Vom Bulletin Board System (BBS) zur Declaration of the Independence of Cyberspace

BA Hauptseminararbeit

Josias Bruderer

31. August 2021

Abstract

Diese Seminararbeit untersucht die Rolle der Bulletin Board Systems (BBS) in der Entwicklung und dem Austausch des Zeitgeistes der techno-utopischen Community rund um die 1996 formulierte Declaration of Independence of Cyberspace. Dazu wird ein Textkorpus von BBS Textdateien mittels NLP (Worthäufigkeiten inkl. Scattertext, Kategoriengrösse, Dateien pro Jahr, Entitäten, Topic Diversity mittels LDA und NeuralLDA) untersucht und mit der *Declaration* verglichen. Die Analyse belegt die in der Literatur beschriebene Entwicklung der Popularität der BBS. Innerhalb des Datensatzes werden Hinweise auf die *Declaration* gefunden, diese sind aber weniger deutlich als erwartet. Sehr deutlich hingegen ist die grosse Datenmenge an pornografischen Inhalten, die nicht in diesem Ausmasse erwartet wurde.

Autor	Josias Bruderer (josias.bruderer@stud.unilu.ch)
Matrikelnr.	S19-455-070

Seminar	The ABC of Computational Text Analysis
Semester	Frühlingssemester 2021
Studiengang	Universität Luzern, BA Gesellschafts- und Kommunikationswissenschaften
Eingereicht	am 31. August 2021 bei Alex Flückiger und Prof. Dr. Sophie Mützel

Inhaltsverzeichnis

1 Einleitung

> We will create a civilization of the Mind in Cyberspace. May it be more humane and fair than the world your governments have made before.
>
> — *John Perry Barlow, Davos, 8. Feburar 1996.*

Als Antwort auf den *Telecommunications Act* der Vereinigten Staaten von 1996 veröffentlichte John Perry Barlow die Schrift *A Declaration of the Idependence of Cyberspace*. Darin erklärt er den Cyberspace zu einem Ort «where anyone, anywhere may express his or her beliefs, no matter how singular, without fear of being coerced into silence or conformity.» (Barlow 1996) Die *Declaration* wurde zum Symbol einer Bewegung, die auf die Befreiung der Internetnutzer_innen hoffte (Turner 2006: 13), gleichzeitig bildete die *Declaration* einen Mythos des unabhängigen Internets, eines utopischen Raumes, wie er von James Bennett (2015) und Daniel Kreiss (2015) beschrieben wird.

Cyberspace ist allerdings nicht ein Begriff, den es seit der Erfindung des Internets in den frühen 90er Jahren gibt. Bereits vor dem Internet existierten Computer Netzwerke, über welche kommuniziert oder Daten ausgetauscht wurden. Eines davon sind die Bulletin Board Systems (BBS), Server, auf die über einen am Telefonnetz angeschlossenen Computer, unter anderem Texte hoch- und heruntergeladen werden konnten.

Diese Seminararbeit soll der Frage auf den Grund gehen, inwiefern die BBS eine Rolle für die Entwicklung und den Austausch des Zeitgeistes spielte, der sich in der techno-utopischen Community rund um die 1996 formulierte *Declaration* wiederfinden lässt. Die Fragestellung ermöglicht eine Annäherung an zwei aktuelle Diskussionen: Erstens galten BBS als dezentrale und unabhängige Kommunikationsmedien, ähnlich wie dies soziale Medien nach dem Platzen der Dotcom-Blase[1]

[1] Die Ideologie des Web 2.0 und exemplarisch die Entwicklung von WhatsApp vertiefte ich in einer Proseminararbeit, eingereicht im Juli 2020 bei Markus Unternährer. In der vorliegenden Arbeit soll die Zeit vor Web 2.0 behandelt werden.

taten oder heute auf Blockchain basierten Technologien tun. Die Entwicklung der BBS und des Internets verlief allerdings nicht nach den Wünschen der frühen Nutzer_innen, worauf im zweiten Kapitel genauer eingegangen wird. Auch die aktuell vielversprechenden Technologien erfahren Druck von Industrie, Staat und Gesellschaft - ein gutes Beispiel dazu ist die Entwicklung von Bitcoin.[2] Zweitens wird die *Declaration* auch heute noch, 25 Jahre nach deren Publikation, reichlich zitiert oder darauf Bezug genommen, wenn es um Meinungsäusserungsfreiheit, die staatliche Regulierung des Internets oder die Monopolstellungen von einigen wenigen Technologieunternehmen geht.[3] Auch wenn das Dokument heute keine direkte Anwendung findet, scheint es ein prägendes Element der Internetgeschichte zu sein und nicht als Artefakt in den unzähligen Nullen und Einsen zu verstauben.

Um den Hintergrund der *Declaration* zu verstehen, kann der historische Kontext, welcher im zweiten Kapitel behandelt wird, dienen. Eine weitere Herangehensweise, die in dieser Arbeit behandelt und im Mittelpunkt stehen wird, ist die Analyse bestehender Originaldaten (BBS Textfiles). Das methodische Vorgehen dazu wird im dritten Kapitel beschrieben. Darauf folgen die Ergebnisse dieser Analyse sowie deren Interpretation. Abschliessend wird ein Fazit gezogen, einerseits, ob das Vorgehen der Datenanalyse geeignet ist, um die Anliegen der Community rund um die *Declaration* herum besser zu verstehen und andererseits welche Erkenntnisse daraus gewonnen werden können, respektive wie an diese Analyse angeschlossen werden kann.

[2] Artikel und Schlagzeilen dazu gibt es reichlich. Eine passende Parallele zu BBS könnte mit der staatlichen Regulierung des Dienste, wie dies in China geschieht, gezogen werden. Wired berichtet über die Machtverhältnisse rund um Bitcoin: `https://www.wired.com/story/opinion-bitcoins-greatest-feature-is-also-its-existential-threat` (zugegriffen 10.08.2021)

[3] Ein Beispiel wäre Jochen Steinbickers Kapitel «Überwachung und die Digitalisierung der Lebensführung» (2019), in welchem er Bezug zu Edward Snowden und der NSA nimmt und daran die Vorstellung eines offenen und unabhängigen Internets, wie dies die Declaration fordert, destruiert.

2 Untersuchungsgegenstand

Forschung zum Internet und zu dessen Entwicklung wurde reichlich betrieben und ist in unzähligen Texten aus verschiedensten wissenschaftlichen Disziplinen festgehalten. Auffällig dabei ist, wie in der Einleitung beschrieben, dass auf die *Declaration* ab 1996 häufig Bezug genommen wird, während die BBS, bis auf einige technische Leitfäden und die Dokumentation von Jascon Scott (2005), nur marginal in der Literatur erwähnt werden. Im folgenden wird ein Überblick über die BBS und die *Declaration* gegeben. Zudem wird versucht, eine Skizze der ihr zugrunde liegenden Utopie zu zeichnen.

2.1 Bulletin-Board System (BBS)

«The Computerized Hobbyist Bulletin Board System is a personal computer based system for message communication among experimenters. People with terminals or computers equipped with modems call in to leave and retrieve messages.» (Christensen und Suess 1978: 150) So wurde die Idee der BBS von deren Erfindern im November 1978 in der Computerzeitschrift *BYTE* erstmals veröffentlicht. Der Zweck war es, eine Plattform für das Sammeln von Newsletter-Artikel des *Chicago Area Computer Hobbyist Exchange*, anstelle eines Anrufbeantworters, zu bieten. So simpel diese Entwicklung auch klingen mag, «[...] the results were definitely impressive. The original home-brewed internet, BBS was primitive but quickly proved revolutionary.» (Gilbertson 2010) Mit dem Aufkommen von leicht bedienbaren Computern Mitte der Neunzigerjahren stieg die Anzahl der betriebenen BBS in die Zehntausenden (Britannica Academic 2021).

Wichtig zu erwähnen ist, dass die mit der Zeit entstandenen BBS verschiedene Verwendungszwecke hatten: von günstigen E-Mail Plattformen, über Filesharing Plattformen und Softwarepiraterie bis hin zu grossen Dienstleistungsbüros wie beispielsweise CompuServe. Auch Echtzeitkommunikation im Sinne von Telefonkonferenzen oder mehrspieler Games wurden angeboten. (Dvorak und Anis 1990: 208–209, 223) Ebenfalls waren BBS nicht eigenständige und völlig neue Er-

findungen. Einerseits nutzten Sie bestehende Technologien wie das Telefonnetz oder Computer, zudem waren viele ihrer Funktionen keine völligen Neuheiten. Der grosse Unterschied bestand in der vergleichsweise niederschwelligen Vernetzung von Computern über Modems und mit den BBS standardisierte Datenaustauschplattformen (Scott 2005: 00:08:35). Andererseits waren BBS nur ein Teil der Vernetzung. Deren Leistung entfalteten Sie durch Entwicklungen wie das FidoNet, mittels welchem verschiedene BBS verbunden wurden und E-Mail Übermittlungen ermöglicht wurden (Quarterman 1990: 254).

Das World Wide Web löste die BBS ab, die Anzahl betriebener BBS-Server nahm nach dem Höhepunkt Mitte der Neunzigerjahre drastisch ab. Gewisse BBS passten sich an und sind auch heute noch übers Internet erreichbar. Nichts desto trotz erscheinen die BBS als ein wichtiger Teil der Internetgeschichte oder wie Scott es in seinem Dokumentarfilm formuliert: «The BBS helped bring home computers from kits and novelties to required equipment in homes, work, and schools. It could be argued that BBSes helped speed acceptance and growth of the Internet to the size it has become.» (Scott 2005: 00:34:00)

2.2 A Declaration of Intependence of Cyberspace

Die *Declaration* ist einer der Texte, der die Gedanken und Ansichten einer technoutopischen Bewegung widerspiegeln. Referenzen darauf sind in der Literatur zur Internetgeschichte oder zur Diskussion der digitalen Privatsphäre häufig zu finden. Um deren Entstehung und Relevanz nachvollziehen zu können, werden folgend drei Perspektiven zusammengefasst:

Einerseits kann aus einer Perspektive des Raumverständnisses auf die *Declaration* geblickt werden: Der Cyberspace wird in der *Declaration* als ein Raum jenseits dessen betrachtet, was Staaten und Autoritäten zu kontrollieren vermögen, respektive dazu befugt wären. Diese neue Raumimagination, wie dies Manfred Fassler formuliert, entstand aufgrund der globalen Telefonnetze und deren «kulturelle[n] Codierung für Fernanwesenheit oder zeiteinheitliche Fernräume» (2008: 188). Fassler spricht von User Generated Spaces und Cyberlocalism und identifi-

ziert Barlows *Declaration* als einen solchen: «Raum hat immer Autoren und Autorinnen, Macher und Macherinnen. Das war aus meiner Sicht auch die wichtige Nachricht John Perry Barlows, als er in seiner *Declaration of Independence in Cyberspace* gegen die alten Raummächte schrieb und in alter Cyberfreak-Manier den greatefull space feierte, in dem die Kybernetik der Freiheit bzw. Freiheit der Kybernetik herrschte.» (Fassler 2008: 191) Diese Haltung wird auch stark deutlich, in dem Barlow formuliert: «Cyberspace does not lie within your [govermental] borders. Do not think that you can build it, as though it were a public construction project. You cannot. It is an act of nature and it grows itself through our collective actions.» (Barlow 1996)

Diese räumliche Distinktion kann kritisch hinterfragt werden. Damit eröffnet sich eine zweite Perspektive, jene des Mythos der Unabhängigkeit: Entgegen Barlows Haltung, der Cyberspace sei kein öffentliches Projekt, kann argumentiert werden, dass der Cyberspace, und damit exemplarisch auch die BBS, auf Wissen und Infrastruktur basieren und von externen Faktoren beeinflusst werden. «[M]edia spaces are always entangled in economic relations, governmental and regulatory structures, and the workings of institutions.» (Kreiss 2015: 120) Gleichzeitig muss berücksichtigt werden, dass die Ideen der *Declaration* keine neue Vision war. Kreiss nennt exemplarisch die Metapher des «frontiers», die für die 1990 gegründete Electronic Frontier Foundation (EFF) zentral war. Sie setzte sich für die Stärkung des Schutzes der Freiheiten für die Online-Kommunikation ein, unter anderem auch für das Recht auf Privatsphäre von BBS Benutzenden (Britannica Academic 2016). Barlows *Declaration*, die im Namen der EFF verfasst wurde, kann als Versuch gelesen werden, die Interessen der EFF zu manifestieren und öffentlich bekannt zu machen. Wie Kreiss anmerkt, erreichte die *Declaration* zwar grosse kulturelle Reichweite, stellt aber auch ein äusserst umstrittener Gegenstand dar: «Within the scholarly literature, for instance, the Declaration is now often perfunctorily cited as a footnote to early technology culture and its extreme rhetoric [...] and misguided regulatory thinking [...] during the early boom years of Silicon Valley.» (Kreiss 2015: 124) Der Mythos der Unabhängigkeit im Cyberspace und der Kampf dafür bildet trotzdem nach wie vor einen Rahmen für Projekte

wie Wikipedia oder Linux. Dies, in dem er Beteiligten eine Vorstellung über sich selbst und deren Tätigkeit ermöglicht und sie damit motiviert, etwas für die Gemeinschaft beizutragen. (Kreiss 2015: 130)

Als dritte Perspektive soll der historische Kontext rund um die *Declaration* betrachtet werden. Wie bereits erwähnt, war sie eine Antwort auf den Telecommunications Act von 1996 (Britannica Academic 2016; Kreiss 2015: 121; Turner 2006: 13). Damit es dazu kam, war eine breite Vorgeschichte notwendig und fast wichtiger noch, die Etablierung einer Utopie, an welcher die frühen Internet-Nutzer_innen festzuhalten versuchten. Ein Teil der Vorgeschichte ist die Entwicklung der BBS, wie im vorherigen Teilkapitel beschrieben. Vernetzte Computertechnik war allgegenwärtig «and in its shiny array of interlinked devices, pundits, scholars, and investors alike saw the image of an ideal society: decentralized, egalitarian, harmonious, and free.» (Turner 2006: 1) Fred Turner beschreibt in seinem Buch «From Counterculture to Cyberculture» die Entwicklung der Vorstellung einer Bedrohung durch Computer, hin zu einem öffentlichen Verständnis von Computern, die Hoffnung auf eine globale Harmonie versprachen (2006: 2, 5–6). So kam es, dass «[t]hirty years later [1996], the same aspects of computing that threatened to dehumanize the students of the Free Speech Movement promised to liberate the users of the Internet.» (Turner 2006: 13)

Diese Art eines unabhängigen Mediums, dass es vermag, die Benutzenden des Internets zu befreien, muss nach James Bennett (2015) als utopisches Ideal verstanden werden. Utopie ist hier im Sinne einer Vision zu verstehen, die es zu verfolgen gilt und als Ausdruck für eine bessere Lebensweise steht (Levitas 2010: 1; 2003: 4). Und das ist, was die *Declaration* tat, sie «promised the new digital citizens of the online world a commonweal outside the terrestrial, outmoded structures of state, capitalism and ‹old media.›» (Bennett 2015: 6)

2.3 Forschungslücke

Die Abschnitte zu BBS und der *Declaration* stellen nur eine Zusammenfassung eines sehr viel grösseren Diskurses dar. Während die BBS vor allem als Werkzeuge beschrieben werden und ausgehend davon auf eine kleine Gruppe von Computer-Enthusiasten geblickt wird, wird die *Declaration* als eines der Symbole für die Utopie einer grösseren sozialen Bewegung dargestellt. Eine Analyse der effektiven Inhalte der BBS ist nicht zu finden. Annäherungen an die *Declaration* sind fast ausschliesslich historischer Art. Mit der im Zentrum dieser Arbeit stehenden Fragestellung soll ein Einblick in die Inhalte der BBS gewonnen werden und darin nach Auffälligkeiten in Bezug auf die *Declaration* gesucht werden.

3 Methode zur Untersuchung von textfiles.com

Für diese Arbeit konnte ein Archiv mit BBS Text-Dateien ausfindig gemacht werden. Es wird von Scott unterhalten, der nebst Archivar und Historiker Mitwirkender bei archive.org sowie Regisseur vom Film «BBS: The Documentary» ist. Der Datensatz erscheint aufgrund seines Umfanges ($N = 58'000$ Text-Dateien) und Scotts Hintergrund als geeignet, um eine Analyse von BBS Inhalten durchzuführen. Ebenfalls steht ein verkleinerter Datensatz («favorite 100») zur Verfügung, der für die Analyse hilfreich ist.

Die Analyse erfolgt in drei Schritten: Im ersten Schritt wird der verkleinerte Datensatz manuell auf inhaltliche Auffälligkeiten untersucht und daran ein Modell zur Bereinigung des gesamten Datensatzes entwickelt.[4] Anschliessend wird drittens der gesamte Datensatz bereinigt und gefiltert. Abschliessend wird mithilfe von Techniken der NLP (Natural Language Processing) der Textkorpus untersucht und damit die Fragestellung zu beantworten versucht.

[4] Dieser Schritt wurde im Mini-Projekt im Rahmen des Seminars «The ABC of Computational Text Analysis» durchgeführt und ist dokumentiert unter: https://github.com/josiasbruderer/jason-scotts-favorite-100 (zugegriffen 23.08.2021)

3.1 Überblick

Im verkleinerten Datensatz, «a ‹best of› collection of one hundred textfiles that [Scott] think[s] capture the spirit of this site and the unique culture that it attempts to preserve» (Scott o. J.), werden Auffälligkeiten zu Jahr, Länge, Struktur und Inhalt festgehalten und fliessen in den folgenden Abschnitt *Datenbereinigung* ein. Wichtig erscheint an dieser Stelle, dass es sich um durchgehend unstrukturierte Texte handelt. Auch sind die Daten trotz derer Menge «Ergebnisse historisch kontingenter Entscheidungen, wie Auswahl, Formatierung, Kombination etc. und als solche weder neutral, transparent oder objektiv, sondern immer konstruiert» (Mützel u. a. 2018: 112). Dies ist für die Auswertung und das Ziehen von Schlüssen relevant. Die folgenden Punkte geben einen besseren Überblick über den Datensatz:[5]

1. Die **Variation** der Textdateien ist relativ gross und reicht von kurzen witzigen Beiträgen und Unterhaltungsverläufen über ASCII Art bis hin zu detaillierten technischen Instruktionen und Dokumentationen sowie einer Masterarbeit und einem ganzen Buch.
2. In gewissen Textdateien wird ein **«Read X times»** ausgewiesen. Diese Zahlen sind relativ niedrig (meist <100).
3. **Ungültige Zeichen** kommen häufig vor (z.B. «\u1a\u1a\u1a»). Anhäufungen von **Sonderzeichen** dienen zur Formatierung.
4. Verschiedene **Datumsformate** sind zu finden.
5. Es kann nicht davon ausgegangen werden, dass die Textdateien **orthografisch** fehlerfrei sind. Ebenfalls ist mit ***Gunk*** (*«replacing U for You, o for O, Z for S, and similar gunk»*) zu rechnen.
6. **Inhaltlich** kommt von sauber recherchierten Artikeln und Facts bis hin zu wilder Fiktion und Ironie alles vor.

[5] Die Protokollierung der manuellen Untersuchung des Datensatzes ist in Anhang 1: Manuelle Untersuchung Top100 und der Sourcecode in Anhang 2: Sourcecode Top100 zu finden.

7. Textfiles.com führt neben den Dateien auch Titel (inkl. Jahr wenn vorhanden) und Kategorisierung sowie zum Teil eine Beschreibung auf. Diese **Metadaten** können für die Analyse nützlich sein.

3.2 Datenbereinigung

Aus den aufgelisteten Feststellungen wird sodann ein Plan entwickelt, wie die Textfiles bereinigt werden. Das Kapitel «Schöne Daten» von Sophie Mützel et al. (2018) macht deutlich, dass die Datenbereinigung oft nur marginal erwähnt wird, obschon diese elementar ist. Die Bereinigung soll für diese Arbeit daher genau durchdacht und transparent dokumentiert werden.

