.1145/3340531.3412782?cid=99659134098>.
 - [132] Johannes Kiesel, Kevin Lang, Henning Wachsmuth, Eva Hornecker, and Benno Stein. Investigating Expectations for Voice-based and

- Conversational Argument Search on the Web. In Luanne Freund, Heather O'Brien, Ioannis Arapakis, Orland Hoeber, and Irene Lopatovska, editors, *2020 Conference on Human Information Interaction & Retrieval (CHIIR 2020)*, pages 53–62. ACM, March 2020. doi: 10.1145/3343413.3377978. URL <https://dl.acm.org/doi/10.1145/3343413.3377978?cid=99659134098>.
- [133] Johannes Kiesel, Xiaoni Cai, Roxanne El Baff, Benno Stein, and Matthias Hagen. Toward Conversational Query Reformulation. In Omar Alonso, Marc Najork, and Gianmaria Silvello, editors, *2nd International Conference on Design of Experimental Search & Information Retrieval Systems (DESires 2021)*, CEUR Workshop Proceedings, pages 91–101, September 2021. URL <http://ceur-ws.org/Vol-2950/paper-12.pdf>.
- [134] Johannes Kiesel, Lars Meyer, Florian Kneist, Benno Stein, and Martin Potthast. An Empirical Comparison of Web Page Segmentation Algorithms. In Djoerd Hiemstra, Maria-Francine Moens, Josiane Mothe, Raffaele Perego, Martin Potthast, and Fabrizio Sebastiani, editors, *Advances in Information Retrieval. 43rd European Conference on IR Research (ECIR 2021)*, volume 12657 of *Lecture Notes in Computer Science*, pages 62–74, Berlin Heidelberg New York, March 2021. Springer. doi: 10.1007/978-3-030-72240-1\5. URL <https://dl.acm.org/doi/10.1145/3340531.3412782?cid=99659134098>.
- [135] Johannes Kiesel, Lars Meyer, Martin Potthast, and Benno Stein. Meta-Information in Conversational Search. *ACM Transactions on Information Systems (ACM TOIS)*, 39(4), August 2021. ISSN 1046-8188. doi: 10.1145/3468868. URL <https://doi.org/10.1145/3468868>.
- [136] Johannes Kiesel, Damiano Spina, Henning Wachsmuth, and Benno Stein. The Meant, the Said, and the Understood: Conversational Argument Search and Cognitive Biases. In Stephan Schlögl, Martin Porcheron, and Leigh Clark, editors, *3rd Conference on Conversational User Interfaces (CUI 2021)*, New York, July 2021. ACM. doi: 10.1145/3469595.3469615. URL <https://dl.acm.org/doi/10.1145/3469595.3469615?cid=99659134098>.
- [137] Johannes Kiesel, Volker Bernhard, Marcel Gohsen, Josef Roth, and Benno Stein. What is That? Crowdsourcing Questions to a Virtual Exhibition. In *2022 Conference on Human Information Interaction &*

- Retrieval (CHIIR 2022)*. ACM, March 2022. doi: 10.1145/3498366.3505836.
- [138] Yoon Kim. Convolutional Neural Networks for Sentence Classification. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*. Association for Computational Linguistics, 2014. doi: 10.3115/v1/d14-1181.
 - [139] Aniket Kittur, Ed Chi, Bryan A. Pendleton, Bongwon Suh, and Todd Mytkowicz. Power of the Few vs. Wisdom of the Crowd: Wikipedia and the Rise of the Bourgeoisie. *WWW Journal*, 1(2):19, 2007.
 - [140] Aniket Kittur, Bongwon Suh, Bryan A. Pendleton, and Ed H. Chi. He Says, She Says: Conflict and Coordination in Wikipedia. In Mary Beth Rosson and David J. Gilmore, editors, *25th Conference on Human Factors in Computing Systems (CHI 2007)*, pages 453–462. ACM, 2007.
 - [141] Aniket Kittur, Ed H. Chi, and Bongwon Suh. Crowdsourcing User Studies with Mechanical Turk. In Mary Czerwinski, Arnold M. Lund, and Desney S. Tan, editors, *26th Conference on Human Factors in Computing Systems (CHI 2008)*, pages 453–456. ACM, 2008. doi: 10.1145/1357054.1357127.
 - [142] Jürgen Knauth. Orwellian-times at SemEval-2019 Task 4: A Stylistic and Content-based Classifier. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
 - [143] Christian Kohlschütter and Wolfgang Nejdl. A Densitometric Approach to Web Page Segmentation. In *17th Conference on Information and Knowledge Management (CIKM 2008)*, pages 1173–1182. ACM, 2008. doi: 10.1145/1458082.1458237.
 - [144] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate Detection Using Shallow Text Features. In *3rd International Conference on Web Search and Data Mining (WSDM 2010)*, pages 441–450. ACM, 2010.
 - [145] Milos Kovacevic, Michelangelo Diligenti, Marco Gori, and Veljko M. Milutinovic. Recognition of Common Areas in a Web Page Using Visual Information: A Possible Application in a Page Classification. In *20th International Conference on Data Mining (ICDM 2002)*, pages 250–257. IEEE Computer Society, 2002. doi: 10.1109/ICDM.2002.1183910.

- [146] Robert Kreuzer, Jurriaan Hage, and Ad Feelders. A Quantitative Comparison of Semantic Web Page Segmentation Approaches. In *15th International Conference on Web Engineering (ICWE 2015)*, pages 374–391. Springer, 2015. doi: 10.1007/978-3-319-19890-3_24.
- [147] Travis Kriplean, Ivan Beschastnikh, and David W. McDonald. Articulations of Wikiwork: Uncovering Valued Work in Wikipedia Through Barnstars. In Bo Begole and David W. McDonald, editors, *12th Conference on Computer Supported Cooperative Work (CSCW 2008)*, pages 47–56. ACM, 2008.
- [148] Klaus Krippendorff. On the Reliability of Unitizing Continuous Data. *Sociological Methodology*, 25:47–76, 1995. ISSN 00811750, 14679531. URL <http://www.jstor.org/stable/271061>.
- [149] Naveen Kumar. Password in Practice: An Usability Survey. *Journal of Global Research in Computer Science*, 2(5):107–112, 2011.
- [150] Ranjitha Kumar, Arvind Satyanarayan, César Torres, Maxine Lim, Salman Ahmad, Scott R. Klemmer, and Jerry O. Talton. Webzeitgeist: design mining the web. In Wendy E. Mackay, Stephen A. Brewster, and Susanne Bødker, editors, *31st Conference on Human Factors in Computing Systems (CHI 2013)*, pages 3083–3092. ACM, 2013. doi: 10.1145/2470654.2466420.
- [151] Srijan Kumar, Francesca Spezzano, and V.S. Subrahmanian. VEWS: A Wikipedia Vandal Early Warning System. In Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams, editors, *21th International Conference on Knowledge Discovery and Data Mining (KDD 2015)*. ACM, 2015.
- [152] Cynthia Kuo, Sasha Romanosky, and Lorrie Faith Cranor. Human Selection of Mnemonic Phrase-based Passwords. In Lorrie Faith Cranor, editor, *2nd Symposium on Usable Privacy and Security (SOUPS 2006)*, pages 67–78. ACM, 2006. ISBN 1-59593-448-0. doi: 10.1145/1143120.1143129.
- [153] Stacey Kuznetsov. Motivations of Contributors to Wikipedia. *SIGCAS Computers and Society*, 36(2), 2006. doi: 10.1145/1215942.1215943.
- [154] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. Prominent Features of Rumor Propagation in Online Social

- Media. In Hui Xiong, George Karypis, Bhavani M. Thuraisingham, Diane J. Cook, and Xindong Wu, editors, *13th International Conference on Data Mining (ICDM 2013)*, pages 1103–1108. IEEE Computer Society, 2013. doi: 10.1109/ICDM.2013.61.
- [155] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computing*, 1(4):541–551, December 1989.
- [156] Nayeon Lee, Zihan Liu, and Pascale Fung. Team yeon-zi at SemEval-2019 Task 4: Hyperpartisan News Detection by De-noising Weakly-labeled Data. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [157] Stephan Lewandowsky, Laura Smillie, David Garcia, Ralph Herwig, Jim Weatherall, Stefanie Egidy, Ronald E. Robertson, Cailin O'Connor, Anastasia Kozyreva, Philipp Lorenz-Spreen, Yannic Blaschke, and Mark Leiser. *Technology and Democracy: Understanding the influence of online technologies on political behaviour and decision-making*. Publications Office of the European Union, 2020. ISBN 978-92-76-24088-4. doi: 10.2760/709177. JRC122023.
- [158] Xiaoli Li, Tong-Heng Phang, Minqing Hu, and Bing Liu. Using Micro Information Units for Internet Search. In *11th International Conference on Information and Knowledge Management (CIKM 2002)*, pages 566–573. ACM, 2002. doi: 10.1145/584792.584885.
- [159] Jimmy Lin, Ian Milligan, Jeremy Wiebe, and Alice Zhou. Warcbase: Scalable Analytics Infrastructure for Exploring Web Archives. *Journal on Computing and Cultural Heritage (JOCCH)*, 10(4):22:1–22:30, July 2017.
- [160] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *13th European Conference on Computer Vision (ECCV 2014)*, pages 740–755. Springer, 2014. doi: 10.1007/978-3-319-10602-1_48.
- [161] Yunfei Long, Qin Lu, Rong Xiang, Minglei Li, and Chu-Ren Huang. Fake News Detection Through Multi-Perspective Speaker Profiles. In Greg Kondrak and Taro Watanabe, editors, *Eighth International Joint*