Bereits beim Herunterladen der Textfiles sollen unpassende Dateien (z.B. Bilder oder Audio) exkludiert werden. Gleichzeitig sollen von den verbleibenden Dateien entsprechende Metadaten gespeichert werden. Dann erfolgt die erste Bereinigung, nämlich das Entfernen von Formatierungen und anderen nicht relevanten Elemente:

- Bereinigung von ungültigen Zeichen (z.B. \x1a)
- Bereinigung von Zeichen, die kein Text repräsentieren
- Bereinigung von Formatierungszeichen (Zeilenumbrüche, Tabulatore)

Anschliessend werden zusätzliche Metadaten generiert:

- Kategorie, Dateiname
- Zeichenanzahl (roh & bereinigt)
- Durchschnittliche Spaltenbreite (roh)
- Anteil von Fliesstext gegenüber Sonderzeichen (roh)
- Jahr des Textfiles (Annahme: zwischen 1960-1999)

Anhand dieser Metadaten kann eine erste Analyse des Datensets gemacht werden.[6] Einerseits sollen ungeeignete Kategorien bereits zu diesem Zeitpunkt aus der Analyse ausgeschlossen werden, andererseits sollen geeignete Parameter für

[6] Dies ist in Anhang 3: textfiles.com Analyse dokumentiert.

die Reduktion des Datensets gefunden werden. Die Analyse kommt zu folgendem Schluss:

- Folgende Kategorien werden ausgeschlossen, da deren durchschnittliches Verhältnis an sinnvollen Zeichen zur Dateilänge (charratioB) unter 0.8 liegt. Das sind Sammlungen von ASCII-Art, Softwarecode, Bilder oder ähnlichem, aber nicht Texte: *tap*, *floppies*, *exhibits*, *artifacts*, *piracy*, *art*, *fidonet-on-the-internet*
- Folgende Kategorien werden aus der Gesamtbetrachtung ausgeschlossen, da deren Charakter jener von Zeitschriften und nicht nutzergenerierter Inhalten entspricht: *magazines*, *digest*
- Dateien, deren Verhältnis von Text (inkl. Satz- und Leerzeichen) zu Dateilänge unter 0.95 sind, werden ausgeschlossen, denn es sollen vor allem Fliesstexte und Unterhaltungen bei der Analyse erfasst werden.
- Dateien, deren Länge zwischen 300 und 30'000 Zeichen liegen, sollen in die Analyse eingeschlossen werden. Kürzere Texte erscheinen kaum aussagekräftig und längere Texte verzerren einerseits das Resultat stark, andererseits erhöhen sie die Dauer der Berechnungen in der Analyse enorm.

Mit der gewählten Filterung verbleiben 5'510 Textdateien, die in den Textkorpus einfliessen.[7] Der Gesamtumfang entspricht ca. 62 Millionen Zeichen. Diese Messwerte sind relevant, da damit die benötigte Rechnerleistung[8] berechnet werden kann (1GB Arbeitsspeicher pro 1 Million Zeichen). Zusätzlich zu den Textdateien werden die *Declaration* und einige charakteristische Textdateien als Datenset in den Textkorpus aufgenommen, damit mit diesen schliesslich vergleiche angestellt werden können.

[7] Die Bereinigung und Filterung des Datensatzes ist in Anhang 4: Sourcecode dokumentiert. Eine genauere Dokumentation inklusive Textkorpus ist online verfügbar: https://github.com/josiasbruderer/bbs-for-independence/ unter 03_workspace und 03_workspace/states (zugegriffen 30.08.2021)

[8] Es steht eine virtuelle Maschine mit folgender Konfiguration zur Verfügung: 24 x Intel Xeon CPU E5-2690 @ 2.90GHz; 200 GB Memory; 250 GB SSD Disk; Manjaro Linux 21.1 (Kernel 5.13.12-1-MANJARO)

3.3 Analyse mittels NLP

Unter Natural Language Processing (NLP) werden Methoden zum maschinellen Analysieren, Modellieren und Verstehen von menschlicher Sprache verstanden (Vajjala u. a. 2020). Allerdings ist dies nicht mit der maschinellen Erfassung von Sinn oder Bedeutung zu verwechseln, wie Emily M. Bender und Alexander Koller (2020) argumentieren. Denn um ein solches Modell zu schaffen, wird eine Software mit einer grossen Menge an Daten trainiert und «if the training data is only form, there is not sufficient signal to learn the relation `M` between that form and the non-linguistic intent of human language users, nor `C` between form and the standing meaning the linguistic system assigns to each form.» (Bender und Koller 2020: 5187) Für gewisse Anwendungsbereiche von NLP hat dies ernstzunehmende Konsequenzen.[9] Es ist aber gerade diese Eigenschaft, dass das Modell nur den Daten entspricht, mit welchen es gefüttert wird, die in der vorliegenden Arbeit zunutze gemacht wird.

Aus dem bereinigten Datensatz soll ein Textkorpus generiert werden, welcher mithilfe der Python Bibliotheken «textacy», «scattertext» und «OCTIS» untersucht wird. Dabei werden folgende Analysen durchgeführt:

- Worthäufigkeiten absolut und Anzahl vorkommende Dateien: Gesamtdatensatz, Datensatz *Declaration*, pro Textfile-Kategorie
- Kategoriengrösse: Gesamtdatensatz
- Entitäten: Gesamtdatensatz, Datensatz *Declaration*
- Scattertext Analyse: Gesamtdatensatz verglichen mit der *Declaration* und dem Datensatz *Declaration*
- Anzahl Dokumente pro Jahr: Gesamtdatensatz
- Topic Diversity mittels OCTIS (LDA und NeuralLDA): Gesamtdatensatz, Datensatz *Declaration*

[9] Exemplarisch am «BERT neural network» wird das beschrieben im Artikel «Machines Beat Humans on a Reading Test. But Do They Understand?», erschienen im Quanta Magazine (Pavlus 2019).

Für die Auswertung mittels OCTIS (Terragni u. a. 2021) wird auf Basis des Textkorpus, welcher für die anderen Analysen genutzt wurde, ein separater Datensatz erstellt. Die Tests werden mittels *OCTIS Dashboard* durchgeführt und die Analyse erfolgt mit den Modellen LDA (Srivastava und Sutton 2017) und NeuralLDA (Blei u. a. 2003).[10]

4 Ergebnisse

Im Folgenden werden die Ergebnisse der Analyse dargelegt. Der Gesamtdatensatz entspricht den bereinigten und gefilterten Textdateien von textfiles.com. Die ersten zwei Analysen *Dokumente pro Jahr* und *Kategoriengrösse* dienen dazu, den Textkorpus zu beschreiben und die Beschaffenheit und Menge an Dateien pro Jahr, respektive pro Kategorie einzuschätzen. Mit den *Worthäufigkeiten* wird eine einfache Analyseart gewählt, um einen rudimentären Vergleich des Textkorpus mit der *Declaration* zu ermöglichen. Die *Entitäten* ähneln den *Worthäufigkeiten*, ermöglichen allerdings eine Analyse, die sich auf bestimmbare Einheiten (Personen, Organisationen, Länder etc.) konzentriert. Die *Scattertext* Visualisierung vereinfacht den grafischen Vergleich der *Worthäufigkeiten*. Schliesslich bietet die *Topic Diversity* Analyse eine Methode zur Generierung von Sprachmodellen und innerhalb davon die Clusterbildung von Themen.

4.1 Dokumente pro Jahr

Die Auswertung der Jahreszahlen der Dokumente spiegelt den Verlauf der Popularität der BBS, analog der in der Literatur beschrieben Entwicklung. Die beiden Messmethoden korrelieren stark ($r = 0.91$), wobei die Methode *lyear* gegenüber *eyear* einen deutlicheren Verlauf zeigt. In der manuellen Untersuchung der Top100 erreichte *lyear* eine höhere Korrelation mit den manuell bestimmten Jahreszahlen

[10] Die Testresultate sind online verfügbar unter: https://github.com/josiasbruderer/bbs-for-independence/tree/main/03_workspace/states/state_OCTIS (zugegriffen 30.08.2021)

als *eyear*, was auch in dieser Auswertung zuzutreffen scheint. Diese Auffallend vielen Textdateien zwischen ca. 1988 und 1995 werden auch die anderen Ergebnisse beeinflussen.

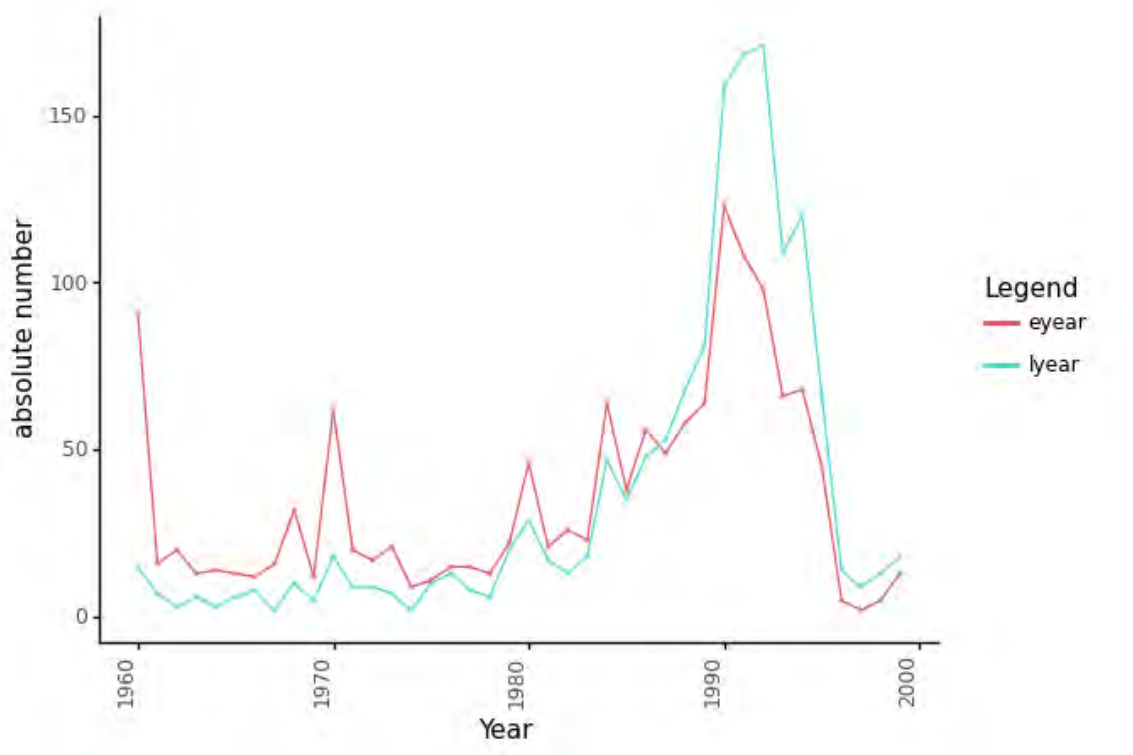


Abbildung 1: Dokumente pro Jahr: Gesamtdatensatz ohne Kategorie «sex»

4.2 Kategoriengrösse

Die Kategorisierung der einzelnen Textdateien wurde auf textfiles.com bereits vorgenommen. In diese Kategorien wird auch der ausgewertete Textkorpus gegliedert, wobei die Kategorie «sex»[11] knapp 40% des Textkorpus ausmacht. Auch Politics (14%) und Occult (7%) kommen oft vor. Die restlichen 40% sind auf 36 Kategorien aufgeteilt. Die Betrachtung der Gesamtzeichenanzahl anstelle der Anzahl Dateien hat keinen Einfluss auf die Reihenfolge.

[11] Die Kategorien werden in Anführungszeichen gekennzeichnet. Die Kategorie «sex» steht für sexuality und nicht Geschlecht, was in den Resultaten auch deutlich hervorgeht.

4.3 Worthäufigkeiten

Die Worthäufigkeiten bestätigen, was in der Kategoriengrösse beobachtet wurde. Während im Datensatz der *Declaration* Wörter wie *world, government, people, computer, system, abuse, technology* (Auswahl aus *Freqency* > 10) vorkommen, sind es beim Gesamtdatensatz *cock, body, people, pussy, ass, girl, woman, tongue, suck, fuck* (Auswahl aus *Frequency* > 7000). Wörter, die der Kategorie «sex» zuzuordnen sind, überwiegen. Abgesehen vom Wort *People* stimmen keine weiteren sinntragenden Wörter in den häufigsten Wörter der beiden Auswertungen überein. Die Zusammenstellung der Worthäufigkeiten des Gesamtdatensatzes ohne die Kategorie «sex» enthält Wörter wie *people, system, world, life, computer, power, government, law, information, change, control, game* (Auswahl aus *Frequency* > 3000). In dieser Auswertung können Wörter gefunden werden, die mit der Auswertung der *Declaration* übereinstimmen: *computer, government, life, people, system, world etc.*

Die Anzahl Dateien, in denen ein jeweiliges Wort vorkommt, erscheint wenig hilfreich zu sein. Die häufigsten Wörter tragen keinen bestimmbaren Sinn, z.B. *time, start, year*, weniger häufige Wörter sind auch in dieser Auswertung durch die Kategorie «sex» verzerrt.

4.4 Enitäten

In der Auflistung der Entitäten fällt auf, dass im Gesamtdatensatz Namen sehr hoch in der Rangliste stehen, gefolgt von Bezeichnungen für den amerikanischen Staat und deren Organisationen (FBI, CIA, NSA). Dies kann auch bei der *Declaration* beobachtet werden, allerdings ist ein Vergleich aufgrund der unterschiedlichen Textmengen kaum sinnvoll. Die Analyse ohne die Kateogire «sex» weisst deutlich weniger Namen auf. Spannender ist das Verhältnis von ausgewählten Begriffen zu der Häufigkeit, mit der FBI (399) oder CIA (486) vorkommt: *Apple* (325), *Constitution* (270), *IBM* (166), *EFF* (80), *Electronic Frontier Foundation* (30), *Fox* (30), *Neidorf* (36), *Radio Shack* (37), *Democratic* (37), *Cyberspace* (29), *Declaration of Independence* (23)

4.5 Scattertext

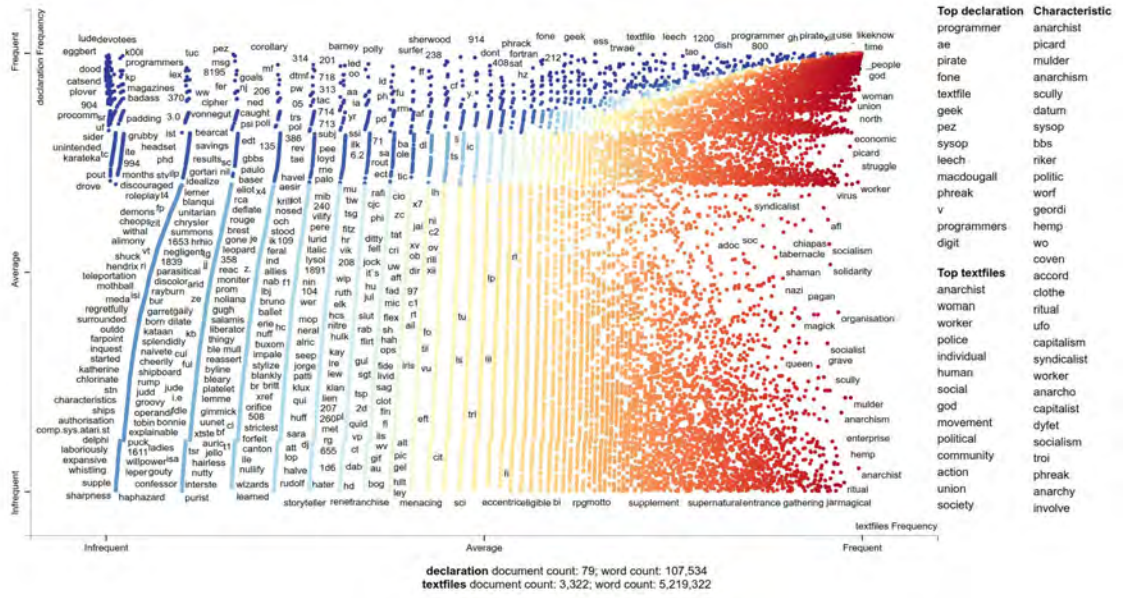


Abbildung 2: Scattertext Visualisierung: Kategorie «100» zu Gesamtdatensatz ohne Kategorie «sex»

Die Scattertext Visualisierung zeichnet erneut ein recht deutliches Bild. Zwischen der *Declaration* und den BBS-Textfiles ist kein diagonaler Trend und somit keine oder kaum eine Korrelation erkennbar.[12] Eine Auswertung ohne die Kategorie «sex» zeigt ein ähnliches Resultat. Wird die Scattertext Visualisierung für den Gesamtdatensatz, verglichen mit der Kategorie «100», generiert, kann eine Korrelation festgestellt werden. Dies deutet auf unzureichende Vergleichsdaten innerhalb der *Declaration* hin, während die Kategorie «100» die notwendige Grösse hat.

4.6 Topic Diversity

Mittels Topic Diversity Analysis wird einerseits der Gesamtdatensatz untersucht und zum Vergleich dazu der Datensatz der *Declaration*. Zur Auswertung wird nach einer Iteration gesucht, bei der die Topic Diversity hoch ist und gleichzeitig die Anzahl Topics ca. 10 beträgt. Insgesamt werden vier Experimente durch-

[12] Zur Interpretation von Scattertext siehe: https://towardsdatascience.com/interpreting-scattertext-a-seductive-tool-for-plotting-text-2e94e5824858 (zugegriffen 30.08.2021)

geführt: je Datensatz (*Declaration* und Gesamtdatensatz) einmal mittels LDA (Srivastava und Sutton 2017) und einmal mittels NeuralLDA (Blei u. a. 2003).

Wie zu erwarten, können in den Experimenten des *Declaration* Datensatzes Topics gefunden werden, die sich um die Schwerpunkte der *Declaration* drehen. Die Topic Diversity ist bei LDA eher niedrig (0.3375), während bei NeuralLDA eine hohe (0.8857) festgestellt werden kann. Dies spiegelt sich auch in den Schlagwortwolken.[13] Während LDA eine ziemlich wortgetreue Abbildung der *Declaration* erstellt, ist diese bei NeuralLDA zwar sinngemäss, aber nicht wortgetreu.

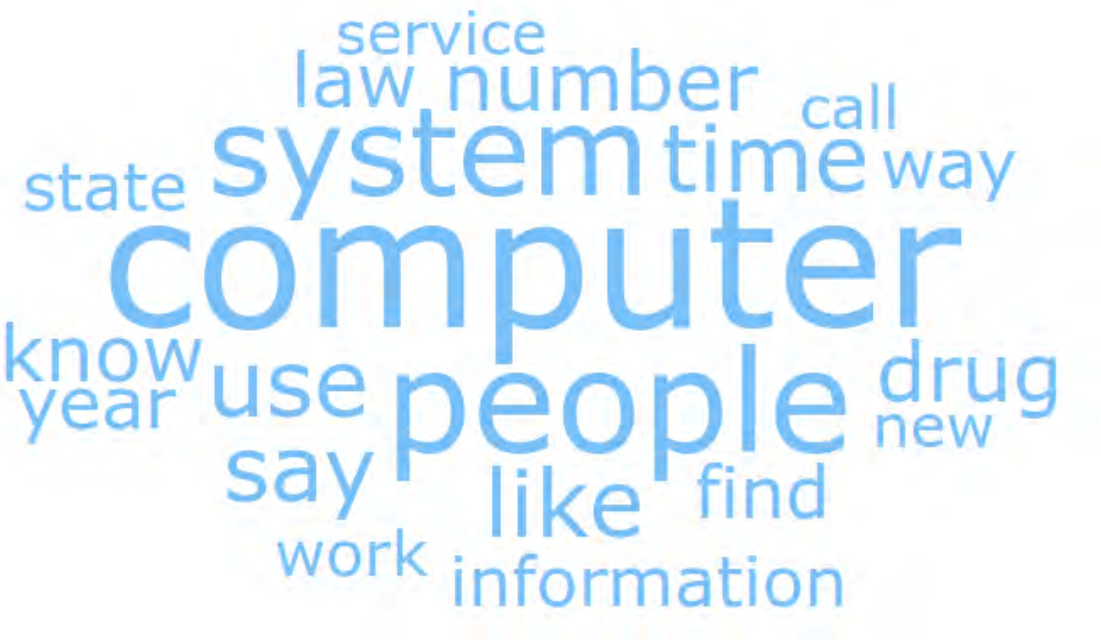


Abbildung 3: Gesamtdatensatz: Topic mittels LDA (Iteration: 3, Run: 0, Topic: 0)

Beim Gesamtdatensatz können mittels LDA ebenfalls Experimente mit eindeutigen Topics gefunden werden (eine davon in Abbildung X). Die Topic Diversity (0.3375) befindet sich in einem ähnlichen Bereich wie bei der *Declaration*. Mittels NeuralLDA können für den Gesamtdatensatz keine Experimente durchgeführt werden, die sinnvolle Topics generieren.

[13] Die Abbildungen der Topics sind zu finden in Anhang 5: Resultate

5 Schlussbetrachtung

Die Analyse von textfiles.com zeigt, dass sich der zeitliche Verlauf der Häufigkeiten der Textfiles mit der in der Literatur beschriebenen Entwicklung der BBS deckt. Dabei erwies sich die Messmethode «lyear» (im Anhang 6: Messmethode lyear beschrieben) in der manuellen Untersuchung der Top 100 sowie beim Gesamtdatensatz als ausreichend genau ($r = 0.91$) um eine Aussage zur zeitlichen Entwicklung zu treffen.

Die Kategorie «sex» macht einen grossen Teil des Gesamtdatensatzes aus und ist in jeder Analysemethode prominent. Dies deutet auf einen thematischen Schwerpunkt innerhalb der BBS hin. Der Vergleich der Entitäten mit und ohne der Kategorie «sex» deutet aufgrund der hohen Differenz an Vornamen darauf hin, dass fiktionale Narrative, Erzählungen, Profile oder ähnliche Inhalte gegenüber sachlichen Informationen oder Diskussionen überwiegen. Die Annahme, dass es sich in der Kategorie «sex» vorwiegend um Pornografie handelt, ist damit naheliegend.

Die Auswertung ohne die Kategorie «sex» zeigt eine Korrelation von Wörtern, die der *Declaration* und dem Gedankengut rund um ihre zentralen Anliegen entsprechen. Die Worthäufigkeit alleine enthält allerdings keinen Kontext und ist daher kaum interpretierbar. Während mittels Scattertext keine deutliche Korrelation erkennbar ist, liefert die Topic Diversity Analyse mittels LDA Evidenz dafür, dass Themen im Bereich des Techno-Utopismus in den Textdateien gefunden werden können. Die Scattertext-Analyse zeigt eine Korreklation bei der Verwendung der Kategorie «100» anstelle der *Declaration* als Vergleichskategorie. Dies deutet darauf hin, dass die Kategorie «100», zumindest teilweise, stellvertretend für den Gesamtdatensatz, betrachtet werden kann. Scotts Beschreibung der Kategorie «100» trifft somit zu: «[it] capture[s] the spirit of this site [textfiles.com] and the unique culture that it attempts to preserve.» (Scott o. J.)

Resümierend ist festzustellen, dass der Zeitgeist im Textkorpus nachweisbar ist, allerdings weniger deutlich als erwartet. Wörter und Themen in Bereich Sexualität sind hingegen sehr prominent. Um auf die Fragestellung zurückzukommen: Stichworte und Themen rund um die techno-utopische Community und *Declarati-*

on können in den BBS Dateien gefunden werden, allerdings weniger als anarchistische Freiheitsbewegung, sondern als Treffpunkt von Computer Geeks, die ihre Gedanken zu Technologie, Gesellschaft und Welt austauschen.

6 Fazit

Im Befund dieser Auswertung liegt eine gewisse Ironie. Während die *Declaration* verlautbaren lässt: «Ours is a world [(cyberspace)] that is both everywhere and nowhere, but it is not where bodies live» oder an einer anderen Stelle «Our identities have no bodies, so, unlike you, we cannot obtain order by physical coercion» (Barlow 1996) wird in den Textdateien viel Aufmerksamkeit der Körperlichkeit und Sexualität gewidmet. Eine der Botschaften aus Avenue Q, «The internet is for porn» (Moore 2003), ist demnach auch bei BBS nicht realitätsfern.

Die Bewältigung der grossen Datenmenge war eine Herausforderung. Obschon dies bereits zu Beginn der Arbeit klar war und dies bei der Programmierung berücksichtigt wurde (Multicore Processing, Speicherung von Zwischenschritten), mussten im Verlauf der Analyse weitere Möglichkeiten zur Filterung implementiert werden. Schliesslich konnte der Korpus von rund 5'000 Dateien auf dem zur Verfügung stehenden Server in akzeptabler Dauer (~2.5 Stunden pro Durchlauf) analysiert werden.

Die grosse Varianz der Textdateien machte die Auswertung anspruchsvoll, wenn auch nicht unmöglich. Die Analyse ermöglichte einige spannende Einblicke in den Datensatz von textfiles.com. Auch wenn sich der Zusammenhang des Gesamtdatensatzes mit der *Declaration* nicht mit einer einzelnen Zahl quantifizieren lässt, konnten Hinweise auf die techno-utopische Community gefunden werden. Deutlich hingegen ist das Resultat der Analyse der Jahreszahlen. Damit wurde innerhalb der Textdateien ein Beleg für die in der Literatur beschriebene Entwicklung gefunden.

Die Analyse der Jahreszahlen könnte auf den kompletten Datensatz von textfiles.com ausgeweitet werden, um ein noch deutlicheres Bild zu erhalten. Auch

wäre es spannend, die Entwicklung der BBS in Bezug auf deren Zugänglichkeit und Niederschwelligkeit zu untersuchen – vor allem in Anbetracht des Gewinnes an Popularität mit dem Aufkommen der Heimcomputer und dem rasannten Popularitätsverlust wegen deren Ablösung durch das Internet.

> Dial the system at (312) 528-7141. When you hear the answer tone, connect your modem, and press return several times until the system determines your transmission rate. If you have a reasonably good telephone line, and a moderately strong modem signal, you should be able to communicate with the system.
>
> — *Ward Christensen und Randy Suess, November 1978, BYTE Magazin*

7 Literatur

Barlow, J.P., 1996. A Declaration of the Independence of Cyberspace [WWW Document]. Electronic Frontier Foundation. URL https://www.eff.org/cyber%73pace-independence (zugegriffen 21.07.2021).

Bender, E.M., Koller, A., 2020. Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data. Gehalten auf der Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, S. 5185–5198. https://doi.org/10.18653/v1/2020.acl-main.463

Bennett, J., 2015. Introduction - The Utopia of Independent Media: Independence, Working with Freedom and Working for Free, in: Bennett, J., Strange, N. (Hrsg.), Media independence: working with freedom or working for free?, Routledge research in cultural and media studies. Routledge, Taylor & Francis Group, New York ; London, S. 117–136.

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022.

Britannica Academic, 2021. bulletin-board system (BBS) [WWW Document]. Encyclopædia Britannica. URL http://academic.eb.com/levels/collegiate/a%72ticle/627505 (zugegriffen 21.07.2021).

Britannica Academic, 2016. Electronic Frontier Foundation (EFF) [WWW Document]. Encyclopædia Britannica. URL http://academic.eb.com/levels/co%6C%6Cegiate/article/627505 (zugegriffen 21.07.2021).

Christensen, W., Suess, R., 1978. Hobbyist Computerized Bulletin Board. Byte Magazine Volume 03.

Dvorak, J., Anis, N., 1990. Dvorak's guide to PC telecommunications. Osborne McGraw-Hill, Berkeley.

Fassler, M., 2008. Cybernetic Localism: Space, Reloaded, in: Döring, J., Thielmann, T. (Hrsg.), Spatial Turn. transcript Verlag, S. 185–218. https://doi.org/10.14361/9783839406830-008

Gilbertson, S., 2010. Feb. 16, 1978: Bulletin Board Goes Electronic [WWW Document]. Wired. URL https://www.wired.com/2010/02/0216cbbs-first-bbs%2Dbulletin-board/ (zugegriffen 10.08.2021).

Kreiss, D., 2015. A Vision of and for the Networked World - John Perry Barlow's A Declaration of the Independence of Cyberspace at Twenty, in: Bennett, J., Strange, N. (Hrsg.), Media independence: working with freedom or working for free?, Routledge research in cultural and media studies. Routledge, Taylor & Francis Group, New York ; London, S. 117–136.

Levitas, R., 2010. The concept of utopia. Peter Lang, Oxford.

Levitas, R., 2003. Introduction: The Elusive Idea of Utopia. History of the Human Sciences 16, 1–10. https://doi.org/10.1177/0952695103016001002

Moore, J., 2003. Avenue Q.

Mützel, S., Saner, P., Unternährer, M., 2018. Schöne Daten! Konstruktion und Verarbeitung von digitalen Daten, in: Houben, D., Prietl, B. (Hrsg.), Datengesellschaft. transcript Verlag, S. 111–132.

Pavlus, J., 2019. Machines Beat Humans on a Reading Test. But Do They Understand? [WWW Document]. Quanta Magazine. URL https://quantamagazine.org/machines-beat-humans-on-a-reading-test-but-do-they-understand%2D20191017/ (zugegriffen 25.08.2021).

Quarterman, J.S., 1990. The matrix: computer networks and conferencing systems worldwide. Digital Press, Bedford, Mass.

Scott, J., o. J. Jason Scott's Top 100 Textfiles [WWW Document]. T E X T F I L E S. URL http://textfiles.com/100/ (zugegriffen 16.05.2021).

Scott, J., 2005. BBS: The Documentary.

Srivastava, A., Sutton, C., 2017. Autoencoding Variational Inference For Topic Models. arXiv:1703.01488 [stat.ML].

Steinbicker, J., 2019. Überwachung und die Digitalisierung der Lebensführung, in: Stempfhuber, M., Wagner, E. (Hrsg.), Praktiken der Überwachten: Öffentlichkeit und Privatheit im Web 2.0. Springer VS, Wiesbaden, S. 79–96.

Terragni, S., Fersini, E., Galuzzi, B.G., Tropeano, P., Candelieri, A., 2021. OCTIS: Comparing and Optimizing Topic models is Simple!, in: Proceedings of the

16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations. Association for Computational Linguistics, Online, S. 263–270.

Turner, F., 2006. From counterculture to cyberculture: Stewart Brand, the Whole Earth Network, and the rise of digital utopianism. University of Chicago Press, Chicago.

Vajjala, S., Majumder, B., Gupta, A., Surana, H., 2020. Practical natural language processing: a comprehensive guide to building real-world NLP systems, First edition. ed. O'Reilly, Beijing Boston Farnham Sebastopol Tokyo.

8.1 Anhang 1: Manuelle Untersuchung Top100

file	size	csize1	csize2	column	ratioA	ratioB	year	eyear	lyear	type	year	structure	content	others	descByScott
914bbs.txt	3968	3896	3968	56.19	0.74	1.77	1984	1984	1984	textfile	1984	table (separated by /)	BBS Numbers		914 Area Code BBS List, by Dan Gelman (January 15, 1984) A snapshot of the typical BBSes you might find in an area code, in this case, mine. A good portion of the 'General' boards you see listed were in fact Phreak or Pirate boards. Keeping an active account on all your local systems could be quite time-consuming.
actung.hum	332	323	332	39.25	2.77	8.49	nd.	nd.	nd.	textfile	nd	text	funny poem	Relaxen und watch das blinkenlights...	Relaxen und watch das blinkenlights... This sort of small, quaint humor file could be found lurking across many different kinds of BBSes and mainframes. Origin: Unknown, although it very likely could date back to the 60's or 70's.
ad.txt	1664	1629	1664	53.07	0.51	1.55	nd.	nd.	nd.	textfile	nd	tag file			Call The Upside Down BBS! A typical "Tag File" for a Bulletin Board System, in this case a classic Apple II with 64k of memory. To entice you over, the BBS offers you everything up to and including the two floppy disks located in the floppy drives. Besides being an interesting approach for a BBS ad, this short file also shows the variety of devices you could hook to an Apple II, including devices you could hook to other devices.
adventur.txt	7758	7627	7758	66.34	1.72	17	nd.	nd.	nd.	textfile	nd	text	Walkthrough: Crowther and Wood's original classic "Adventure"	https://quuxplusone.github.io/Advent/	Adventure: Solving it in Easy Steps, by The Rom Raider and Doctor Digital Don't read this file if you haven't played Crowther and Wood's original classic "Adventure"! This file is a solid example of a "Walk-through", where the goal was to present an easy, no-thinking solution to the classic thinking person's game: text adventures. While these games could present hours (or days or weeks) of fun trying to solve the puzzles and pitfalls, many people were content to just be given the answer and go through the game blindly, watching as every step they made was the exact right one. To a smaller degree, there was a constant one-upsmanship with Walkthroughs, where whoever could come out with the "solve" for a game the soonest after it was released (or even before) was the King of the Hill.
angela.art	6656	6546	6656	59.05	0.13	4.61	nd.	nd.	nd.	textfile	nd	ASCII Art	naked woman		ASCII Art of "Angela" A solid, classic example of an ASCII Nude, brimming with joy and text-based sexiness. Some of these were hand-drawn, while others used primitive digitizers and software that translated graphics to text to give surprisingly realistic photos when seen from a distance. Naturally, these files were a hot trade online.
anonymit	34657	34657	34657	52.12	2.77	17.83	1960-1992	1960	1992	textfile	1992	text	Article about protection of the writer from general harassment and investigation		The Joy of Handles, by Mahatma Kane Jeeves and David Lescohier This series of articles attacks the issue of anonymity and handles from a completely different set of perspectives; that is, the protection of the writer from general harassment and investigation, and not necessarily that of promoting unwelcome or illegal ideas. An informative read.
applemaf.hum	21388	20896	21388	54.36	1.81	6.26	1979-1986	1979	1986	textfile	1986	text, messages	insight into Apple piracy world	Sherwood Forest BBSes → influential force	The Apple Mafia Story, as told to Red Ghost, 1986 This interesting insight into the comings and goings of the Apple piracy world of the early-mid 80's shows the battle between the older class of pirates and the new breed of "r0dentz" that has been waged for the last 20 years. This file also gives a history (and hardware list) of the Sherwood Forest BBSes, which were among my all-time favorite boards, and probably a pretty darn influential force in the world that textfiles.com presents.
apples.txt	8960	8548	8960	30.49	1.15	4.34	1985	1985	1985	textfile	1984	messages	apple "warez" message base	Format: «MESSAGE #12:»	Typical Apple Piracy Message Base, circa 1984 This pristine capture of a 14-message apple "warez" message base shows a gamut of user types converging in one place to trade boasts, information, and programs. From Sherlock Apple's boast of "I have em all!!!!!" to Creative Cracker and Key Master's BBS ads, you can see how these places became hotbeds of activity and information. Key master and I traded textfiles back then; I thought nothing of calling a BBS called "The 4th Reich".
arttext.fun	4020	3910	4020	46.34	1.45	4.92	nd.	nd.	nd.	textfile	nd	text	guide to write textfiles		The Art of Writing Textfiles, by The Bronze Rider Bronze rider weighs in with his opinions on how to write proper textfiles, probably in response to some lack of quality in files up to that point. (This file is incomplete for the moment, but you'll get the idea.)
b00g!.hum	6623	6496	6623	60.66	1.69	5.64	1986	1986	1986	textfile	1986	text	essay about boog	Anarchy Inc. - important player	B00g and the Art of Zen, by Anarchy Incorporated This file started a weird "b00g" craze that perpetuated itself for a number of years across a lot of BBSes that I was on/involved in. Then again, Anarchy Inc. was one of those groups you could depend on for some really excellent writing no matter what the subject was about.
balls.txt	2845	2781	2845	60.38	1.57	6.47	1990	1990	1990	textfile	1990	text	how to clear customer balls ;-)	Example of BBS Disclaimer	Mouse Balls Available as a Field Replacement Unit A classic example of a somewhat plausible file making the rounds for years and years. This likely-true file discussing how to wash the balls from Computer Mice took on a life of its own and still shows up occasionally. Surely a giggle, if not a guffaw.

basicom4.phk	17717	17308	17717	55.05	1.96	7.63	1978-1990	1978	1990	textfile	1984	text	Clean Content, describing Bell Telephone System		Bioc Agent's Telecommunications Series, Part IV BIOC Agent 003 was one of those rare phone phreaks who could both assimilate information around him and present it in a well-written, forthright manner. Often, many of the phreaking textfiles of the time were poorly written, hastily formatted, and lacking in any perspective beyond how to break or get freebies from a computer or network. Bioc got a lot of attention with his clear writing and informative series "The Basics of Telecommunications", which appeared in the summer of 1984. This example from the series, part 4 (of 7) covered both the hierarchy and electronic network of the Bell Telephone System. Groundbreaking.
basicom5.phk	18867	18483	18867	57.64	1.87	8.87	1984	1984	1984	textfile	1984	text	Clean Content, describing Telephone System, bibliography		Bioc Agent's Telecommunications Series, Part V Another example of BIOC's writing in the Basics of Telecommunications Series, this section dealt entirely with the basic telephone, including the wiring and the electronic aspects. Notably, BIOC gives a bibliography where he got a lot of his information (something pretty much not done beforehand) and additionally covers the theory of operation of the infamous "Black Box", as well. Excellent.
bbsdeath.pro	7639	7490	7639	63.83	2.85	15.79	1977	1977	1977	textfile	1982	text	Downturn of BBS	What has killed bbs systems? - bbs only for program upload and download?	Whatever Happened to REAL Bulletin-Board Systems? What strikes me about this file was that it was written around 1982 and decries how out-of-touch, vicious, and impersonal the BBS world has grown for the writer. Note the interesting reasons he gives for the downturn of BBS's.
bc760mod.ham	1408	1379	1408	51.85	1.78	4.63	nd.	nd.	nd.	textfile	1988	messages	VHF/UHF RADIO Hacking Guide		Modification of the Uniden Bearcat BC950XLT for Cellular Frequencies, by John Stover (March 29, 1988) The problem was major: Cellular phones could be listened to by any ham radio scanner being made. The normal solution: Produce better cell phones, that did a better job of encrypting/scrambling transmissions. The solution the government took: Force all creators of any ham radio scanners to remove the capability of ham radios to listen to the frequencies chosen for the cellular phones. Naturally, the companies did this in the cheapest way possible, often just running a short-circuiting wire such that attempts to go to those frequencies would be unsuccessful. And naturally, files such as this one made themselves available, where you were told how to take out that wire and restore the machine to full functionality. Was the point to listen in on people? No. The point was crippling technology to hide things from people flies in the face of the spirit of technology. 5 short lines, and the efforts from the unknowing are thwarted. The power of textfiles.
beefstar.hum	2965	2890	2965	55.2	2.27	11.15	nd.	nd.	nd.	textfile	nd	text	poem about fucking beef...		"Hey how much for someone to, you know, screw with the beef?" This is the only beef pornography I've ever seen. Sexual Surrealism at its best.
bhbb1.hac	9837	9837	9837	33.54	1.93	7.42	1980	1980	1980	textfile	1985	text	Guide to use «blue box»		Better Homes and Blue Boxing, Part I, by Mark Tabas (January 7, 1985) Mark Tabas came along in 1985 and wrote a series of chatty, friendly files about all the fun you could have with the mysterious Blue Box, the most famous of all the Phreak Boxes that rose out of the 1970's and 1980's. This box, when used in conjunction with a 2600hz tone across a phone line, allowed you to seize control a telephone line as if you were an operator and do all sorts of neat, crazy stuff. By 1985 these boxes were becoming obsolete (with the advent of Electronic Switching System, or ESS) but this file harkens back to this interesting era. A sign of the great works LOD would create for the next decade.
billrights.fun	3746	3659	3746	56.7	3.88	18.41	nd.	nd.	nd.	textfile	1993	text	re-tooled Bill of Rights	**Take a closer look and compare to declaration of cyberspace!** https://www.eff.org/c	The Bill of Rights "Lite", by John Perry Barlow This re-tooled Bill of Rights, rephrased to reflect most of the constitutional issues arising in cyberspace and in general everyday life, hit the nail on the head as to how far the government had strayed from its original plan. Barlow, a co-founder of the Electronic Freedom Foundation, makes his beliefs about the modern world known in just a couple screenfuls of slashed-up constitutional law.
black.box	4460	4314	4460	34.45	1.06	4.97	nd.	nd.	nd.	textfile	nd	text	Guide to use «black box» free calling		To All Who Dare -- The Black Box For many people, this simple little text file was the opening door into the world of the Phone Phreak, a world where a simple application of technology meant a subversion of the great and powerful Bell System. In this case, the Black Box would convince the telephone company that your phone was still ringing, even though you'd picked the phone up and were chatting happily through the buzzing rings. With its name owing to the 1970's era "Blue Box", the Black Box was the final spark to ignite a stream of steady "box" files, each one a more flamboyant and wild color and each promising the world.