- Conference on Natural Language Processing (IJCNLP 2017)*, pages 252–256. Asian Federation of Natural Language Processing, 2017. URL <https://aclanthology.info/papers/I17-2043/i17-2043>.
- [162] Hao Ma, Raman Chandrasekar, Chris Quirk, and Abhishek Gupta. Improving Search Engines Using Human Computation Games. In David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy J. Lin, editors, *18th Conference on Information and Knowledge Management (CIKM 2009)*, pages 275–284. ACM, 2009. doi: 10.1145/1645953.1645990.
 - [163] Jerry Ma, Weining Yang, Min Luo, and Ninghui Li. A Study of Probabilistic Password Models. In *35th Symposium on Security and Privacy (S&P 2014)*, pages 689–704. IEEE Computer Society, 2014. doi: 10.1109/SP.2014.50.
 - [164] Jing Ma, Wei Gao, and Kam-Fai Wong. Detect Rumors in Microblog Posts Using Propagation Structure via Kernel Learning. In Regina Barzilay and Min-Yen Kan, editors, *55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 708–717. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1066.
 - [165] Ling Ma, Nazli Goharian, and Abdur Chowdhury. Automatic data extraction from template generated web pages. In Hamid R. Arabnia and Younghong Mun, editors, *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2003)*, pages 642–648. CSREA Press, 2003.
 - [166] Amr Magdy and Nayer Wanas. Web-based Statistical Fact Checking of Textual Documents. In *2nd International Workshop on Search and Mining User-generated Contents (SMUC '10)*, pages 103–110. ACM, 2010. ISBN 978-1-4503-0386-6. doi: 10.1145/1871985.1872002.
 - [167] Tomohiro Manabe and Keishi Tajima. Extracting Logical Hierarchical Structure of HTML Documents Based on Headings. *Proceedings of the VLDB Endowment*, 8(12):1606–1617, 2015. doi: 10.14778/2824032.2824058.
 - [168] David R. Martin, Charless C. Fowlkes, Doron Tal, and Jitendra Malik. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In *8th International Conference on Computer Vision (ICCV 2001)*, volume 2, pages 416–423. IEEE Computer Society, July 2001.

- [169] Michelle L. Mazurek, Saranga Komanduri, Timothy Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, Patrick Gage Kelley, Richard Shay, and Blase Ur. Measuring Password Guessability for an Entire University. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *2013 Conference on Computer & Communications Security (CCS 2013)*, pages 173–186. ACM, 2013. doi: 10.1145/2508859.2516726.
- [170] Olena Medelyan, David Milne, Catherine Legg, and Ian H. Witten. Mining Meaning from Wikipedia. *International Journal of Human-Computer Studies*, 67:716–754, 2009. doi: 10.1016/j.ijhcs.2009.05.004.
- [171] Benjamin Meier, Thilo Stadelmann, Jan Stampfli, Marek Arnold, and Mark Cieliebak. Fully convolutional neural networks for newspaper article segmentation. In *14th International Conference on Document Analysis and Recognition (ICDAR 2017)*, volume 1, pages 414–419. IEEE, 2017.
- [172] Luis Meneses, Richard Furuta, and Frank Shipman. Identifying "Soft 404" Error Pages: Analyzing the Lexical Signatures of Documents in Distributed Collections. In Panayiotis Zaphiris, George Buchanan, Edie Rasmussen, and Fernando Loizides, editors, *16th International Conference on Theory and Practice of Digital Libraries (TPDL 2012)*, pages 197–208. Springer, 2012. doi: 10.1007/978-3-642-33290-6_22.
- [173] Delia Mocanu, Luca Rossi, Qian Zhang, Marton Karsai, and Walter Quattrociocchi. Collective Attention in the Age of (Mis)Information. *Computers in Human Behavior*, 51:1198–1204, October 2015. ISSN 0747-5632. doi: 10.1016/j.chb.2015.01.024.
- [174] Jose G. Moreno, Yoann Pitarch, Karen Pinel-Sauvagnat, and Gilles Hubert. Rouletabille at SemEval-2019 Task 4: Neural Network Baseline for Identification of Hyperpartisan Publishers. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [175] Osman Mutlu, Ozan Arkan Can, and Erenay Dayanik. Team Howard Beale at SemEval-2019 Task 4: Hyperpartisan News Detection with BERT. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [176] Arvind Narayanan and Vitaly Shmatikov. Fast Dictionary Attacks on Passwords Using Time-space Tradeoff. In Vijay Atluri, Cather-