bofh.1	3854	*3737*	*3854*	*50.99*	*2.28*	*10.5*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Story: Simon the Sysadmin who would torture and ruin his users		The Bastard Operator From Hell #1, by Simon Travaglia As the BBS kids of the world grew into full-fledged System Operators, they found that no textfile out there caught the frustrations and issues of a computer hacker saddled with responsibility more than the BOFH (Bastard Operator from Hell) series. Stretching through many files and continuing to this day as a magazine column, these textfiles set out an alternate-world Simon the Sysadmin who would torture and ruin his users in the pursuit of more free time and lager. The initials BOFH have become one of the better-kept inside jokes of the System Admin trade, and these files have become immortal.
captain.phk	2759	*2759*	*2759*	*62.86*	*1.55*	*5.23*	*1984*	*1984*	*1984*	*textfile*	1984	text	Repost Article from Infoworld		An Apple for the Captain, by Steven Wozniak (October 1, 1984) BIOC Agent 003 transcribes an Infoworld article that mentions a funny story about Captain Crunch (John Draper), an employee of Apple, reprogramming an Apple II so that it would dial up PBX lines to get free phone codes. In a few short paragraphs, Steven Wozniak describes Phone Phreaking with an innocent sense of fun and exploration, using common technology.
captmidn.txt	24394	*23981*	*24394*	*68.91*	*2.74*	*17.22*	*1985-1987*	*1985*	*1987*	*textfile*	1986	text	Article about overroding HBOs satellite signal		The Story of Captain Midnight This textfile, source unknown, tells the story of Captain Midnight, a lone satellite operator who overrode HBO's signal with a warning against charging $12.95 a month and scrambling their signal. This sudden seizure of the HBO signal caught the media (and the government)'s attention, and he was soon caught. This textfile saves the memory of a fellow who took matters into his own hands, and blew out a showing of Pee Wee's Big Adventure as a bonus.
catstuff.app	9818	*9610*	*9818*	*50.82*	*1.24*	*2.86*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Guide to use «Apple Cat II»		Expanding Your Apple Cat II, by The Ware-Wolf The Novation Apple Cat Modem was one of those pieces of technology that you just can't believe ever got out into the market, and which stands as a straight example of the creativity that lives in this world. Built simply to be a flexible modem, this piece of technology contained 4 digital to analogue converters and several other unexpected ports and switches that caused it to be used as a clock, answering machine/voice mailbox, hold button, voice changer, and music player. Simply put, this modem was beyond belief. This textfile helps show some of the amazing modifications to this modem that were devised by its users.
cDc-0200.txt	124155	*124155*	*124155*	*56.83*	*1.96*	*11.99*	*1964-1993*	*1964*	*1993*	*textfile*	1992	text, messages	«but for the way these parodies perfectly capture a lot of the cultural forces that ran through that time. (The warez vs. textfiles debate, the self-aggrandizement of older hackers, the completely bizzare spelling styles, etc.)»	**Take a closer look and compare to declaration of cyberspace!**	The cDc #200 Higgledy-Piggledy-Big-Fat-Henacious-Mega-Mackadocious You-Can't-Even-Come-Close-So-Jump-Back-K-BOOMIDY-BOOMIDY-BOOM File, by Swamp Ratte' The Cult of the Dead Cow continues to be a prominent force in the online world, but when they started in the late 1980's they were just another text-file writing group, copying heavy metal lyrics and printing bomb ingredient lists. Swamp Ratte's perseverance and leadership caused cDc to break out of that mode, however, and by the time they'd released their 200th textfile (In December of 1992) they'd been around for over 6 years, forever by BBS standards. To celebrate, Swamp Ratte' wrote this file, which I consider to be an all-time classic not only because of the dead-on parodies of the BBS world of the 1980's that run through it, but for the way these parodies perfectly capture a lot of the cultural forces that ran through that time. (The warez vs. textfiles debate, the self-aggrandizement of older hackers, the completely bizzare spelling styles, etc.) This file truly goes above and beyond in every way. A great reference file to see if you can get all the jokes when reading other files here.
codegeek.txt	20841	*20393*	*20841*	*53.61*	*1.4*	*8.77*	*1993-1993*	*1993*	*1993*	*textfile*	1993	text	Guide to «Code of the Geeks»		Robert Hayden's Code of the Geeks v1.01 The Geek Code is one of those bizzare Internet-only phenomenons that would only rise up among a culture dominated by the intelligensia: a code that, through an application of letters with plus or minus modifications, would indicate the hobbies, desires, or public aspects of that person, easily machine-readable, but to anyone who didn't know the code, completely indecipherable.
copyprot.pro	11392	*11218*	*11392*	*71.25*	*3.58*	*24.83*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Article about Copy-Protection		Copy Protection: A History and Outlook Dt writes a quick overview (intended for publication, and therefore a little more balanced than it normally would have been) about the history of copy protection and some of the methods used on both sides in the war over software.
crossbow	29200	*28649*	*29200*	*56.22*	*2.68*	*16.38*	*1984-1992*	*1984*	*1992*	*textfile*	1992	text	Speech about Cryptography	**Take a closer look and compare to declaration of cyberspace!**	From Crossbows to Cryptography: Thwarting the State Via Technology Chuck Hammill of the Libertech Project comes out with a jaw-dropping defense of technology as a liberating force, through the use of cryptography and communication, and applies it to his (cynical) view of history and the nature of Governments. A speech given at the Future of Freedom Conference that is at once balanced, intruiging, revolutionary, bitter, hopeful, and inspiring. All around, ahead of its time and relevant to this very minute.

dec.wars	31839	*31839*	*31839*	*59.6*	*2.41*	*9.9*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Story: ADVENTURES OF LUKE VAXHACKER		DEC WARS: The Continuing Saga of the Adventures of Luke Vaxhacker One of the earlier and one of the best cross-cultural fan fiction files, combining the world of Digital's VAX series of computers with the Star Wars movies. Peppered throughout this file, tons of inside VAX jokes combine with Star Wars references, making it one of the geekiest, nerdiest files you could come across online. This genre has exploded out of control since then, but at the time, it was something really new, and a ton of fun.
diskgone.ana	627	*614*	*627*	*50.08*	*1.85*	*4.65*	*nd.*	*nd.*	*nd.*	*textfile*	1985	text	Floppydisk Bomb		When He Boots It, It Boots Him! From Ziggy Stardust This explosive device sticks in the mind because of both the pure nastiness of the situation (booby trapping a floppy disk to turn it into a bomb) and the reason given for a person to risk someone else's life: they didn't trade pirated programs honestly.
dodontae.hum	9385	*9211*	*9385*	*63.2*	*2.28*	*8.41*	*1985*	*1985*	*1985*	*textfile*	1986	text	Guide to Ascii Express	sharing/warez	The Do's and Don'ts of Ascii Express, by Quasimoto The story of Ascii Express is one of a telecommunications company adding a small feature to allow remote downloads, that spread into a massive underground network of pirated applications throughout the Apple II community. These "AE Lines" provided quick, simple access to other floppy drives across the country, and became a subculture all their own. This file purports to give some suggested etiquette for AE lines, only to be deflated quite humorously by Count Nibble at the end.
easymony.ana	6709	*6564*	*6709*	*55.34*	*2.25*	*7.97*	*1986-1986*	*1986*	*1986*	*textfile*	1993	text	Guide to easy money: drugs, girls, pick-pocketing	No-man`s land between fact and fantasy	A Guide to Easy Money, by The Flash (January 4, 1986) Even a cursory read of this file shows that the Flash knew not one molecule of what he was talking about. This complete lack of knowledge in the dark arts of Street Economy obviously didn't stop him from publishing a series of files on how to succeed in them. At this no-man's land between fact and fantasy, you get a great insight into the author's idea of how the world works, and how easy he thought the world of crime was. (Ostensibly, the Flash has gone on to a nice, quiet life somewhere.)
eatingmsh.drg	2028	*2028*	*2028*	*40.1*	*2.67*	*10.46*	*1993*	*1993*	*1993*	*textfile*	1992	messages	Magic Mushrooms on Pizza		Can You Put Psychedelic Mushrooms on Pizza? A pretty funny example from a Usenet posting in alt.drugs. Somehow, I can imagine this happening. And oh, he's the MANAGER! While a lot of drug files tend to be boring chemical lists or long and drawn-out philosophical discussions, this file makes you think twice about who's working the cash register.
eel_bye.txt	7680	*7543*	*7680*	*64.4*	*1.26*	*7.62*	*1992*	*1992*	*1992*	*textfile*	nd	text	Goodby message of «The Eel» → reflecting the downside of pirating	**Take a closer look and compare to declaration of cyberspace!**	The Eel Says Goodbye to the Pirate World What really strikes me about The Eel's farewell to the piracy world of 1992 is that while he says that one of his primary motivations for leaving is his current circle of "real" friends, the rest of the file goes to show he has dozens of other "real" friends as well. No doubt in the years after his break away from life on the modem he's built even more circles of friends, but one can't help wondering if he doesn't read this file and think of what else he threw away besides his collection of "warez".
elite.cmd	5946	*5748*	*5946*	*34.9*	*1.78*	*6.02*	*1982*	*1982*	*1982*	*textfile*	nd	text	Guide to be Elite		The Elite Commandments As the word "Elite" came to be bandied about in BBSes, people started to separate themselves between the "Elite" and "Unelite". Specifically, this was just another way to look down on others based on completely arbitrary, meaningless reasons. This file skewers that attitude in a list of "commandments" that best represent the mindset of the self-named "elite". As a bonus, several inside jokes from the era are presented in a "gossip weekly" parody at the end.
elites.txt	3113	*3055*	*3113*	*51.67*	*3.07*	*16.29*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Young BBS User raging and complaining		Someone Completely Blows Up A young BBS user (I don't know where this came from) suddenly begins ranting about everything that bothers him about being on BBSes. His complaints take on a heartwarming quirkiness, looking back.
ethics.txt	8140	*7987*	*8140*	*59.79*	*2.84*	*19.3*	*nd.*	*nd.*	*nd.*	*textfile*	1995	text	nice formated guidelines for BBS usage		Ethics for BBS Users A well-written, nicely-formatted, completely pedantic file that lectures you on every aspect of being a BBS user. This file was part of a trend of Sysops explaining to users how great they had it for having BBSes to call, and to appreciate the work behind them. They were rarely successful, but you do what you can. The invitation to download the file and display it on other BBSes meant that some new users would be subjected to this file automatically. The "wearing a tie to school" side of the BBS world.
famous.bug	52609	*51527*	*52609*	*55.93*	*2.88*	*11.01*	*1961-1985*	*1961*	*1985*	*textfile*	1995	text	computer bugs listed		Famous Computer Bugs, compiled by Dave Curry and John Shore This ARPAnet-compiled lists of computer glitches through history shows some wonderful perspective on disasters and screw-ups through history (mostly the 60's and 70's) and shows you the interesting vulnerabilities that have cropped up over time. Some of them, such as a probe suddenly losing contact with earth, are scarily sobering, but others, such as the Multics bug (the swapper-out process would swap out the swapper-in process!) make you just want to snort, assuming you snort at that sort of thing. Geeky, and cute.

feh-1	54625	*54622*	*54625*	*44.82*	*2.12*	*7.99*	*1995-1995*	*1995*	*1995*	*textfile*	1992	text, messages	Hacker Magazin		Fuckin' Eleet Haxor Issue #1, July 1st 1995 As the World Wide Web really started to take hold in the middle of the 1990s, textfiles became a rarer and rarer entity, usually leaning instead to HTML pages and graphics to get the information (and the point) across. In the case of FEH, indicative of the textfiles of the time, it is sometimes very hard to tell where the parody and where the seriousness lay within the issue. While a lot of it seems to be a thought-out send-up of the badly-spelling hacking community, some serious and researched information is included as well. This magazine went on to several additional issues, each of them a little more serious than the last. For better or worse, this is how things came to be in the culture.
gems.txt	75172	*73969*	*75172*	*65.2*	*2.65*	*13.26*	*1960-1992*	*1960*	*1992*	*textfile*	1989	text	Article about Gems	very nice formatted!	GEMS: The Untold Story, by the Video Vindicator (February 1, 1992) The Video Vindicator came late into the game (early 1990's) but produced some of the most wonderful files from that time period. This file caught my attention because he chose a subject that would normally be of very little interest to the BBS crowd (Gemology) and takes it to a completely new level by turning it into both a wonderful history lesson about Gems (I learned a lot in this file) and twisting it into yet another way to scam the planet for some extra bucks. Breathtaking in his audacity, and completely slick in his delivery. One to watch.
groupass.phk	1809	*1763*	*1809*	*55.39*	*2.93*	*11.65*	*nd.*	*nd.*	*nd.*	*textfile*	1989	messages	Funny description of an issue in telephone system (dog as grounding)		
hack_ths.txt	148109	*148109*	*148109*	*60.42*	*1.21*	*21.25*	*1971-1989*	*1971*	*1989*	*textfile*	1984	text	Thesis: The social organization of the computer underground by GORDON R. MEYER	**Take a closer look**	The Social Organization of the Computer Underground, The Thesis of Gordon Meyer Mr. (Probably Dr.) Meyer's Sociology paper on the social structure in Hacking, Phreaking, and Pirating groups is interestingly dedicated to George Hayduke (writer of the "Get Even" series of revenge books) and Barry Freed (The pseudonym taken by Abbie Hoffman when he was a fugitive from the FBI). This indicates not a little leaning on the side of the groups he's analyzing. He goes over impressions that the media have about hackers, what being a hacker, phreak or pirate consists of as he sees it, and then tries to draw conclusions of what this all means. Written in 1989, this file takes on the subculture with a non-hysterical point of view that makes it very easy to read. Worth the time.
hack1.hac	7960	*7960*	*7960*	*75.54*	*2.93*	*18.75*	*1984*	*1984*	*1984*	*textfile*	1994	text	Guide to hacking	«idea of hacking for knowledge, and leaving no footprints and destroying nothing in the process»	The Basics of Hacking: Introduction, by The Knights of Shadow The Knights of Shadow produced a collection of Hacking How-Tos that instructed users how to get around a variety of mainframes, including Digital and Data General Machines. While the information in these texts might not be as relevant, this introduction stands apart for its preaching the idea of hacking for knowledge, and leaving no footprints and destroying nothing in the process. Their definition of "Hacker" is rather interesting, as well. A clever set of files.
hack11a.txt	692945	*679083*	*692945*	*56.38*	*3.28*	*16.75*	*1960-1995*	*1960*	*1995*	*textfile*	1986	text	Example of Project Gutenberg E-Text Transcript	~400 Page Book!	The Project Gutenberg E-Text of Bruce Sterling's Hacker Crackdown In 1993, Science Fiction author Bruce Sterling aimed his considerable writing talents to dissecting and understanding all the forces at work between Hackers, Police, and the people they've had an effect on. He does a very admirable job. While no book (so far) has captured the 1980's computer hacker experience perfectly, this book makes you come away with a feeling that the major issues were touched on and that no-one (on either side) got the short shaft. (This is a major accomplishment in itself.) Sterling is an excellent writer, and while compared to other works in this directory this textfile is a bit on the mammoth side, it's worth it. This file also highlights the work of the great Project Gutenberg, which for 20 years has endeavored to transcribe as many classic works to electronic texts as their staff of volunteers will let them. There's an entire directory of these important and breathtakingly huge projects on textfiles.
hack7.txt	3988	*3893*	*3988*	*53.46*	*2.13*	*14.52*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	What it is to be a hacker…	**Take a closer look and compare to declaration of cyberspace!**	The Conscience of a Hacker, by The Mentor (January 8, 1986) The Mentor's angry scream against the authorities he saw as trying to crush his spark and the spark of people like him. For some reason, this file became the flashpoint that a number of books (including Bruce Sterling's) used as an example of the oppression of the intelligent and the motivations behind the fine art of hacking. Whether it stands up to this sort of light or not, it's a clear statement from someone who feels a lot of pain; and that's what communication is all about
highdoc.ana	1408	*1382*	*1408*	*61.64*	*2.68*	*17.29*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Guide to get high from Gatorade		How to get Really Soaring High on Gatorade, by Max Madd Some of the most entertaining textfiles are the ones where the writer is obviously hot to write one, but has no actual information to report on. Instead, they pull up a in-the-rough concept, like getting high off a fruit drink, and create a short file around it. In this case, the key is to get high off the thirst-quenching Gatorade, by merely drinking it as fast as possible. Nearly a third of the entire file is a legal disclaimer, a typical gesture that is very likely meaningless in the long run.

feh-1	54625	*54622*	*54625*	*44.82*	*2.12*	*7.99*	*1995-1995*	*1995*	*1995*	*textfile*	1992	text, messages	Hacker Magazin		Fuckin' Eleet Haxor Issue #1, July 1st 1995 As the World Wide Web really started to take hold in the middle of the 1990s, textfiles became a rarer and rarer entity, usually leaning instead to HTML pages and graphics to get the information (and the point) across. In the case of FEH, indicative of the textfiles of the time, it is sometimes very hard to tell where the parody and where the seriousness lay within the issue. While a lot of it seems to be a thought-out send-up of the badly-spelling hacking community, some serious and researched information is included as well. This magazine went on to several additional issues, each of them a little more serious than the last. For better or worse, this is how things came to be in the culture.
gems.txt	75172	*73969*	*75172*	*65.2*	*2.65*	*13.26*	*1960-1992*	*1960*	*1992*	*textfile*	1989	text	Article about Gems	very nice formatted!	GEMS: The Untold Story, by the Video Vindicator (February 1, 1992) The Video Vindicator came late into the game (early 1990's) but produced some of the most wonderful files from that time period. This file caught my attention because he chose a subject that would normally be of very little interest to the BBS crowd (Gemology) and takes it to a completely new level by turning it into both a wonderful history lesson about Gems (I learned a lot in this file) and twisting it into yet another way to scam the planet for some extra bucks. Breathtaking in his audacity, and completely slick in his delivery. One to watch.
groupass.phk	1809	*1763*	*1809*	*55.39*	*2.93*	*11.65*	*nd.*	*nd.*	*nd.*	*textfile*	1989	messages	Funny description of an issue in telephone system (dog as grounding)		
hack_ths.txt	148109	*148109*	*148109*	*60.42*	*1.21*	*21.25*	*1971-1989*	*1971*	*1989*	*textfile*	1984	text	Thesis: The social organization of the computer underground by GORDON R. MEYER	**Take a closer look**	The Social Organization of the Computer Underground, The Thesis of Gordon Meyer Mr. (Probably Dr.) Meyer's Sociology paper on the social structure in Hacking, Phreaking, and Pirating groups is interestingly dedicated to George Hayduke (writer of the "Get Even" series of revenge books) and Barry Freed (The pseudonym taken by Abbie Hoffman when he was a fugitive from the FBI). This indicates not a little leaning on the side of the groups he's analyzing. He goes over impressions that the media have about hackers, what being a hacker, phreak or pirate consists of as he sees it, and then tries to draw conclusions of what this all means. Written in 1989, this file takes on the subculture with a non-hysterical point of view that makes it very easy to read. Worth the time.
hack1.hac	7960	*7960*	*7960*	*75.54*	*2.93*	*18.75*	*1984*	*1984*	*1984*	*textfile*	1994	text	Guide to hacking	«idea of hacking for knowledge, and leaving no footprints and destroying nothing in the process»	The Basics of Hacking: Introduction, by The Knights of Shadow The Knights of Shadow produced a collection of Hacking How-Tos that instructed users how to get around a variety of mainframes, including Digital and Data General Machines. While the information in these texts might not be as relevant, this introduction stands apart for its preaching the idea of hacking for knowledge, and leaving no footprints and destroying nothing in the process. Their definition of "Hacker" is rather interesting, as well. A clever set of files.
hack11a.txt	692945	*679083*	*692945*	*56.38*	*3.28*	*16.75*	*1960-1995*	*1960*	*1995*	*textfile*	1986	text	Example of Project Gutenberg E-Text Transcript	~400 Page Book!	The Project Gutenberg E-Text of Bruce Sterling's Hacker Crackdown In 1993, Science Fiction author Bruce Sterling aimed his considerable writing talents to dissecting and understanding all the forces at work between Hackers, Police, and the people they've had an effect on. He does a very admirable job. While no book (so far) has captured the 1980's computer hacker experience perfectly, this book makes you come away with a feeling that the major issues were touched on and that no-one (on either side) got the short shaft. (This is a major accomplishment in itself.) Sterling is an excellent writer, and while compared to other works in this directory this textfile is a bit on the mammoth side, it's worth it. This file also highlights the work of the great Project Gutenberg, which for 20 years has endeavored to transcribe as many classic works to electronic texts as their staff of volunteers will let them. There's an entire directory of these important and breathtakingly huge projects on textfiles.
hack7.txt	3988	*3893*	*3988*	*53.46*	*2.13*	*14.52*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	What it is to be a hacker…	**Take a closer look and compare to declaration of cyberspace!**	The Conscience of a Hacker, by The Mentor (January 8, 1986) The Mentor's angry scream against the authorities he saw as trying to crush his spark and the spark of people like him. For some reason, this file became the flashpoint that a number of books (including Bruce Sterling's) used as an example of the oppression of the intelligent and the motivations behind the fine art of hacking. Whether it stands up to this sort of light or not, it's a clear statement from someone who feels a lot of pain; and that's what communication is all about
highdoc.ana	1408	*1382*	*1408*	*61.64*	*2.68*	*17.29*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Guide to get high from Gatorade		How to get Really Soaring High on Gatorade, by Max Madd Some of the most entertaining textfiles are the ones where the writer is obviously hot to write one, but has no actual information to report on. Instead, they pull up a in-the-rough concept, like getting high off a fruit drink, and create a short file around it. In this case, the key is to get high off the thirst-quenching Gatorade, by merely drinking it as fast as possible. Nearly a third of the entire file is a legal disclaimer, a typical gesture that is very likely meaningless in the long run.

killsant.hum	3497	*3410*	*3497*	*57.29*	*1.6*	*5.27*	*1994*	*1994*	*1994*	*textfile*	nd	text	Example of Fiction vs. Fact		How to Kill Santa Claus Dead! by the Outland The Outland (of the Neon Knights) ran the first board I was ever a Sysop of, Milliways. You would never know it if you met him, but when the Outland sat down to write files for Metal Communications and the Neon Knights, he would just spew forth some of the most violent, anarchistic, nihilist text to come out of any file-writing group. Besides Santa Claus, other targets of his files included the Easter Bunny, mailboxes, the local neighborhood, and in one case, I recall, the entire planet. He was certainly in character with the other members of the Neon Knights, where violence was the main driving force in the writing, but knowing him personally gave me insight that for him, it was actually just all fiction, another neat thing to do.
killshco.ana	16886	*16541*	*16886*	*62.29*	*2.81*	*11.34*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Anarchy style: smash your school!		The SchoolStoppers' Textbook The Yippies, or "Youth International Party" were a political group from the late 60's and early seventies that were really the grandfathers of a lot of the computer "underground" that flourishes today. They staged protests, wrote interesting books and articles, and published the Youth International Party Line (YIPL) which later became TAP, a predecessor of the currently famous 2600 magazine. Among their famous members were Abbie Hoffman and Jerry Rubin. This reprinted article is essentially a checklist for causing utter anarchy at your local school, so as to completely disrupt the learning process. Angry but witty, this was where a lot of later "anarchy" files took their style from, knowing it or not.
krckwczt.app	137510	*134363*	*137510*	*48.67*	*1.59*	*4.92*	*1985*	*1985*	*1985*	*textfile*	nd	text	Guide to Cracking		Kracowicz' Kracking Korner: The Basics of Kracking Parts 1-9 The world of Apple II Copy Protection was a hot battlefield throughout the Apple II's production life, with commercial developers spending thousands on new copy protection schemes and crackers rising out of the woodwork to "crack" these programs, make them copyable, and distribute them. With Krakowicz, you had a Software Company's worst nightmare: A literate, intelligent crackist who made a supreme effort to teach others. What is most striking about this series of files is not just the amount of detail and research that Kracowicz put into his writing to make it understandable to others, but his unique hardware-based solutions to the software that was being piped through his Apple II. By creating boards, switches, and hot-wiring his chips, he could exert incredible control over the programs he was attempting to crack. The companies didn't have a chance. Kracowicz stands alone.
lay-girl.txt	16308	*15955*	*16308*	*56.32*	*1.69*	*10.9*	*nd.*	*nd.*	*nd.*	*textfile*	1985	text	Guide to to get laid		The Complete Guide of Laying a Girl v1.1, from John Smith Probably the most amusing textfiles I occasionally stumbled across were those attempting to teach you the birds and the bees, or at least how to get laid. Usually in the form of "how-to" guides, these textfiles were usually completely out of left field, totally lacking in any accuracy or truly helpful information, and more likely than not someone's complete fantasy from watching too many teen exploitation flicks. In the case of this particular specimen, Mr. Smith seems to have as weak a grasp on the English language as he does on the particulars of intercourse or romance. Such stunning phrases as "Stack you hand gently under her trousers and move your hand more deeply evert time" guarantee that you're going to take this file with an oven-sized grain of salt. Sadly, this file is among the best of the bunch -- many of the others indicated rape or kidnapping as appropriate means to seduction. A fountain of ignorance.
leeches!.hum	11312	*11055*	*11312*	*53.46*	*1.93*	*7.88*	*1979-1994*	*1979*	*1994*	*textfile*	1987	text	What are leeches? (fun)		The Society of "Leeches" in the Telecommunications World, by Mister I/O "Mister I/O" was the first name of The Outland, who later went on to join the Neon Knights and Metal Communications. In this file, he skewers the world of "Leeches", users who connect to systems and take all the files without donating any of their own. This particular kind of file (ridiculing other groups within the subculture) were plentiful by this time, but I think his stands out for that completely bizzare chart of the lineage of Leeches. Additionally, he even throws in some mathematical equations to determine your "leechiness". This file was written before his files took on a much more violent (but still witty) turn, as mentioned previously.

lod-1	213571	*213571*	*213571*	*56.44*	*1.58*	*5.9*	*1976-1994*	*1976*	*1994*	*textfile*	1984	text, table, scheme	Article with Schemes and Tables (separated by «|»)		The Legion of Doom/Hacking Technical Journal Volume 1, by the LOD I can remember when the Legion of Doom was just one of a group of punk kids hanging around on some of the same BBSes that I frequented. But someone or several someones within the group threw a ton of energy into the LOD, and they quickly rose to the forefront of Hacker/Phreaking groups of the time. The advantage of years of hindsight and a number of books have brought the group much more fame and regard historically, but even the most cynical or skeptical observer had to admit; this group produced. A prime example are these Legion of Doom/Hacker Technical journals, an indirect response to Phrack and other hacker magazines. The LOD/H Tech Journal was heavy, meaty stuff for the time (1987) with schematics, statistics, and even attributions to the files that had come before it. The series is worth reading as a prime example of the "highbrow" hackers, who put on the airs of having the knowledge to share, while raising the ire of those who didn't.
lozers.hum	24778	*24214*	*24778*	*56.44*	*2.13*	*9*	*1979-1985*	*1979*	*1985*	*textfile*	1985	text	Biggest well known Lozers		The Official 1984 Lozerlist, by The Atom (March 3, 1985) The Atom's bile-spewing, gossip-heavy "Loser List" comes in as one of the best examples of an all-around "Rag File" that I've seen, where the writer takes everyone on and attempts to completely slam them into the ground on all sides. Personal info, analysis of the groups they belong to, rumors and innuendo are all up for grabs as he decimates a lot of the "big names" in the 914/212/718 BBS world (as well as a few from the midwest). It appears a lot of his hatred comes from the Richard Sandza articles (also in the top 100), but more than that, he has it in for nearly anyone who has become in some way "famous" or "legendary" in the Phreak and Piracy world. Quite a piece of work. I never found a 1985 list or anything since.
ludeinfo.hum	9298	*9110*	*9298*	*59.09*	*2.25*	*8.3*	*1984*	*1984*	*1984*	*textfile*	nd	text	Luding: sport?		Basic, Intermediate and Advanced Luding, by Sinbad Sailor While calling an interesting BBS called the 1985 BBS (the last four digits were 1985, it came up on January 1st of 1985 and went down December 31st, 1985), I stumbled upon this small social phenomenon called "Luding". The board was reeking with it; there was a Luding sub-board filled with messages and there were a good number of instructional "Luding" files, not to mention some Luding poetry and fiction. Naturally, this really threw me for a loop, and while the actual idea of "Luding" is somewhat tame, these set of files stand out to me as excellent examples of how just writing about a small little fad in your hometown could blow the entire thing out of proportion to the point that you thought you were really missing out on something big.
mathimp.txt	4177	*4097*	*4177*	*52.86*	*2.65*	*10.83*	*nd.*	*nd.*	*nd.*	*textfile*	1986	text	Story: Polly Nomial (Math Geek humor)		Impure Mathematics: The Story of Polly Nomial What we have here is probably the all-time marathon-running Usenet humor file. A collection of mathematic inside jokes used to tell a running set of sexual innuendo, this story of the hapless Polly Nomial and Curly Pi was the kind of file you'd trip over time and time again in different file directories. There are a good amount of sequels, but you can't beat the original. Math Geek humor; what beats that?
miami.hum	8107	*7940*	*8107*	*61.2*	*2.96*	*18.49*	*nd.*	*nd.*	*nd.*	*textfile*	1992	text	Story: Captain Goodnight		Go Bare, by Captain Goodnight (August 25, 1986) Captain Goodnight's fictional short story is worth reading for the somewhat accurate feeling it gives of living a life primarily through a computer. There's some bad spelling and weird images mixed into the paragraphs, but on the whole, you'll come away from it either remembering memories from your own childhood or feel like you're peering into someone's life. A real gas of a story.
mindvox	66115	*66115*	*66115*	*67.59*	*2.14*	*32.91*	*1978-1993*	*1978*	*1993*	*textfile*	1985	text	About MindVox (early ISP)		Voices in My Head: The Mindvox Overture by Patrick Kroupa Mr. Kroupa's announcement of the creation of his new ISP caught a lot of people off guard. The tack from which he made his announcement was to harken back to the 1980's, and all the unique cultural forces at work then, that expressed themselves on the modem. In many ways, he saw a lot of what I've seen in terms of a need to preserve or maintain that time in some fashion. He additionally saw a need to make those times come back by creating Mindvox, his ISP by way of his company Phantom Technologies. I know that mindvox was wildly successful for a time, but the full story of its downfall escapes me. Meanwhile, this text still survives and shows an amazing breadth of insight into the BBS world of the preceding decade. This document was reprinted everywhere, so it definitely hit a nerve. I just wish we had a little more in our hands to show for it.

modemlif.hac	11074	*10913*	*11074*	*71.21*	*2.87*	*16.86*	*1983*	*1983*	*1983*	*textfile*	1984	text	Frustration by downed BBSes → a Call for better?	**Take a closer look and compare to declaration of cyberspace!**	The Modem Life: Is it Worth it? By Bryan Nomad (May 26, 1985) [?] Bryan, frustrated by downed BBSes, busy signals, abuse and hatred in message bases, and the financial drain of being s SysOp, takes a different tack. He writes this heartfelt message to the BBS world at large to ask people to remember that they're all people, they're all part of a community, and it's not about how much abuse you can spew into the air, but about getting to know one another and maybe make some friends. How effective is this file at getting into a cynical heart? Who knows. One can always hope it did some good. Thank you, Bryan.
nighhack.omn	9225	*9225*	*9225*	*68.21*	*2.43*	*15.3*	*1984*	*1984*	*1984*	*textfile*	nd	text	Story: NIGHT OF THE HACKERS		The Night of the Hackers, by Richard Sandza, transcribed by The Reflex There are two notable angles to this file. The more simpler one is that it was transcribed by The Reflex of Omnipotent Incorporated, a regular on The Works BBS and an all-around prolific and intelligent writer. He chose as an important addition to his body of work this article from Newsweek, published in 1984. The second angle is this article, "Night of the Hackers", which brought right to the forefront a lot of those mysterious terms that hung in the air as private knowledge and secrets for only those "in the know" in the BBS world. Sherwood Forest, Blottoland, Plovernet, tele-trials all got (unwanted) national exposure in this article, and things just weren't the same afterwards. The aftermath of Richard Sandza's life is documented in "Revenge of the Hackers", also in this collection. Studied with the span of years from it, the article is solid, interesting, and pretty accurate.
noise.ana	3679	*3581*	*3679*	*40.98*	*1.24*	*3.44*	*nd.*	*nd.*	*nd.*	*textfile*	1981	text	How to build variable-pitched frequency generator		Variable-Pitched Frequency Generator, or How to Annoy Your Teachers, by Captain Quieg The Captain provides you with an accurate (I had one built), inexpensive, and mostly harmless prank electronic toy that would emit an extremely high-pitched noise, such that many people could get headaches from prolonged exposure. Not exactly a gentle diversion, but unforgettable. While this particular diagram worked, it was a relatively rare thing to come away from an electronics document and have everything function as claimed. Educational.
paging_g.ame	4275	*4170*	*4275*	*49.57*	*2.11*	*14.16*	*1981*	*1981*	*1981*	*textfile*	nd	text	Article about Paging		The Paging Game, by Jeff Berryman Trying to explain this one to anyone who doesn't know the concept of "memory paging" is a little bit of a losing battle. Essentially, a machine with memory that pages will throw out unused parts of programs or files onto a disk drive, ready to bring it back if you really use it. If you think MY explanation's a little choppy, wait'll you real THIS file. A golden piece of work from 1981.
peat.hum	532	*521*	*532*	*56.67*	*2.13*	*8.33*	*nd.*	*nd.*	*nd.*	*textfile*	1987	text	short joke		The Peat Moss Incident The best size-reaction ratio of the textfiles collection. Surreal.
pezrambl.oct	20975	*20660*	*20975*	*68.49*	*1.83*	*9.33*	*1987-1987*	*1987*	*1987*	*textfile*	1985	text	Essay about Textfiles	**Take a closer look**	Mr. Pez's Rambling About Textfiles and Leeching, by Mr. Pez (March 13, 1987) Mr. Pez was another Works regular, always there with a quick wit and a really snooty attitude. Besides being on The Works, he also graced several other 914 boards and The Dark Side of the Moon (408) with his acidic attitude. In this file, he decided to just turn on his word processor and talk about whatever for as long as he could. To our advantage, the subject he chooses is Textfiles, which gives us insight into the type of world a teenager lived through in 1987 BBS terms. Most amusingly to me, he recounts the time I sat there copying all my textfile collection onto his hard drive trying to convince me to have his guitar and some cash. (The attempt was unsuccessful, and Donna married some other guy.) A personal trip down memory lane, but relevant to get some insight into where my editorial bent is as well.
phrack.01.phk	29195	*29195*	*29195*	*50.72*	*1.05*	*6.55*	*1985*	*1985*	*1985*	*textfile*	1989	text	Magazine: Phrack	voice of the underground	Phrack Magazine Volume One, Issue One, edited by Taran King (November 17, 1985) The Phreaking/Hacking magazine that changed everything. While other electronic magazines existed before Phrack, none took the voice of the underground and presented itself as such a dominating, matter-of-fact entity as the Phracks have. Through the years, Phrack has always been dependable as a solidly-written, interest-gathering, packed-with-talent compilation of hot topics going throughout the Hacker Underground as most people have come to understand it. Electronic zines as a general force were usually created so that individual writers' work wouldn't get lost in the wash of sites; by hooking up with a dozen other articles, relatively monstrous 50k-100k files could stand out from the endless grouping of 2k and 15k files that others were putting out. It worked.. very few people don't remember Phrack in some fashion if they were involved in BBSes in the late 80's. This brings us to the other example issue…

phrack.29.phk	235777	*235777*	*235777*	*63.08*	*2.15*	*11.78*	*1960-1994*	*1960*	*1994*	*textfile*	1984	text	Magazine: Phrack		Phrack Magazine 4th Anniversary Issue, Volume Three, Issue 29 (November 17, 1989) By the fourth year of publication, Phrack is an institution. The issue opens with a profile of Emmanuel Goldstein, the enigmatic and steadfast editor of 2600 Magazine (which has gone on to become a major institution itself) and progressing into deeply technological discussions involving money transfer and Internet protocols. By this time the Phrack World News, an overview of the social and legal scene around the culture had become a staple of the issues. Unstoppable.
pokelist.app	19769	*19300*	*19769*	*46.5*	*1.11*	*2.93*	*1977-1998*	*1977*	*1998*	*textfile*	1988	text	technical information about apple II		The Wizard's Call, Peek and Poke list for the Apple][(May 1984) Part of the immense charm of the Apple II series of computers was how they would encourage their users to learn everything they could about the system, to reprogram, modify, hardwire, and otherwise mess with all aspects of the machines. What this meant was that people were getting a knowledge of the Apples that could far outstrip almost all the other personal computers of the time. Evidence of the depth of this knowledge shows in files like this one, where a good portion of the total memory locations have been mapped and all sorts of neat features make themselves known. By the end run of the Apple II's main life (late 80's) this machine could accomplish a breathtaking amount of tasks. Geeky, but a lot of fun.
purity.txt	58845	*57834*	*58845*	*58.1*	*1.95*	*10.86*	*1982-1988*	*1982*	*1988*	*textfile*	nd	text	«self test»	cares for omnisexuality	The Unisex, Omnisexual Purity Test v4.00 (April 23, 1988) One of the interesting things that arose out of the ARPAnet and Internet was the way that an unbelievable amount of energy could be focused on a single project, causing it to turn into a Wonder Of The World in a very short period of time. In the case of this file, the goal became to determine a person's "purity" by creating a list of questionable non-innocent acts that they could perform in life, and whatever percentage they had not yet done, was their purity. Of course, after dozens of entries into this document, it's become this complete other world, with every degrading, exciting, bizzare thing that someone could do with someone else (or themselves, or a group of people, or food) is listed. There are actually divergent, unrelated versions of this idea up on this site, but I chose this file because it lists out a great pedigree that goes back to 1982, and it's particularly well-written. Head-swimmingly sick.
real.pgmrs	23955	*23955*	*23955*	*58.44*	*2.83*	*18.16*	*1969-1982*	*1969*	*1982*	*textfile*	1987	text	Geeky humor about programming		Real Programmers Don't Use Pascal One of the most interesting fads to hit the online world were the "Real" files. Based loosely on the pop culture book "Real Men Don't Eat Quiche", these files presented a framework where others could just list their idea of what a "real" hacker or golfer or restaurant or whatever. It's an addictive way to describe things, and this explains the dozens and dozens of "real" files that pervaded BBSes throughout the decade. In the case of the "Real Programmers" file, the writing style of the author is particularly well-crafted (although I can't really judge the accuracy of his assertions) and it therefore has a very large distribution. Geeky humor.
realpez.oct	16384	*16079*	*16384*	*52.23*	*0.89*	*13.64*	*1987*	*1987*	*1987*	*textfile*	nd	text	Real PEZ idk		Real PEZ Devotees, by Mr. Pez The Works BBS's own Mr. Pez makes his own contribution to the "Real" files canon with this file about the followers of his BBS, "Pez Devotees". In the case of this file, the combination of his mention of all the different things he personally liked (including clothes, bands, sports and writing style) combined with his edging into nearly all aspects of a person's life to provide guidelines to be one of his devotees, makes this one of my favorite files. It should be noted that this file is a derivative of the original, the "Real Pirate's Guide", below.
realpira.hum	6529	*6324*	*6529*	*48.56*	*2.36*	*8.13*	*1984*	*1984*	*1984*	*textfile*	1984	text	real pirate guide		The Real Pirate's Guide by Rabid Rasta [?] Seizing the opportunity to make a humor file based on the now-popular "Real Men Don't Eat Quiche" idea, Rabid Rasta made what is generally agreed upon to be the first of the BBS world's "Real" files, files which explain what the difference between "Real" and "Fake" was. In the case of his file, he puts down what makes a "Real" pirate, including assertions about computer and modem speed, writing style, and spelling. With the exception of "The Real Programmer's Guide" (which shows up a little earlier than this file, although only on ARPAnet and not on the BBS/AE world), this file seems to have been the one that started it all. The observations he makes are both humorous, and insightful into where the world was in 1984 if you were living your life through a modem.

revhack.omn	6266	*6266*	*6266*	*69.28*	*2.07*	*12.83*	*1984*	*1984*	*1984*	*textfile*	nd	text	Following up on Story: NIGHT OF THE HACKERS → REVENGE OF THE HACKERS	Hackers hit Sandza badly	Revenge of the Hackers, by Richard Sandza, Transcribed by The Reflex The writer of "Night of the Hackers" for Newsweek revealed in this followup article that his life became quite a living hell for him after the original article was published. His credit cards were hacked, phone calls came 24 hours a day, and, to a lesser relevance, he went on "tele-trial". Tele-trial appears to mean that he had a message base fill up with messages about him. Either way, his article shows the kind of full-on attack that the hacker community was capable of pulling off at the time, completely unorganized and out for blood. The Reflex, as always, does an impeccable transcription job.
revolt.dj	2358	*2262*	*2358*	*36.02*	*1.49*	*9.34*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	peom		Dick and Jane Instigate a Popular Revolution, by The Deth Vegetable and Iskra Deth Vegetable and Iskra were both co-sysops of The Works way back when, before they both joined the Cult of the Dead Cow and moved on to greener pastures. (Actually, Iskra then came back and became the SysOp of the most recent incarnations of the Works, so it's all kind of circular in a way.) In any event, they got their heads together in the early 1990's and wrote this file, a parody of the "Dick and Jane" books that my generation lightly heard of. Short, sweet, and funny. Oh, and political.
safeinfo.fun	2176	*2131*	*2176*	*54.89*	*2.44*	*8.03*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Tec-flexing		The Safehouse Blueprints, from the Safehouse BBS The Safehouse BBS was one of the mid-80's "Super" BBSes that were pushing not only that they were the places to be, but that the technology and pure computing power at their disposal made them inherently irresistible. Now, looking back over the years with a more solid technical knowledge, some of the claims in this BBS information file are questionable (It's a modular program? What does that matter!) But you can see an example of how technology presented its own inherent sexiness to the BBS user.
sexsatan.hum	8979	*8822*	*8979*	*66.43*	*1.92*	*8.41*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Story: erotic, heavy metal / geek combination		Sex with Satan, by Psychoe Ah, where do you begin with "Sex with Satan". Psychoe's tale of murder, sexual mayhem, the Lord of Darkness and overactive babysitters never fails to floor me with its perfect blend of comic rant and erotic literature. You come away from reading it completely dazed and unsure what you just read. While you begin to read the file thinking you're to be subjected to another set of poorly spelled pseudo-erotic scribblings, you quickly realize as the character dies that something is quite amiss. After a short time you can't keep track of who is who and why the poor guy keeps getting lit on fire. Another great example of the Heavy Metal/Geek combination that showed up in a small number of AE lines in the 80's.
spock.art	5904	*5821*	*5904*	*69.98*	*0.1*	*0.29*	*nd.*	*nd.*	*nd.*	*textfile*	nd	ASCII Art	Mr. Spock		Digitized Picture of Star Trek's Mr. Spock One of the nicer ASCII Art Files out there, this appears to have been some sort of program output that looked at a graphics file of Mr. Spock and then mapped it to different ASCII characters based on how "dark" those letters and numbers and characters were. If you print it out so that it's black letters on a white background, it looks pretty darn impressive. Of course, actually printing it out and putting it on a wall would immediately tap you as a geek, so be careful, and have a shredder at hand.
stupidki.hac	4942	*4843*	*4942*	*62.42*	*3.04*	*15.98*	*nd.*	*nd.*	*nd.*	*textfile*	1987	text	call for action on destructiveness of the «kids»		The Destructiveness of the "Kids", from an Anonymous Source A writer reaches out to the community to lament the loss of the days of the Open System and the unprotected BBS, and gives some (occasionally gleeful) descriptions of the steps he's been forced to take to protect himself. The exact tone of this missive is a little hard to track: in some places he's nostalgic and out of sorts about the way the world has changed, and in others he's the first to the forefront of implementing clampdowns of security and creating traps for any hapless hackers attacking his BBS. Definitely makes you think, if not too hard.
suicide	10438	*10215*	*10438*	*63.63*	*1.65*	*9.65*	*1987*	*1987*	*1987*	*textfile*	nd	text	representation of oddiness of neon knight group		Getting Others to Commit Suicide, by The Blade of the Neon Knights The Blade was one of the leaders of the Neon Knights, who themselves were an elite portion of the Metal Communications team, purveyor of textfiles throughout the BBS world, but mostly through AE lines. With their unique blend of Heavy Metal, Satanism, and Geek Chic, the group put out some of the odder files out there. In the case of this file, The Blade tells you how to drive others to self-destruction, so as to make it easier to get into college or a well-paying job. A graphic collection of sick suggestions; just the soft of thing you could expect from this group. Are people really afraid of this sort of thing?
taoprogram.pro	27266	*26293*	*27266*	*53.99*	*1.46*	*6.49*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Book about Tao redone for programming		The Tao of Programming, by Geoffrey James Mr. James' gentle and intelligent retelling of the idea of the Tao redone for programming, hardware, and software. This soft of file, meant to be a light parody of another, more famous work, ends up standing up pretty well on its own. There's one for documentation as well, and there's other more pop-culture books expounding on the Tao you can purchase these days. A good read if you believe in it or you don't.