- ine A. Meadows, and Ari Juels, editors, *12th Conference on Computer and Communications Security (CCS 2005)*, pages 364–372. ACM, 2005. doi: 10.1145/1102120.1102168.
- [177] Duc-Vu Nguyen, Thin Dang, and Ngan Nguyen. NLP@UIT at SemEval-2019 Task 4: The Paparazzo Hyperpartisan News Detector. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
 - [178] Nam P. Nguyen, Guanhua Yan, My T. Thai, and Stephan Eidenbenz. Containment of Misinformation Spread in Online Social Networks. In *4th Annual Web Science Conference (WebSci 2012)*, pages 213–222. ACM, 2012. ISBN 978-1-4503-1228-8. doi: 10.1145/2380718.2380746.
 - [179] Jakob Nielsen and Kara Pernice. *Eyetracking Web Usability*. Pearson Education, 2010. ISBN 9780321714077.
 - [180] Zhiyuan Ning, Yuanzhen Lin, and Ruichao Zhong. Team Peter-Parker at SemEval-2019 Task 4: BERT-Based Method in Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
 - [181] Hartmut Obendorf, Harald Weinreich, Eelco Herder, and Matthias Mayer. Web Page Revisitation Revisited: Implications of a Long-term Click-stream Study of Browser Usage. In *25th Conference on Human Factors in Computing Systems (CHI 2007)*, pages 597–606. ACM, 2007. doi: 10.1145/1240624.1240719.
 - [182] Nick Obradovich, Ömer Özak, Ignacio Martín, Ignacio Ortúñoz-Ortíz, Edmond Awad, Manuel Cebrián, Rubén Cuevas, Klaus Desmet, Iyad Rahwan, and Ángel Cuevas. Expanding the measurement of culture with a sample of two billion humans, Sep 2020. URL osf.io/preprints/socarxiv/qkf42/.
 - [183] Chitu Okoli, Mohamad Mehdi, Mostafa Mesgari, Finn Årup Nielsen, and Arto Lanamäki. *The People's Encyclopedia Under the Gaze of the Sages: A Systematic Review of Scholarly Research on Wikipedia*. SSRN eLibrary, 2012.
 - [184] Niko Palić, Juraj Vladika, Dominik Cubelić, Ivan Lovrencic, and Jan Snajder. TakeLab at SemEval-2019 Task 4: Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.

- [185] Katherine Panciera, Aaron Halfaker, and Loren Terveen. Wikipedians Are Born, Not Made: A Study of Power Editors on Wikipedia. In Stephanie D. Teasley, Erling C. Havn, Wolfgang Prinz, and Wayne G. Lutters, editors, *2009 International Conference on Supporting Group Work (GROUP 2009)*, pages 51–60, 2009. doi: 10.1145/1531674.1531682.
- [186] Olga Papadopoulou, Giorgos Kordopatis-Zilos, Markos Zampoglou, Symeon Papadopoulos, and Yiannis Kompatsiaris. Brenda Starr at SemEval-2019 Task 4: Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [187] Chitra Pasupathi, Baskaran Ramachandran, and Sarukesi Karunakaran. Web Document Segmentation Using Frequent Term Sets for Summarization. *Journal of Computer Science*, 8(12): 2053–2061, 2012. doi: 10.3844/jcssp.2012.2053.2061.
- [188] Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. Automatic Detection of Fake News. *CoRR*, abs/1708.07104, 2017. URL <http://arxiv.org/abs/1708.07104>.
- [189] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep Contextualized Word Representations. *2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*, pages 2227–2237, 2018. doi: 10.18653/v1/n18-1202.
- [190] Plato. *Phaedrus*. Oxford World Classics. Oxford University Press, Oxford, 2002.
- [191] John O. Pliam. On the Incomparability of Entropy and Marginal Guesswork in Brute-Force Attacks. In Bimal K. Roy and Eiji Okamoto, editors, *First International Conference in Cryptology in India (INDOCRYPT 2000)*, volume 1977, pages 67–79. Springer, 2000. doi: 10.1007/3-540-44495-5_7.
- [192] Ingmar Poese, Steve Uhlig, Mohamed Ali Kâafar, Benoit Donnet, and Bamba Gueye. IP Geolocation Databases: Unreliable? *Computer Communication Review*, 41(2):53–56, 2011. doi: 10.1145/1971162.1971171.
- [193] Martin Potthast and Teresa Hofeld. Overview of the 2nd International Competition on Wikipedia Vandalism Detection. In Vivien