taping.hum	5117	*4964*	*5117*	*43.74*	*2.12*	*8.04*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	tease, slander another member		The Taping I, by Underwarez A particularly virulent example of a "Rag File", a file whose entire purpose was to tease, slander, or otherwise ridicule another member of the same (usually pirating or phreaking) subculture. In this particular rag, a hapless user named Jeff Spicolli is subject to implications of incest, lack of technical knowledge, bestiality, and other similar traits, ending with his voice telephone number. This type of file would start showing up on local AE lines or BBSes and while most of these rants would disappear about a week or two after the Sysops took them down, a few still hung around. The most involved set of rag files to my knowledge would be the Matt Ackeret chronicles from Anarchy Incorporated.
tencoms.pro	1029	*997*	*1029*	*45.95*	*2.06*	*11.4*	*nd.*	*nd.*	*nd.*	*textfile*	1982	text	From BBS to unenjoyable piece		The Ten Commandments of RBBS I was never a big fan of RBBS software, mostly because a lot of the sysops I ran into had the attitude put forward in this file. Reading over the commandments, we find that the SysOp is God, that you must not use handles or speak of things not involving computers, that profanity is unwelcome, and that a full three commandments dictate what kind of advertising you may post. This file is a great example of how attitude could turn a BBS from a place of fun to a stuffed-shirt, bland, unenjoyable piece of cardboard. Then again, posting this kind of file told people what kind of administrator you were right off the bat, enabling easy and quick escape. Run.
top10.news	4272	*4272*	*4272*	*61.49*	*3.08*	*25.87*	*nd.*	*nd.*	*nd.*	*textfile*	1988	text	example of ignorance on a massive scale (of the police)		The Top 10 Media Errors about the Steven Jackson Games Raid This 1992 document from Steve Jackson games responds to some of the most blatant factual errors about the infamous "Raid" on that role-playing game company. In the course of an investigation, the US Secret Service raided Steve Jackson games and confiscated all the materials of a role-playing game called GURPS CYBERPUNK, which was played with dice and cards, and didn't even involve a computer. SJ Games were unable to have their game back for 7 months, during which time it was described as some sort of "hacking manual" that the country had to be protected from. Naturally, when all was said and done, the game was harmless and nothing what the Secret Service claimed it was. A true insight into ignorance on a massive scale.
tr823.txt	106655	*104707*	*106655*	*57.96*	*3.2*	*17.12*	*1972-1988*	*1972*	*1988*	*textfile*	nd	text	Article about The Internet Worm Program	**Take a closer look**	The Internet Worm Program: An Analysis, by Eugene H. Spafford The Internet Worm changed a lot of minds about how interconnected and insecure the Internet was at that time. Using a combination of weaknesses and back doors in common programs, the Worm wended its way throughout the then-small Net and succeeded in crippling it. This document, written during the Aftermath, presents a well-thought-out analysis of all the methodology used by the worm, as well as a general oversight of the state of the Internet of the time. Long, but worth it.
upc.txt	6726	*6726*	*6726*	*55.14*	*1.45*	*7.89*	*nd.*	*nd.*	*nd.*	*textfile*	1986	text	Cracking the UPC (Universal Product Code – Strichcode)		Cracking the Universal Product Code, by Count Nibble Count Nibble makes a second appearance in the top 100 files with his steady, thoughtful explanation of exactly how those silly black lines on every product in the supermarket work. What possible use this knowledge could have in one's day to day existence is pretty darn irrelevant; the point was, it was THERE, we WANTED TO KNOW, and Nibble FOUND IT OUT. That's the spirit of learning. Read and find out too.
urine.box	4141	*4059*	*4141*	*61.88*	*1.34*	*7.43*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	Awarnenes-test because fully fictional		Urine Box Plans, by Wolfgang von Albatross (March 2, 1986) By the time this file appeared on AE lines across the country, "Box" files had proliferated to the point that it was hard to tell who came up with with idea first. As might be expected, parodies began appearing, including the "Blotto Box" (which would supposedly destroy an entire telephone central switching office) and this specimen, which purports to cause the headset on the other end of the line to injure or kill the user. Naturally, this file is complete fiction, but constructed with enough of a straight face to make the unsuspecting collector think they have some sort of accurate textfile. An excellent awareness test.
utopia.hum	3193	*3063*	*3193*	*34.51*	*2.02*	*6.06*	*1984*	*1984*	*1984*	*textfile*	1989	text	utopia logon screen – fun – easy customizable		Login Screen for the Utopia BBS: "May I take your Order Please?" Taking advantage of the easily modifiable source code of their BBS, the Sysops of Utopia BBS constantly (and I do mean constantly) retooled all menus, messages, login and logoff screens, and even system functions. One result of this was that you always had a surprise the next time you connected to the BBS, and you always felt like you were part of a party. Note that the system actually made me type in my password, only to ignore me and continue its merry blather for a few more paragraphs.
vaxen.jok	15163	*14871*	*15163*	*56.83*	*1.72*	*18.24*	*1987*	*1987*	*1987*	*textfile*	1985	text	some fun stuff idk		VAXen, My Children, Just Don't Belong Some Places, by Mike O'Brien Mr. O'Brien's ode to the misunderstood, mistreated VAX has achieved quite a large distribution; I keep finding it buried everywhere, in joke files and computer information sites and just generally all over. Through his sad tale of VAX abuse, Mike keeps you interested to the very end. How much of it is true is left up to the reader, of course, but somehow, it just rings enough with me to consider it real. Interesting.

violence.txt	11481	*11265*	*11481*	*59.72*	*2.14*	*12.55*	*1985-1985*	*1985*	*1985*	*textfile*	1984	text	«Count Nibble» one of the most literate. A pioneer.		Fun! With Random Senseless Violence, by Count Nibble (August 2, 1985) Count Nibble was one of the most literate of all the Apple II-era textfile writers, pre-dating similarly styled groups like Metal Communications and the Cult of the Dead Cow by years with his musical quotes, proper spelling and formatting, and choosing all sorts of esoteric subjects to write about. In the case of this file, he lists out his suggestions for causing mayhem and destruction on a boring summer's night. Through browsing his web presence, an interesting fact about Nibble made itself known to me: He was college-age when this and other files were written, bringing a maturity to both his spelling and general perspective, if not his general outlook on personal property. A pioneer.
warbitch.txt	9225	*9070*	*9225*	*67.03*	*2.31*	*7.47*	*1984*	*1984*	*1984*	*textfile*	nd	text	Guide to verbal warrior		The Code of the Verbal Warrior, or Barney's Bitch Manual Barney Badass, co-sysop of The Glue Ball and heady influence on a number of Chicago BBSes, rants forth with a set of instructions on how to conduct a proper "bitch war". In the vernacular of the Internet, this has come to be known as a "flame war", but the same idea holds: Long after the intended subject has dropped out of debate, two (or more) sides begin a verbal assault on each other that fills the message base with dozens of personal attacks, insinuations, libel, and slander. After a while, everyone not personally involved in the bitchwar is driven away, posting messages on other sub-boards, which causes them to be insulted for not posting on the right sub-board, possibly leading to another bitchwar. Barney Badass, himself, was a true character and the instigator of some of my finest on-line memories.
warning.hum	2537	*2492*	*2537*	*71.97*	*1.23*	*4.46*	*nd.*	*nd.*	*nd.*	*textfile*	nd	text	little funny poem-like warning		Warning: This Machine Breaks Down During Periods of Critical Need! Another light chuckle from the textfiles of the 1970s, this file warns you that the machine is more likely to break down the more you need it. The attempt to parody industrial labelling as well as the attention to clever turns of phrase marks a lot of the "big iron" humor in textfiles from professionals and college students of this time, showing they were looking for a little lightness in their otherwise stressful and highly-taxing occupations.
warnings.ufo	1031	*1006*	*1031*	*54.5*	*2.58*	*16.78*	*nd.*	*nd.*	*nd.*	*textfile*	1985	text	little funny poem-like warning		THE FOLLOWING ARE COMMON SENSE WARNINGS WHEN DEALING WITH A UFO Yes, in the event that you do come across a classic lights-flashing UFO, just follow these simple hints and you probably won't be abducted, garrotted, or experimented on. A must for overseas travellers!
watchem.phk	3635	*3547*	*3635*	*52.41*	*2.28*	*7.45*	*1984*	*1984*	*1984*	*textfile*	1987	text	Waring about how to use BBS(?)		Watching the Watcher Watching You, by Sir Knight (1985) Besides the very memorable title, Sir Knight's file stands out for his call to mistrust others in the subculture as being potential enemies and agents, and to think twice before revealing information about yourself. Most noticeably, his automatic dismissal of anyone asking "how do I do this?" on public boards shows the beginnings of a trend that continues to this day. (Note also, that his portrait of the most trustable phreakers/hackers are anyone jewish, middle-upper-class, and under 19.)
whytext.oct	2223	*2159*	*2223*	*56.62*	*2.12*	*9.15*	*nd.*	*nd.*	*nd.*	*textfile*	1971	text	About why Textfiles are better		Why I Prefer Textfiles, by Jason Scott (February 27, 1987) I make a cameo appearance! This file came out of my mind one summer in my 16th year while I was home sick with the flu. As my BBS was dedicated to textfiles, I thought it would be fun to write one of my own about why textfiles were better. This file was parodied by a few of my friends at the time, but I think, looking back, my heart was in the right place. (It's still there.)
yipl.phk	15312	*15312*	*15312*	*58.62*	*1.91*	*6.41*	*1966-1971*	*1966*	*1971*	*textfile*		text	Partie issue(?)		The Youth International Party Line #1, Transcribed by BIOC Agent 003 BIOC, besides being particularly good at writing textfiles of his own composition, was dedicated to preserving knowledge of the foundation that he and others were building on top of. In the case of Phone Phreaks and later Phreak Magazines, this foundation was YIPL, the house organ of the Yippies, who were a revolutionary youth group of the 1960's. YIPL provided one of the first radical magazines dedicated to learning more about technology. While the first issues (under the influence of Abbie Hoffman and others) merely called upon its readers to use this knowledge as a crowbar to smash the state, later issues (when the magazine renamed itself to the Technological Assistance Party, or TAP) brought forth a love of learning and understanding how technology affected all our lives, and a need to know who was pulling the strings. BIOC does his best to transcribe this issue as close to what it looked like when hastily-scrawled copies were sent out to a few dozen people in June of 1971. Good show.

8.2 Anhang 2: Sourcecode Top100

Mini-Project in the ABC of Computational Text Analysis

Author: Josias Bruderer, Universität Luzern

Date: 26. May 2021

Preparations [R1]

The following lines of code is used for preparing our environment.

In [1]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

# load the necessary libraries

import sys
from pathlib import Path
import textacy
import spacy
import re
import itertools
import numpy as np

import scattertext as st
import pandas as pd

from pathlib import Path
from plotnine import *
import textacy.vsm

project_path = Path.cwd().parent

# prepare to load project specific libraries
module_path = str(project_path / "py/")
if module_path not in sys.path:
    sys.path.append(module_path)

# import data_wrangler module
from modules import data_wrangler
from modules import helpers
```

Data Wrangling [R2]

In this section the required data is downloaded and preprocessed (f.E. unzipped). The module data_wrangler will be used for this.

In [2]:
```
data_url = "http://archives.textfiles.com/100.zip"
data_name = "" #use 100 to only analize the top100 files
data_dir = str(project_path / "data/")
tmp_dir = str(project_path / ".tmp/")

dw = data_wrangler.DataWrangler(tmp_dir, data_dir)

dw.process_zip(data_url, data_name)
```

Analisis [R3]

Preliminary Clarifications [R3.1]

In [3]:
```
# preparation for manual analisis [R3.1.1]

# read textfiles
dataset = dw.get_texts(data_dir + "/" + data_name)

# write metadata to csv file
```

```
f = open("top100_generated.csv", "w+")
for item in dataset:
    f.write(item[1]["name"]+","+
            str(item[1]["length"])+","+
            str(item[1]["length_raw"])+","+
            str(item[1]["avgcolumnsize"])+","+
            str(item[1]["charratioA"])+","+
            str(item[1]["charratioB"])+","+
            str(item[1]["year"])+","+
            str(item[1]["eyear"])+","+
            str(item[1]["lyear"])+","+
            str(item[1]["type"])+","+
            "\r\n")
f.close()
```

Create Textcorpora [R3.2]

In [4]:
```
# read textfiles
dataset = dw.get_texts(data_dir + "/" + data_name)

en = textacy.load_spacy_lang("en_core_web_sm")

# create corpus from processed documents
corpus = textacy.Corpus(en, data=dataset)
```

Create Subcorpora [R3.3]

In [5]:
```
# function to filter by metadata: filter invalid filenames
def filter_by_name(doc):
    return doc._.meta.get("name") != "index.html" and doc._.meta.get("name") != ".ztr-directory"

# function to filter by metadata: filter files that contain less than 50% A-z characters
def filter_by_charratio(doc):
    return doc._.meta.get("charratioA") > 0.5

# create new corpus after applying filter function
subcorpustemp = textacy.corpus.Corpus(en, data=corpus.get(filter_by_name))
subcorpus = textacy.corpus.Corpus(en, data=subcorpustemp.get(filter_by_charratio))
```

Wordcount [R3.4]

In [6]:
```
# get lowercased and filtered corpus vocabulary (R3.3.1)
vocab = subcorpus.word_counts(as_strings=True, normalize= 'lower', filter_stops = True, filter_punct

# sort vocabulary by descending frequency
vocab_sorted = sorted(vocab.items(), key=lambda x: x[1], reverse=True)

# write to file, one word and its frequency per line
fname = './analysis/vocab_frq.txt'
with open(fname, 'w') as f:
    for word, frq in vocab_sorted:
        line = f"{word}\t{frq}\n"
        f.write(line)

vocab_sorted[:25]
```

Out[6]:
```
[('computer', 1267),
 ('system', 932),
 ('like', 822),
 ('people', 813),
 ('time', 715),
 ('new', 646),
 ('use', 576),
 ('phone', 555),
 ('program', 554),
 ('number', 537),
 ('t', 501),
 ('real', 497),
 ('software', 492),
 ('service', 489),
 ('know', 468),
 ('bbs', 452),
 ('information', 445),
 ('file', 430),
 ('hackers', 410),
 ('board', 408),
 ('x', 406),
 ('code', 401),
```

```
('way', 397),
('boards', 391),
('systems', 390)]
```

Export to CSV [R3.5]

In [7]:
```
# merge metadata and actual content for each document in the corpus
# ugly, verbose syntax to merge two dictionaries
data = [{**doc._.meta, **{'text': doc.text}} for doc in subcorpus]

# export corpus as csv
f_csv = './analysis/subcorpus.csv'
textacy.io.csv.write_csv(data, f_csv, fieldnames=data[0].keys())
```

Load CSV file [R3.6]

In [8]:
```
# read dataset from csv file
f_csv = './analysis/subcorpus.csv'
df = pd.read_csv(f_csv)

df_sub = df[(df['text'].str.len() > 10)]

# make new column containing all relevant metadata (showing in plot later on)
df_sub['descripton'] = df_sub[['name', 'year', 'charratioA', 'avgcolumnsize']].astype(str).agg(', '.

# sneak peek of dataset
df_sub.head()
```

Out[8]:

	name	length_raw	length	avgcolumnsize	charratioA	charratioB	year	eyear	lyear	type	
0	declarationbarlow1996.txt	5089	5089	252.70	4.08	114.66	1996	1996	1996	declaration	A Declar c Independ Cybersp
1	peat.hum	532	521	56.67	2.13	8.33	nd.	nd.	nd.	textfile	When we to dra out c pea
2	killsant.hum	3497	3410	57.29	1.60	5.27	1994	1994	1994	textfile	
3	pezrambl.oct	20975	20660	68.49	1.83	9.33	1987-1987	1987	1987	textfile	——— — ——— — —
4	warning.hum	2537	2492	71.97	1.23	4.46	nd.	nd.	nd.	textfile	

Scattertext Plot [R3.7]

In [9]:
```
# import with a short name
import scattertext as st
import pandas as pd

# import all specific/all objects from a module
from pathlib import Path
from plotnine import *
import textacy.vsm

censor_tags = set(['CARD']) # tags to ignore in corpus, e.g. numbers

# stop words to ignore in corpus
en_stopwords = spacy.lang.en.stop_words.STOP_WORDS # default stop words
custom_stopwords = set(['[', ']', '%'])
en_stopwords = en_stopwords.union(custom_stopwords) # extend with custom stop words

# create corpus from dataframe
# lowercased terms, no stopwords, no numbers
# use lemmas for English only, German quality is too bad
corpus_speeches = st.CorpusFromPandas(df_sub, # dataset
                              category_col='type', # index differences by ...
                              text_col='text',
                              nlp=en, # German model
                              feats_from_spacy_doc=st.FeatsFromSpacyDoc(tag_types_to_censor=censor_ta
                              ).build().get_stoplisted_unigram_corpus(en_stopwords)

# produce visualization (interactive html)
```

```
html = st.produce_scattertext_explorer(corpus_speeches,
            category='declaration', # set attribute to divide corpus into two parts
            category_name='declaration',
            not_category_name='textfiles',
            metadata=df_sub['descripton'],
            width_in_pixels=1000,
            minimum_term_frequency=5, # drop terms occurring less than 5 times
            save_svg_button=True,
)

# write visualization to html file
fname = "../public/viz_declaration_textfiles.html"
open(fname, 'wb').write(html.encode('utf-8'))
```

Out[9]: 4750122

Docs per Year [R3.8]

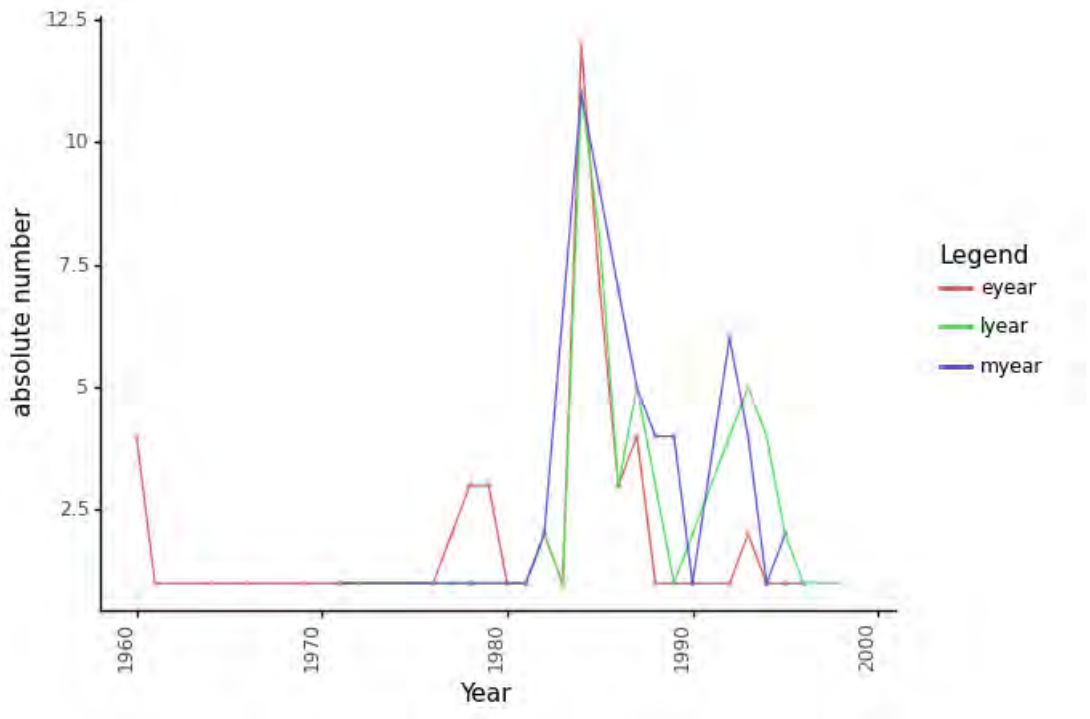
In [10]:
```
dtmp = df_sub.groupby('eyear').agg({'text': "count" }).reset_index().rename(columns={'text':'count'}
dtmp = dtmp.rename(columns={"eyear": "year"})
dtmp.insert(2,"type","eyear")
docs_per_year = dtmp

dtmp = df_sub.groupby('lyear').agg({'text': "count" }).reset_index().rename(columns={'text':'count'}
dtmp = dtmp.rename(columns={"lyear": "year"})
dtmp.insert(2,"type","lyear")
docs_per_year = docs_per_year.append(dtmp, ignore_index=True)

dtmp = pd.read_csv('top100_years.txt', delimiter = ",").groupby('myear').agg({'text': "count" }).res
dtmp = dtmp.rename(columns={"myear": "year"})
dtmp.insert(2,"type","myear")
docs_per_year = docs_per_year.append(dtmp, ignore_index=True)

docs_per_year = docs_per_year[docs_per_year["year"] != "nd."]
docs_per_year['year'] = pd.to_numeric(docs_per_year['year'])

(ggplot(docs_per_year, aes('year', 'count', color='type', group='type'))
 + geom_point(alpha=0.5, stroke = 0)
 + geom_line()
 + theme_classic()
 + labs(x = "Year",
        y = "absolute number",
        color = "Legend")
 + theme(axis_text_x = element_text(angle = 90, hjust = 1))
 + scale_x_continuous(limits = (1960,1999))
)
```



Out[10]: <ggplot: (8737743464162)>

Correlations for docs per year

In [11]:
```
# sorry for ugly...
dummy = pd.DataFrame([{"year":1960, "count": 0},
                      {"year":1961, "count": 0},
                      {"year":1962, "count": 0},
                      {"year":1963, "count": 0},
                      {"year":1964, "count": 0},
                      {"year":1965, "count": 0},
                      {"year":1966, "count": 0},
                      {"year":1967, "count": 0},
                      {"year":1968, "count": 0},
                      {"year":1969, "count": 0},
                      {"year":1970, "count": 0},
                      {"year":1971, "count": 0},
                      {"year":1972, "count": 0},
                      {"year":1973, "count": 0},
                      {"year":1974, "count": 0},
                      {"year":1975, "count": 0},
                      {"year":1976, "count": 0},
                      {"year":1977, "count": 0},
                      {"year":1978, "count": 0},
                      {"year":1979, "count": 0},
                      {"year":1980, "count": 0},
                      {"year":1981, "count": 0},
                      {"year":1982, "count": 0},
                      {"year":1983, "count": 0},
                      {"year":1984, "count": 0},
                      {"year":1985, "count": 0},
                      {"year":1986, "count": 0},
                      {"year":1987, "count": 0},
                      {"year":1988, "count": 0},
                      {"year":1989, "count": 0},
                      {"year":1990, "count": 0},
                      {"year":1991, "count": 0},
                      {"year":1992, "count": 0},
                      {"year":1993, "count": 0},
                      {"year":1994, "count": 0},
                      {"year":1995, "count": 0},
                      {"year":1996, "count": 0},
                      {"year":1997, "count": 0},
                      {"year":1998, "count": 0},
                      {"year":1999, "count": 0}])

m = dummy.append(docs_per_year[docs_per_year['type'] == "myear"][["year","count"]]).groupby('year').
e = dummy.append(docs_per_year[docs_per_year['type'] == "eyear"][["year","count"]]).groupby('year').
l = dummy.append(docs_per_year[docs_per_year['type'] == "lyear"][["year","count"]]).groupby('year').
```

In [12]:
```
print("r_{myear mit eyear} = ",np.corrcoef(m["count"], e["count"])[0, 1])
print("r_{myear mit lyear} = ",np.corrcoef(m["count"], l["count"])[0, 1])
```

```
r_{myear mit eyear} =  0.7666121890473637
r_{myear mit lyear} =  0.91481657982525
```

Entities [R3.9]

In [13]:
```
entities = []

for doc in subcorpus.docs:
    for ent in textacy.extract.entities(doc):
        try:
            entities += [{"text": ent.text, "label": ent.label_, "explain":spacy.explain(ent.label_)
        except:
            print("Problem with:", doc._.meta["name"])

# export corpus as csv
f_csv = './analysis/entities.csv'
textacy.io.csv.write_csv(entities, f_csv, fieldnames=entities[0].keys())
```

```
Problem with: top10.news
Problem with: hack11a.txt
Problem with: hack11a.txt
Problem with: hack11a.txt
Problem with: hack11a.txt
Problem with: hack11a.txt
Problem with: famous.bug
Problem with: purity.txt
Problem with: utopia.hum
Problem with: dec.wars
Problem with: dec.wars
Problem with: krckwczt.app
```

In [14]:
```
#df_sub.groupby('eyear').agg({'text': "count" }).reset_index().rename(columns={'text':'count'})
```

```
df_entities = pd.DataFrame(entities,columns=['text','label','explain'])
df_entities_count = df_entities.groupby('text').agg({'label': "count"}).rename(columns={'label':'cou


# write to file, one word and its frequency per line
fname = './analysis/entities_frq.csv'
with open(fname, 'w') as f:
    for i, d in df_entities_count.iterrows():
        line = d["text"]+","+str(d["count"])+"\n"
        f.write(line)

df_entities_count[:25]
```

Out[14]:

	text	count
0	one	426
1	first	359
2	two	302
3	1	285
4	2	260
5	3	223
6	10	172
7	24	170
8	IBM	170
9	4	165
10	5	149
11	Secret Service	132
12	American	124
13	second	113
14	three	105
15	Bell	104
16	20	100
17	One	89
18	6	86
19	8	84
20	Kapor	83
21	today	82
22	12	81
23	Neidorf	76
24	America	74

8.3 Anhang 3: textfiles.com Analyse

Preparation

```
# install stuff that we need later
if (!require("DT")) install.packages('DT')
if (!require("ggplot2")) install.packages('ggplot2')
if (!require("tidyverse")) install.packages('tidyverse')
if (!require("hrbrthemes")) install.packages('hrbrthemes')
if (!require("dplyr")) install.packages('dplyr')

# Load stuff we need later
library(readr)
library(DT)
library(ggplot2)
library(tidyverse)
library(hrbrthemes)
library(dplyr)
library(scales)

# and set the working directory
setwd("~/projects/bbs-for-independence/03_workspace")
```

Import Data

```
# Read dataset summary from csv
dataset <- read_csv("./models/dataset.csv", show_col_types = FALSE)
dataset$charratioDelta = dataset$charratioB - dataset$charratioA
```

Prepare Data

```
# Check the average of length, length_raw, avgcolumnsize, charratioA and charratioB
df = aggregate(x = dataset[, c(4,5,6,7,8,13)],
               by = list(dataset$category),
               FUN = function(x) list(
                 mean = round(mean(suppressWarnings(as.numeric(as.character(x))), na.rm=TRUE), digits = 2),
                 n = length(x)))
df <- do.call(data.frame, df) # bind columns which contain matrices back into the data frame
df <- as.data.frame(lapply(df, unlist)) # convert lists back to vectors

f_selection <- dataset %>% filter(!category %in% c("fidonet-on-the-internet", "tap", "floppies",
                                                   "exhibits", "artifacts", "piracy", "art",
                                                   "magazines", "digest"))
f_magazins <- dataset %>% filter(category == "magazines")
f_digest <- dataset %>% filter(category == "digest")

fun_charratioB_selection <- Vectorize( function(x) { nrow(f_selection %>% filter(charratioB > x)) } )
fun_charratioB_magazins <- Vectorize( function(x) { nrow(f_magazins %>% filter(charratioB > x)) } )
fun_charratioB_digest <- Vectorize( function(x) { nrow(f_digest %>% filter(charratioB > x)) } )

data_fun <- data.frame(x = seq(0,1,0.01),
                       n = c(fun_charratioB_selection(seq(0,1,0.01)),
                                 fun_charratioB_magazins(seq(0,1,0.01)),
                                 fun_charratioB_digest(seq(0,1,0.01))),
                       categories = rep(c("selection", "magazins", "digest"), each = 101))
```

Summarize Data


```
cat("Anzahl Dateien: ", nrow(dataset))
```

```
## Anzahl Dateien:  105470
```

```
cat("Anzahl Kategorien: ", nrow(df))
```

```
## Anzahl Kategorien:  48
```

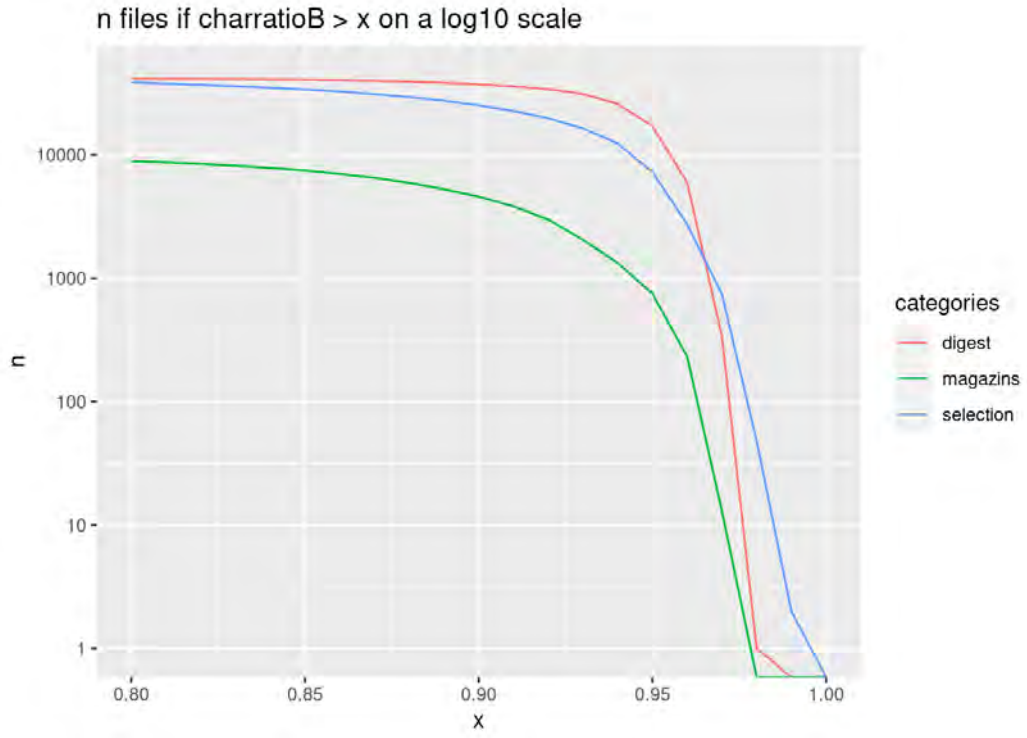
```
cat("Anzahl Dateien bei Filterung von tap, art, floppies, piracy, exhibits, magazines, digest:", nrow(f_selecti
on))
```

```
## Anzahl Dateien bei Filterung von tap, art, floppies, piracy, exhibits, magazines, digest: 44833
```

```
cat("Anzahl Dateien bei zusätzlicher Filterung von charratioB >0.95:", fun_charratioB_selection(0.95))
```

```
## Anzahl Dateien bei zusätzlicher Filterung von charratioB >0.95: 7242
```

```
# draw curve for fun_charratioB
ggplot(data_fun, aes(x, n, col = categories)) +
  geom_line() +
  xlim(0.8, 1) +
  scale_y_continuous(trans = log10_trans()) +
  ggtitle("n files if charratioB > x on a log10 scale")
```




```
datatable(df %>%
  arrange(desc(charratioB.mean)) %>%
  select(Group.1, length.n, length.mean, length_raw.mean, avgcolumnsize.mean,
         charratioA.mean, charratioB.mean, charratioDelta.mean),
  options = list(
    pageLength = 50,
    initComplete = JS("function(settings, json) {
                      $(this.api().table().header()).css({'font-size' : '12px'});
                      $('table.dataTable thead th').css({'padding' : '10px 18px 10px 0px'});
                      }")
    )
  )
```

	Group.1	length.n	length.mean	length_raw.mean	avgcolumnsize.mean	charratioA.mean	charratioB.mean	charratioDelta.mean
1	declaration	1	5089	5089	252.7	0.8	0.99	0.19
2	history	52	18230.58	18278.6	67.65	0.75	0.96	0.21
3	sex	5226	25250.56	25696.85	66.28	0.73	0.95	0.22
4	digest	41958	34516.91	34540.33	58	0.74	0.94	0.2
5	etext	1167	305073.34	307896.73	54.88	0.72	0.94	0.22
6	law	533	29662.52	30135.91	61.38	0.72	0.94	0.22
7	politics	2194	27928.15	28268.88	61.01	0.69	0.94	0.25
8	stories	474	27103.32	27541.75	63.82	0.71	0.94	0.23
9	news	184	12877.33	13061.82	78.99	0.73	0.93	0.2
10	occult	2400	28027.67	28461.17	61.07	0.7	0.92	0.22
11	sf	633	38383.36	39023.21	55.81	0.7	0.92	0.22
12	survival	105	16288.03	16550.34	62.77	0.68	0.92	0.25
13	drugs	1047	17824.46	17985.24	58.56	0.69	0.91	0.22
14	fun	430	25410.16	25883.55	58.23	0.67	0.91	0.24
15	uploads	560	8668.19	8697.51	95.72	0.68	0.91	0.23
16	adventure	550	11923.53	12104.18	66.53	0.66	0.9	0.24
17	apple	1553	17911	18129.07	71.52	0.63	0.9	0.26
18	conspiracy	1111	21273.07	21478.9	58.46	0.69	0.9	0.21
19	food	213	9774.92	9995.34	48.26	0.66	0.9	0.24
20	humor	2061	13307.51	13585.09	52.6	0.68	0.9	0.22
21	rpg	300	41099.01	41704.22	56.14	0.67	0.9	0.24
22	anarchy	2509	14516.98	14765.6	62.18	0.63	0.89	0.25
23	media	164	39436.89	40091.92	51.39	0.65	0.89	0.24
24	ufo	2928	12439.76	12679.46	60.37	0.67	0.89	0.22
25	100	100	28158.48	28532.05	56.12	0.65	0.88	0.24
26	games	980	18863.05	19138.54	60.01	0.63	0.88	0.25
27	internet	849	44187.19	44850.73	54.01	0.66	0.88	0.22
28	groups	1269	13385.23	13555.14	125.96	0.6	0.87	0.27
29	hacking	1107	28180.3	28623.02	62.17	0.62	0.87	0.25
30	magazines	10630	27312.95	27550.01	121.91	0.63	0.87	0.23
31	music	608	25020.44	25376.26	51.51	0.6	0.86	0.26
32	reports	713	11362.56	11559.32	67.8	0.62	0.86	0.24
33	virus	545	14449.13	14753.63	57.81	0.58	0.86	0.28
34	programming	601	37755.52	38504	55.11	0.58	0.85	0.28

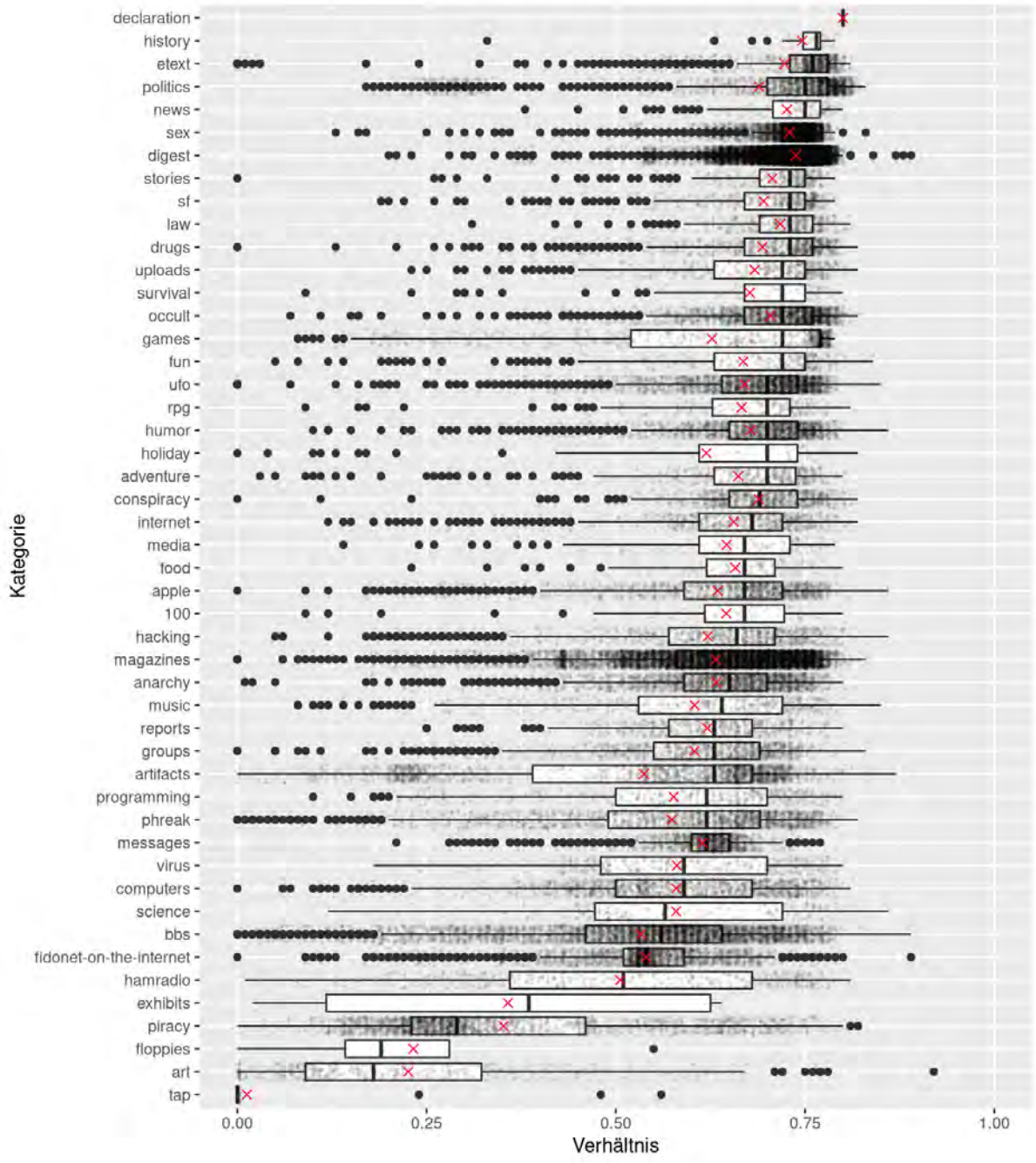
	Group.1	length.n	length.mean	length_raw.mean	avgcolumnsize.mean	charratioA.mean	charratioB.mean	charratioDelta.mean
35	computers	1699	22515.02	22964.09	55.69	0.58	0.84	0.26
36	holiday	73	5081.51	5191.74	140.96	0.62	0.84	0.23
37	phreak	2195	15278.46	15573.29	59.61	0.57	0.84	0.26
38	messages	1543	41495.49	42262.34	50.03	0.61	0.83	0.21
39	bbs	5242	23949.07	24374.74	62.4	0.53	0.81	0.28
40	hamradio	636	14112.25	14384.66	55	0.51	0.81	0.31
41	science	278	18247	18479.9	57.43	0.58	0.81	0.23
42	fidonet-on-the-internet	2498	180826.98	182547.84	49.15	0.54	0.78	0.24
43	art	656	24714.97	24756.06	86.08	0.23	0.74	0.51
44	piracy	2539	9177.07	9326.82	85.12	0.35	0.73	0.38
45	artifacts	2246	37027.04	37305.74	69.53	0.54	0.7	0.16
46	exhibits	4	97722.25	97724.75	119.16	0.36	0.54	0.18
47	floppies	4	8654	8654	29.81	0.23	0.31	0.08
48	tap	102	853.44	877	1.04	0.01	0.02	0.01

Showing 1 to 48 of 48 entries

Previous 1 Next

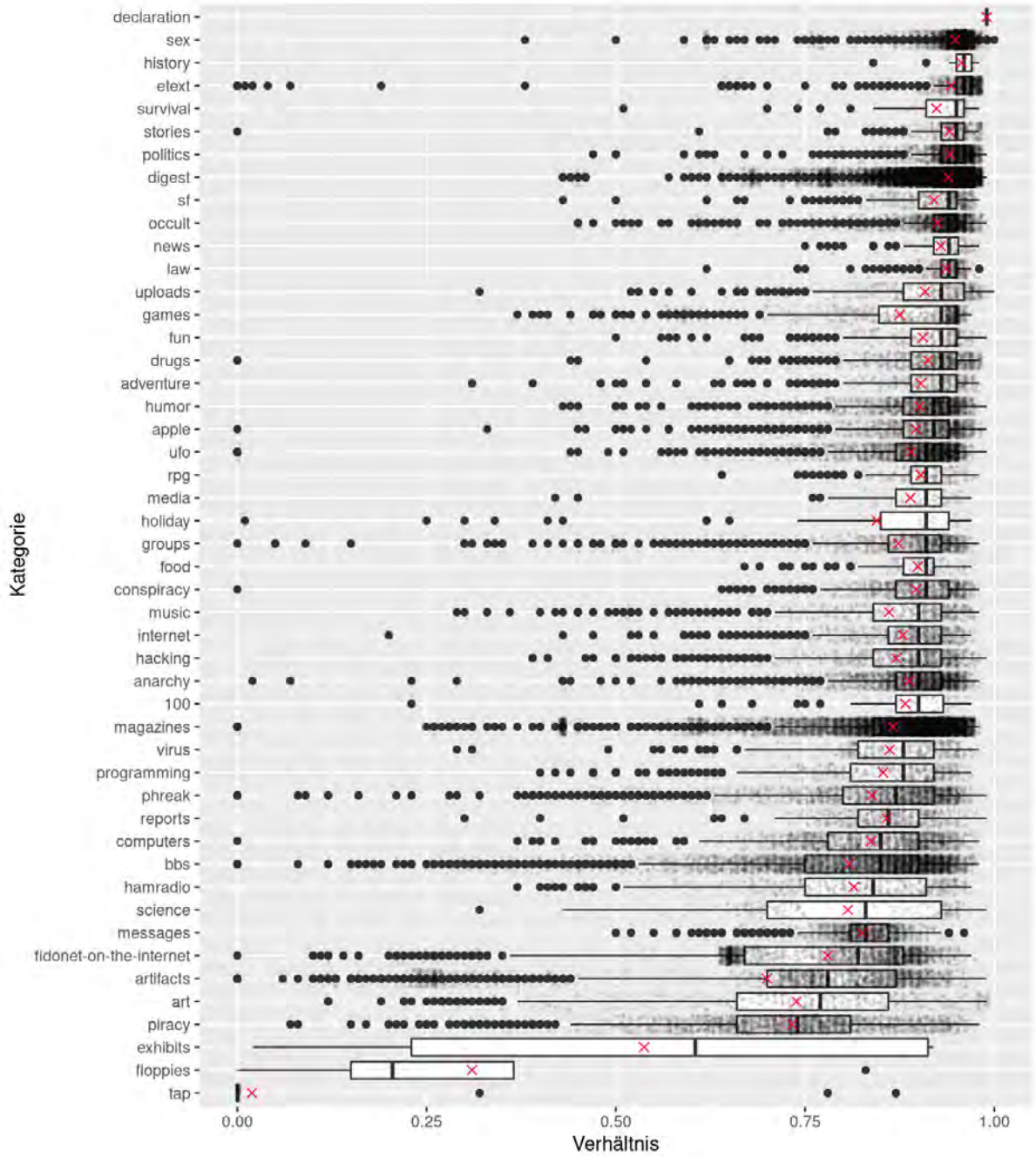
Verhältnis Text (exkl. Satz- und Leerzeichen) zu Dateilänge