- Petras, Pamela Forner, and Paul D. Clough, editors, *Notebook Papers of CLEF 2011 Labs and Workshops*, September 2011. ISBN 978-88-904810-1-7. URL <http://www.clef-initiative.eu/publication/working-notes>.
- [194] Martin Potthast, Benno Stein, and Robert Gerling. Automatic Vandalism Detection in Wikipedia. In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen W. White, editors, *Advances in Information Retrieval. 30th European Conference on IR Research (ECIR 2008)*, volume 4956 of *Lecture Notes in Computer Science*, pages 663–668, Berlin Heidelberg New York, March 2008. Springer. ISBN 978-3-540-78645-0. doi: 10.1007/978-3-540-78646-7_75.
- [195] Martin Potthast, Benno Stein, and Teresa Hofeld. Overview of the 1st International Competition on Wikipedia Vandalism Detection. In Martin Braschler, Donna Harman, and Emanuele Pianta, editors, *Working Notes Papers of the CLEF 2010 Evaluation Labs*, volume 1176 of *Lecture Notes in Computer Science*, September 2010. ISBN 978-88-904810-2-4. URL <http://ceur-ws.org/Vol-1176/>.
- [196] Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A Stylometric Inquiry into Hyperpartisan and Fake News. In Iryna Gurevych and Yusuke Miyao, editors, *56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 231–240. Association for Computational Linguistics, July 2018. URL <https://www.aclweb.org/anthology/P18-1022>.
- [197] Martin Potthast, Tim Gollub, Matti Wiegmann, and Benno Stein. TIRA Integrated Research Architecture. In Nicola Ferro and Carol Peters, editors, *Information Retrieval Evaluation in a Changing World*, The Information Retrieval Series. Springer, Berlin Heidelberg New York, September 2019. ISBN 978-3-030-22948-1. doi: 10.1007/978-3-030-22948-1_5.
- [198] Reid Priedhorsky, Jilin Chen, Shyong (Tony) K. Lam, Katherine Panciera, Loren Terveen, and John Riedl. Creating, Destroying, and Restoring Value in Wikipedia. In Tom Gross and Kori Inkpen, editors, *2007 International Conference on Supporting Group Work (GROUP 2007)*. ACM, 2007. doi: 10.1145/1316624.1316663.
- [199] Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. Truth of Varying Shades: Analyzing Language in Fake

- News and Political Fact-Checking. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 2931–2937. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1317.
- [200] Brenda Reyes Ayala, Mark Edward Phillips, and Lauren Ko. Current Quality Assurance Practices in Web Archiving. Technical report, University of North Texas Libraries, August 2014.
- [201] Victoria Rubin, Niall Conroy, and Yimin Chen. Towards News Verification: Deception Detection Methods for News Discourse. In *Hawaii International Conference on System Sciences (HICSS 2015)*, January 2015.
- [202] Victoria Rubin, Niall Conroy, Yimin Chen, and Sarah Cornwell. Fake News or Truth? Using Satirical Cues to Detect Potentially Misleading News. In *2nd Workshop on Computational Approaches to Deception Detection*, pages 7–17. Association for Computational Linguistics, June 2016. URL <http://www.aclweb.org/anthology/W16-0802>.
- [203] Nick Ruest, Jimmy Lin, Ian Milligan, and Samantha Fritz. The archives unleashed project: Technology, process, and community to improve scholarly access to web archives. In Ruhua Huang, Dan Wu, Gary Marchionini, Daqing He, Sally Jo Cunningham, and Preben Hansen, editors, *Joint Conference on Digital Libraries (JCDL'20)*, pages 157–166. ACM, 2020. doi: 10.1145/3383583.3398513.
- [204] Jeffrey M. Rzeszotarski and Aniket Kittur. Learning from History: Predicting Reverted Work at the Word Level in Wikipedia. In Steven E. Poltrock, Carla Simone, Jonathan Grudin, Gloria Mark, and John Riedl, editors, *15th Conference on Computer Supported Cooperative Work & Social Computing (CSCW 2012)*, pages 437–440. ACM, 2012. doi: 10.1145/2145204.2145272.
- [205] Abdelrhman Saleh, Ramy Baly, Alberto Barrón-Cedeño, Giovanni Da San Martino, Mitra Mohtarami, Preslav Nakov, and James Glass. Team QCRI-MIT at SemEval-2019 Task 4: Propaganda Analysis Meets Hyperpartisan News Detection. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.

- [206] Thaer Samar, Myriam C. Traub, Jacco van Ossenbruggen, Lynda Hardman, and Arjen P. de Vries. Quantifying retrieval bias in Web archive search. *International Journal on Digital Libraries*, 19(1):57–75, March 2018. ISSN 1432-1300. doi: 10.1007/s00799-017-0215-9.
- [207] Andrés Sanoja and Stéphane Gançarski. Block-o-Matic: a Web Page Segmentation Tool and its Evaluation. In *29ème journées "Base de données avancées" (BDA 2013)*, 2013.
- [208] Andrés Sanoja and Stéphane Gançarski. Web Page Segmentation Evaluation. In *30th Annual Symposium on Applied Computing (SAC 2015)*, pages 753–760. ACM, 2015. doi: 10.1145/2695664.2695786.
- [209] Andrés Sanoja and Stéphane Gançarski. Migrating Web Archives from HTML4 to HTML5: A Block-Based Approach and Its Evaluation. In *21st European Conference on Advances in Databases and Information Systems (ADBIS 2017)*, pages 375–393, 2017. doi: 10.1007/978-3-319-66917-5_25.
- [210] Philipp Schaer and Mandy Neumann. Enriching Existing Test Collections with OXPath. In Gareth J. F. Jones, Séamus Lawless, Julio Gonzalo, Liadh Kelly, Lorraine Goeuriot, Thomas Mandl, Linda Cappelato, and Nicola Ferro, editors, *8th International Conference of the CLEF Association (CLEF 2017)*, volume 10456, pages 152–158. Springer, 2017.
- [211] Max Schaub and Davide Morisi. Voter mobilisation in the echo chamber: Broadband internet and the rise of populism in europe. *European Journal of Political Research*, 59(4):752–773, 2020. doi: 10.1111/1475-6765.12373.
- [212] Ralph Schroeder and Linnet Taylor. Big Data and Wikipedia Research: Social Science Knowledge Across disciplinary Divides. *Information, Communication & Society*, 18(9):1039–1056, 2015. doi: 10.1080/1369118X.2015.1008538.
- [213] Saptarshi Sengupta and Ted Pedersen. Duluth at SemEval-2019 Task 4: The Pioquinto Manterola Hyperpartisan News Detector. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [214] Pnina Shachaf and Noriko Hara. Beyond Vandalism: Wikipedia Trolls. *Journal of Information Science*, 36(3):357–370, 2010. doi: 10.1177/0165551510365390.

- [215] Daniel Shaprin, Giovanni Da San Martino, Alberto Barrón-Cedeño, and Preslav Nakov. Team Jack Ryder at SemEval-2019 Task 4: Using BERT Representations for Detecting Hyperpartisan News. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [216] Yuval Shavit and Noa Zilberman. A Geolocation Databases Study. *Journal on Selected Areas in Communications*, 29(10):2044–2056, 2011. doi: 10.1109/JSAC.2011.111214.
- [217] Baoxu Shi and Tim Weninger. Fact Checking in Heterogeneous Information Networks. In *25th International Conference on World Wide Web (WWW 2016 Companion)*, pages 101–102, Republic and Canton of Geneva, Switzerland, 2016. ACM. ISBN 978-1-4503-4144-8. doi: 10.1145/2872518.2889354.
- [218] Craig Silverman, Lauren Strapagiel, Hamza Shaban, Ellie Hall, and Jeremy Singer-Vine. Hyperpartisan Facebook Pages are Publishing False and Misleading Information at an Alarming Rate. <https://www.buzzfeed.com/craigsilverman/partisan-fb-pages-analysis>. BuzzFeed, 2016.
- [219] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556, 2014.
- [220] Robert R. Sokal and Charles D. Michener. A Statistical Method for Evaluating Systematic Relationships. *University of Kansas Science Bulletin*, 38:1409–1438, 1958.
- [221] Peder Sparell and Mikael Simovits. Linguistic Cracking of Passphrases. *IACR Cryptology ePrint Archive*, 2016. URL <https://eprint.iacr.org/2016/246>.
- [222] Alex Spengler and Patrick Gallinari. Document Structure Meets Page Layout: Loopy Random Fields for Web News Content Extraction. In *10th Symposium on Document Engineering (DocEng2010)*, pages 151–160. ACM, 2010. doi: 10.1145/1860559.1860590.
- [223] Vertika Srivastava, Ankita Gupta, Divya Prakash, Sudeep Sahoo, Rohit R. R., and Yeon Hyang Kim. Vernon-fenwick at SemEval-2019 Task 4: Hyperpartisan News Detection using Lexical and Semantic Features. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.

- [224] Thomas Steiner. Bots vs. Wikipedians, Anons vs. Logged-Ins (Redux): A Global Study of Edit Activity on Wikipedia and Wikidata. In Dirk Riehle, Jesús M. González-Barahona, Gregorio Robles, Kathrin M. Mösllein, Ina Schieferdecker, Ulrike Cress, Astrid Wichmann, Brent J. Hecht, and Nicolas Jullien, editors, *International Symposium on Open Collaboration (OpenSym 2014)*, pages 25:1–25:7. ACM, 2014.
- [225] Bozhidar Stevanoski and Sonja Gievská. Team Ned Leeds at SemEval-2019 Task 4: Exploring Language Indicators of Hyperpartisan Reporting. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
- [226] Philip J. Stone, Dexter C. Dunphy, and Marshall S. Smith. *The General Inquirer: A Computer Approach to Content Analysis*. MIT press, 1966.
- [227] Bongwon Suh, Gregorio Convertino, Ed H. Chi, and Peter Pirolli. The Singularity is not Near: Slowing Growth of Wikipedia. In Dirk Riehle and Amy S. Bruckman, editors, *5th International Symposium on Wikis and Open Collaboration (WikiSym 2009)*. ACM, 2009. doi: 10.1145/1641309.1641322.
- [228] Marcella Tambuscio, Giancarlo Ruffo, Alessandro Flammini, and Filippo Menczer. Fact-checking Effect on Viral Hoaxes: A Model of Misinformation Spread in Social Networks. In *24th International Conference on World Wide Web (WWW 2015 Companion)*, pages 977–982. ACM, 2015. ISBN 978-1-4503-3473-0. doi: 10.1145/2740908.2742572.
- [229] Jaime Teevan, Susan T. Dumais, and Daniel J. Liebling. A Longitudinal Study of How Highlighting Web Content Change Affects People’s Web Interactions. In Elizabeth D. Mynatt, Don Schoner, Geraldine Fitzpatrick, Scott E. Hudson, W. Keith Edwards, and Tom Rodden, editors, *28th International Conference on Human Factors in Computing Systems (CHI 2010)*. ACM, April 2010. doi: 10.1145/1753326.1753530.
- [230] Umut Topkara, Mikhail J. Atallah, and Mercan Topkara. Passwords Decay, Words Endure: Secure and Re-usable Multiple Password Mnemonics. In Yookun Cho, Roger L. Wainwright, Hisham Hadad, Sung Y. Shin, and Yong Wan Koo, editors, *2007 Symposium on Applied Computing (SAC 2007)*, pages 292–299. ACM, 2007. doi: 10.1145/1244002.1244072.

- [231] Khoi-Nguyen Tran and Peter Christen. Cross Language Prediction of Vandalism on Wikipedia Using Article Views and Revisions. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2013)*, pages 268–279, 2013. doi: 10.1007/978-3-642-37456-2_23.
- [232] Khoi-Nguyen Tran and Peter Christen. Cross-Language Learning from Bots and Users to Detect Vandalism on Wikipedia. *Transactions on Knowledge and Data Engineering*, 27(3):673–685, 2015. doi: 10.1109/TKDE.2014.2339844.
- [233] Paraskevi Tzekou, Sofia Stamou, Nikos Kirtsis, and Nikos Zotos. Quality Assessment of Wikipedia External Links. In José Cordeiro and Joaquim Filipe, editors, *7th International Conference on Web Information Systems and Technologies (WEBIST 2011)*, pages 248–254. SciTePress, May 2011.
- [234] Udo Undeutsch. Beurteilung der Glaubhaftigkeit von Aussagen. *Handbuch der Psychologie*, 11:26–181, 1967.
- [235] Srinivas Vadrevu, Fatih Gelgi, and Hasan Davulcu. Semantic Partitioning of Web Pages. In *6th International Conference on Web Information Systems Engineering (WISE 2005)*, pages 107–118. Springer, 2005. doi: 10.1007/11581062_9.
- [236] Emmanuel Vincent and Maria Mestre. Crowdsourced Measure of News Articles Bias: Assessing Contributors’ Reliability. In Lora Aroyo, Anca Dumitache, Praveen Paritosh, Alexander J. Quinn, Chris Welty, Alessandro Checco, Gianluca Demartini, Ujwal Gadidaju, and Cristina Sarasua, editors, *1st Workshop on Subjectivity, Ambiguity and Disagreement in Crowdsourcing (SAD 2018)*, pages 1–10. CEUR-WS.org, 2018. URL <http://ceur-ws.org/Vol-2276/#paper-1>.
- [237] Svitlana Volkova, Kyle Shaffer, Jin Yea Jang, and Nathan Oken Hodas. Separating Facts from Fiction: Linguistic Models to Classify Suspicious and Trusted News Posts on Twitter. In Regina Barzilay and Min-Yen Kan, editors, *55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 647–653. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-2102.

- [238] Luis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford. CAPTCHA: Using Hard AI Problems for Security. In Eli Biham, editor, *22nd International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2003)*. Springer, May 2003.
- [239] Kim-Phuong L. Vu, Bik-Lam Belin Tai, Abhilasha Bhargav, E. Eugene Schultz, and Robert W. Proctor. Promoting Memorability and Security of Passwords Through Sentence Generation. In *48th Human Factors and Ergonomics Society Annual Meeting (HFES 2004)*, pages 1478–1482. SAGE, 2004.
- [240] Claudia Wagner, David Garcia, Mohsen Jadidi, and Markus Strohmaier. It's a Man's Wikipedia? Assessing Gender Inequality in an Online Encyclopedia. In Meeyoung Cha, Cecilia Mascolo, and Christian Sandvig, editors, *9th International Conference on Web and Social Media (ICWSM 2015)*, pages 454–463. AAAI Press, 2015.
- [241] William Yang Wang. "Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection. In Regina Barzilay and Min-Yen Kan, editors, *55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 422–426. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-2067.
- [242] Zhongyu Wei, Junwen Chen, Wei Gao, Binyang Li, Lanjun Zhou, Yulan He, and Kam-Fai Wong. An Empirical Study on Uncertainty Identification in Social Media Context. In *51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 58–62. Association for Computational Linguistics, August 2013. URL <http://www.aclweb.org/anthology/P13-2011>.
- [243] Matt Weir, Sudhir Aggarwal, Michael P. Collins, and Henry Stern. Testing Metrics for Password Creation Policies by Attacking Large Sets of Revealed Passwords. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *17th Conference on Computer and Communications Security (CCS 2010)*, pages 162–175. ACM, 2010. ISBN 978-1-4503-0245-6. doi: 10.1145/1866307.1866327.
- [244] Howard T. Welser, Dan Cosley, Gueorgi Kossinets, Austin Lin, Fedor Dokshin, Geri Gay, and Marc Smith. Finding Social Roles in Wikipedia. In *iConference 2011*, pages 122–129. ACM, 2011. doi: 10.1145/1940761.1940778.

- [245] Andrew G. West, Sampath Kannan, and Insup Lee. Detecting Wikipedia Vandalism via Spatio-Temporal Analysis of Revision Metadata. In Manuel Costa and Engin Kirda, editors, *Third European Workshop on System Security (EUROSEC 2010)*, pages 22–28. ACM, 2010. doi: 10.1145/1752046.1752050.
- [246] Ian H. Witten and Timothy C. Bell. The Zero-frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression. *Transactions on Information Theory*, 37(4):1085–1094, July 1991. doi: 10.1109/18.87000.
- [247] You Wu, Pankaj K. Agarwal, Chengkai Li, Jun Yang, and Cong Yu. Toward Computational Fact-checking. *Proceedings of the VLDB Endowment*, 7(7):589–600, March 2014. ISSN 2150-8097. doi: 10.14778/2732286.2732295.
- [248] Jeff Jianxin Yan, Alan F. Blackwell, Ross J. Anderson, and Alasdair Grant. Password Memorability and Security: Empirical Results. *Security and Privacy*, 2(5):25–31, September 2004. doi: 10.1109/MSP.2004.81.
- [249] Diyi Yang, Aaron Halfaker, Robert E. Kraut, and Eduard H. Hovy. Who Did What: Editor Role Identification in Wikipedia. In *10th International Conference on Web and Social Media (ICWSM 2016)*, pages 446–455. AAAI Press, 2016.
- [250] Fan Yang, Arjun Mukherjee, and Eduard Constantin Dragut. Satirical News Detection and Analysis using Attention Mechanism and Linguistic Features. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 1979–1989. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1211.
- [251] Weining Yang, Ninghui Li, Omar Chowdhury, Aiping Xiong, and Robert W. Proctor. An Empirical Study of Mnemonic Sentence-based Password Generation Strategies. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *2016 Conference on Computer and Communications Security (CCS 2016)*, pages 1216–1229. ACM, 2016. doi: 10.1145/2976749.2978346.
- [252] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. Hierarchical Attention Networks for Document Classification. In Kevin Knight, Ani Nenkova, and Owen

- Rambow, editors, 2016 *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016)*, pages 1480–1489, 2016. doi: 10.18653/v1/n16-1174.
- [253] Taha Yasseri, Robert Sumi, and János Kertész. Circadian Patterns of Wikipedia Editorial Activity: A Demographic Analysis. *PLOS One*, 7(1):e30091, 2012. doi: 10.1371/journal.pone.0030091.
 - [254] Chia-Lun Yeh, Babak Loni, and Anne Schuth. Tom Jumbo-Grumbo at SemEval-2019 Task 4: Hyperpartisan News Detection with GloVe vectors and SVM. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
 - [255] Albin Zehe, Lena Hettinger, Stefan Ernst, Christian Hauptmann, and Andreas Hotho. Team Xenophilus Lovegood at SemEval-2019 Task 4: Hyperpartisanship Classification using Convolutional Neural Networks. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.
 - [256] Jan Zeleny, Radek Burget, and Jaroslav Zendulka. Box Clustering Segmentation: A New Method for Vision-based Web Page Preprocessing. *Information Processing & Management*, 53(3):735–750, 2017. doi: 10.1016/j.ipm.2017.02.002.
 - [257] Chiyu Zhang, Arun Rajendran, and Muhammad Abdul-Mageed. UBC-NLP at SemEval-2019 Task 4: Hyperpartisan News Detection With Attention-Based Bi-LSTMs. In *13th International Workshop on Semantic Evaluation (SemEval 2019)*. Association for Computational Linguistics, 2019.