```
dataset %>%
  ggplot( aes(x=reorder(category, charratioA, FUN = median),
              y=charratioA, group=category)) +
    geom_boxplot() +
    theme(
      legend.position="none",
      plot.title = element_text(size=11)
    ) +
    geom_jitter(color="black", size=0.4, alpha=0.05) +
    stat_summary(fun.y=mean, geom="point", shape=4, size=2, color="red", fill="red") +
    coord_flip() +
    ylim(0, 1) +
    xlab("Kategorie") +
    ylab("Verhältnis")
```



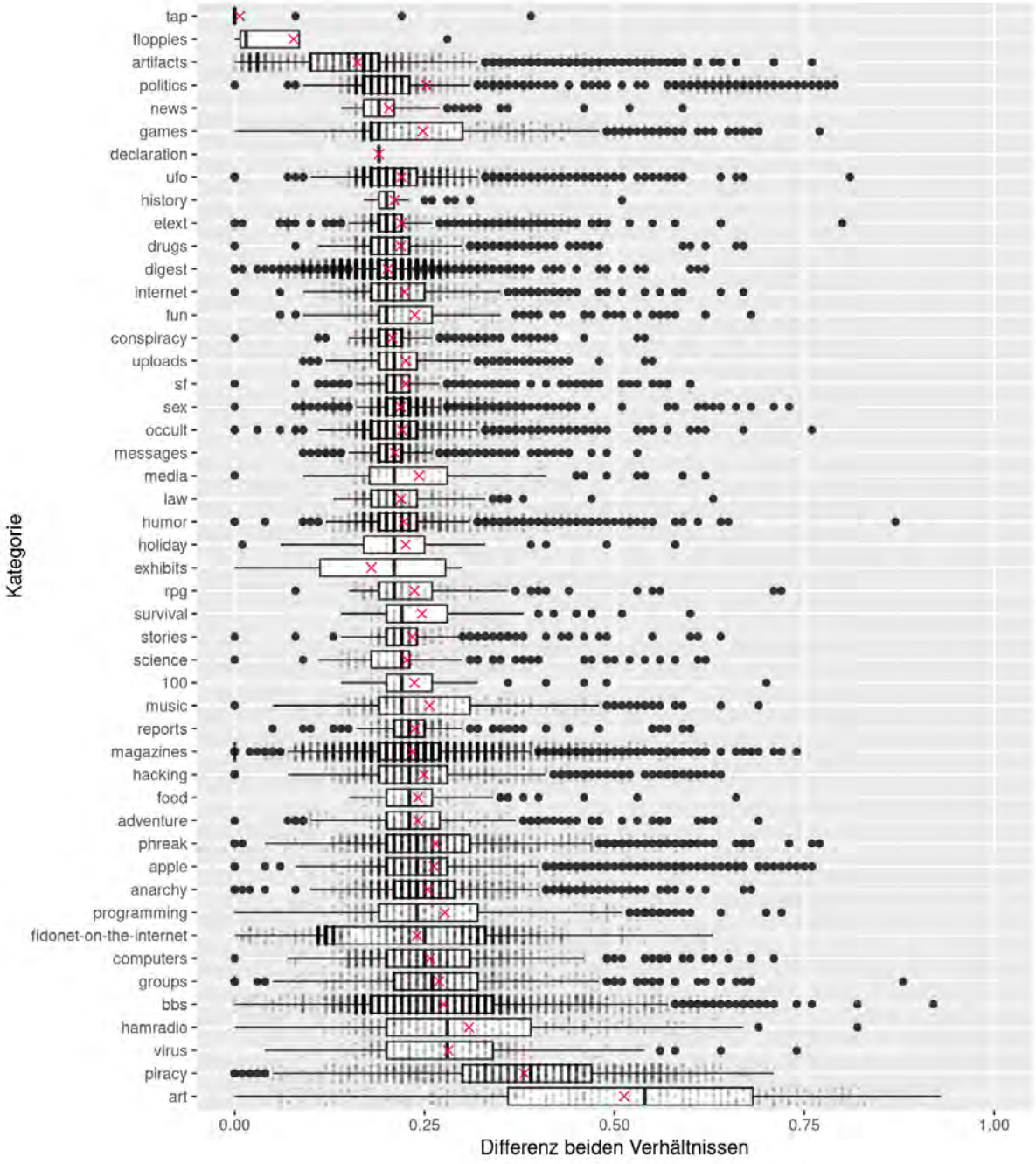
Verhältnis Text (inkl. Satz- und Leerzeichen) zu Dateilänge

```
# create plot: charratioB
dataset %>%
  ggplot( aes(x=reorder(category, charratioB, FUN = median),
              y=charratioB, group=category)) +
    geom_boxplot() +
    theme(
      legend.position="none",
      plot.title = element_text(size=11)
    ) +
    geom_jitter(color="black", size=0.4, alpha=0.05) +
    stat_summary(fun.y=mean, geom="point", shape=4, size=2, color="red", fill="red") +
    coord_flip() +
    ylim(0, 1) +
    xlab("Kategorie") +
    ylab("Verhältnis")
```



Differenz beiden Verhältnissen (inkl. minus exkl. Satz- und Leerzeichen zu Dateilänge)

```
dataset %>%
  ggplot( aes(x=reorder(category, charratioA-charratioB, FUN = median),
              y=charratioB-charratioA, group=category)) +
    geom_boxplot() +
    theme(
      legend.position="none",
      plot.title = element_text(size=11)
    ) +
    geom_jitter(color="black", size=0.4, alpha=0.05) +
    stat_summary(fun.y=mean, geom="point", shape=4, size=2, color="red", fill="red") +
    coord_flip() +
    ylim(0, 1) +
    xlab("Kategorie") +
    ylab("Differenz beiden Verhältnissen")
```



Apply filtering and extend filtering

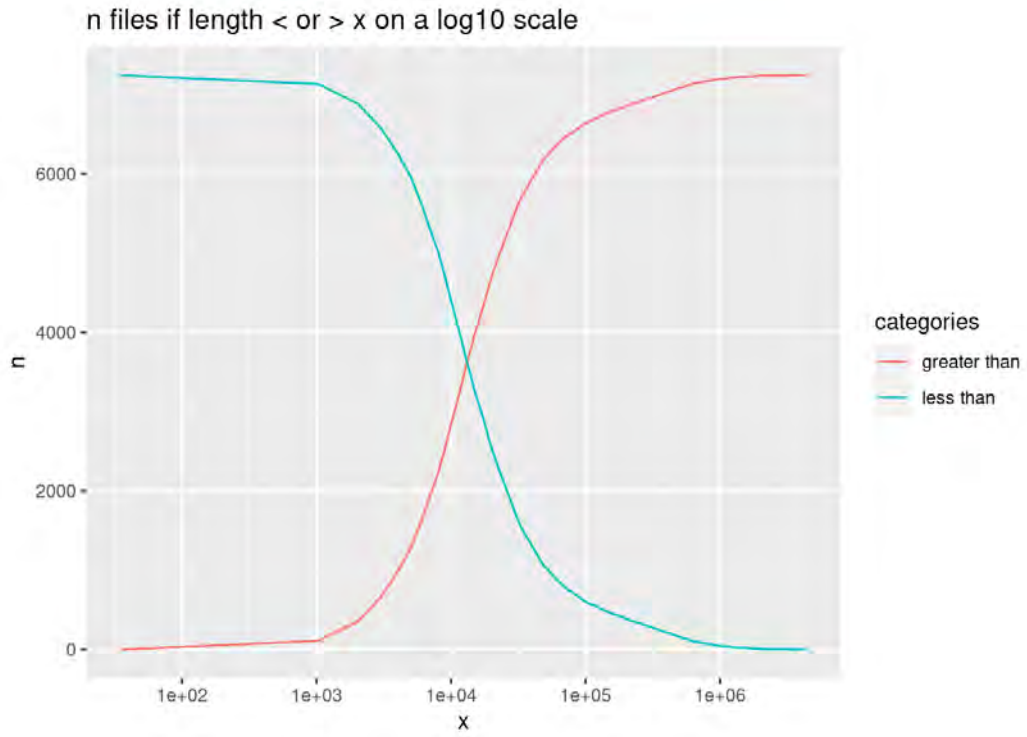
```
data_names_exclude <- c("fidonet-on-the-internet","tap","floppies","exhibits",
                        "artifacts","piracy", "art", "magazines", "digest")

dataset_filtered = dataset %>%
  filter(!category %in% data_names_exclude) %>%
  filter(charratioB > 0.95)

fun_length_selection_lt <- Vectorize( function(x) { nrow(dataset_filtered %>% filter(length < x)) } )
fun_length_selection_gt <- Vectorize( function(x) { nrow(dataset_filtered %>% filter(length > x)) } )

length_fun_seq = seq(min(dataset_filtered$length),max(dataset_filtered$length),1000)
length_fun <- data.frame(x = length_fun_seq,
                         n = c(fun_length_selection_lt(length_fun_seq),
                               fun_length_selection_gt(length_fun_seq)),
                         categories = rep(c("greater than", "less than"), each = length(length_fun_seq)))

ggplot(length_fun, aes(x, n, col = categories)) +
  geom_line() +
  scale_x_continuous(trans = log10_trans()) +
  ggtitle("n files if length < or > x on a log10 scale")
```




```
dataset_filtered_2 = dataset %>%
  filter(!category %in% data_names_exclude) %>%
  filter(charratioB > 0.95) %>%
  filter(length > 300) %>%
  filter(length < 30000)

cat("Anzahl Dateien mit gefilterter Länge: ", nrow(dataset_filtered_2))
```

```
## Anzahl Dateien mit gefilterter Länge:  5510
```

```
cat("Gesammt Länge der Dateien mit gefilterter Länge: ", sum(dataset_filtered_2$length),
    "≙",round(sum(dataset_filtered_2$length)/1000000),"MB ")
```

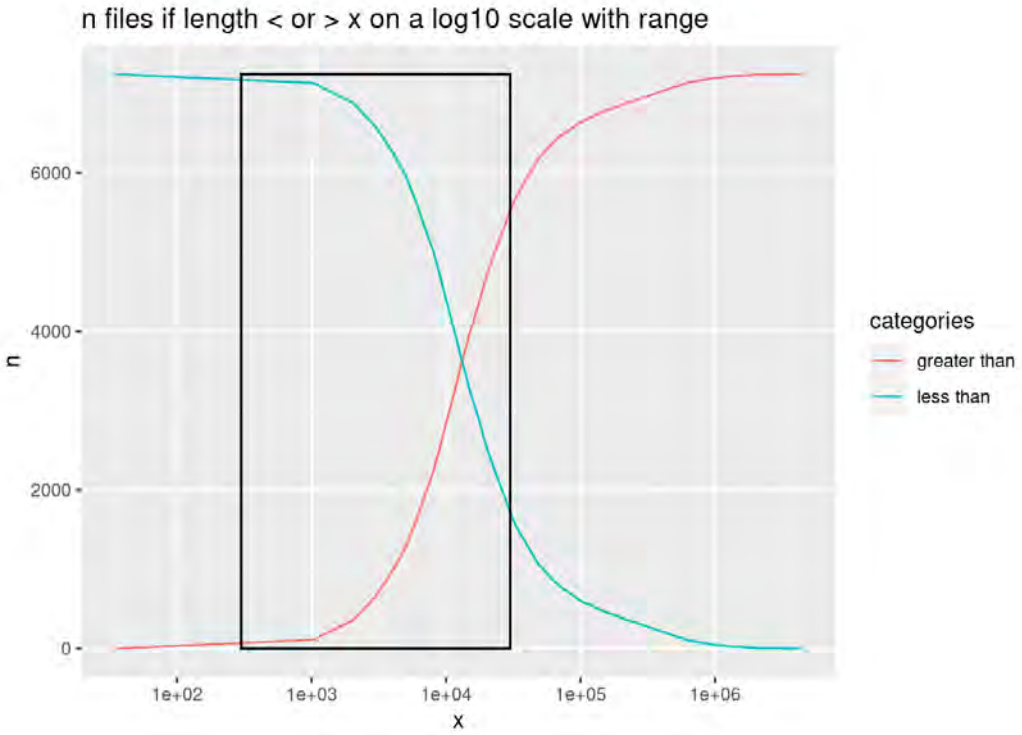
```
## Gesammt Länge der Dateien mit gefilterter Länge:  62128188 ≙ 62 MB
```

```
## Determine highlighted regions
v <- rep(0, length(length_fun_seq))
v[c(round(300/1000):round(30000/1000))] <- 1

## Get the start and end points for highlighted regions
inds <- diff(c(0, v))
start <- length_fun$x[inds == 1]
end <- length_fun$x[inds == -1]
if (length(start) > length(end)) end <- c(end, tail(length_fun$x, 1))

## highlight region data
rects <- data.frame(start=start, end=end, group=seq_along(start))

ggplot(length_fun, aes(x, n, col = categories)) +
  geom_line() +
  scale_x_continuous(trans = log10_trans()) +
  geom_rect(data=rects, inherit.aes=FALSE, aes(xmin=300, xmax=30000, ymin=min(length_fun$n),
                ymax=max(length_fun$n), group=group), color="black", fill="transparent", alpha=0) +
  ggtitle("n files if length < or > x on a log10 scale with range")
```



main.py


```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

######################################################################
# title:    BBS for Independence - data analysis using NLP
# subtitle: HSA in the ABC of Computational Text Analysis
# author:   Josias Bruderer, Universität Luzern
# date:     30. August 2021
# desc:     this script manages the whole analisis of textfiles.com
######################################################################

# Preparations [R1]
"""
The following lines of code is used for preparing our environment.
"""

# load the necessary libraries
import os
import shutil
import sys
import time
from pathlib import Path
import textacy
import spacy
import pandas as pd
import scattertext as st
from plotnine import *
import numpy as np
from multiprocessing import Pool
from multiprocessing.managers import BaseManager
import string
from octis.preprocessing.preprocessing import Preprocessing
# from octis.dataset.dataset import Dataset
# from octis.evaluation_metrics.diversity_metrics import TopicDiversity
# from octis.models.LDA import LDA
# from wordcloud import WordCloud
import csv

project_path = Path.cwd()

# prepare to load project specific libraries
if project_path not in sys.path:
    sys.path.append(str(project_path))

# import modules
from modules import wrangler
from modules import helpers
from modules import nlp_pool
from modules import years

# Data Wrangling [R2]
"""
In this section the required data is downloaded and preprocessed (f.E. unzipped).
The module data_wrangler will be used for this.
"""

number_of_threads = 24

"""
skip_steps = [] # skip nothing
skip_steps = ["download", "cleaning"] # use this after modification on metadata_file_filter
skip_steps = ["download", "cleaning", "metadata-filtering", "modeling", "analysis_freq", "analysis_advance_preparation",
              "analysis_scattertext", "analysis_year", "analysis_octis", "analysis_entities"] # the full list
"""
skip_steps = [] # the full list

data_url = "http://archives.textfiles.com/[name].zip"
data_names = ["100", "adventure", "anarchy", "apple", "art", "artifacts", "bbs", "computers", "conspiracy", "digest",
              "drugs", "etext", "exhibits", "floppies", "food", "fun", "games", "groups", "hacking", "hamradio",
              "history", "holiday", "humor", "internet", "law", "magazines", "media", "messages", "music", "news",
              "occult", "phreak", "piracy", "politics", "programming", "reports", "rpg", "science", "sex", "sf",
              "stories", "survival", "tap", "ufo", "uploads", "virus",
              "fidonet-on-the-internet"]  # categories to download
data_names_exclude = ["fidonet-on-the-internet", "tap", "floppies", "exhibits", "artifacts",
                      "piracy", "art", "magazines", "digest"]  # categories that are excluded and removed from data_names
file_filter = "^.*(\.(jpe?g|png|gif|bmp|zip|mp3|wav))|index\.html?$"  # use this to exclude by filenames
metadata_file_filter = "(x['metadata']['charratioB'] > 0.95) & " \
                       "(x['metadata']['length'] > 300) & (x['metadata']['length'] < 30000)"
metadata_file_filter_declaration = "(x['metadata']['charratioB'] > 0.8) & " \
                                   "(x['metadata']['length'] > 300) & (x['metadata']['length'] < 30000)"
data_dir = Path(project_path / "02_datasets/")
models_dir = Path(project_path / "03_workspace/models/")
analysis_dir = Path(project_path / "03_workspace/analysis/")
octis_dataset_dir = Path(project_path / "03_workspace/OCTIS/preprocessed_datasets/")
tmp_dir = Path(project_path / ".tmp/")

data_dir.mkdir(parents=True, exist_ok=True)
models_dir.mkdir(parents=True, exist_ok=True)
tmp_dir.mkdir(parents=True, exist_ok=True)

threads = []

if "download" not in skip_steps:
    # prepare the threads for loading data
    for names in helpers.chunker_list(data_names, number_of_threads):
        threads.append(wrangler.loader(tmp_dir, data_dir, data_url, names))

```

```
    # start all threads
    for thread in threads:
        thread.start()

    # wait for all threads to finish their work
    running = True
    while running:
        running = False
        for thread in threads:
            if thread.is_alive():
                running = True
                print("waiting for threads to finish...")
                time.sleep(1)

    print("data downloaded successfully")

if "cleaning" not in skip_steps:
    # prepare the threads for cleaning up stuff
    threads = []
    dataset = {}

    # prepare the threads
    for names in helpers.chunker_list(data_names, number_of_threads):
        threads.append(wrangler.cleaner(data_dir, names, file_filter))

    # add declaration
    threads.append(wrangler.cleaner(data_dir, ["declaration"], file_filter))

    # start all threads
    for thread in threads:
        thread.start()

    # wait for all threads to finish their work
    running = True
    while running:
        running = False
        for thread in threads:
            if thread.is_alive():
                running = True
                print("waiting for threads to finish...")
                time.sleep(1)
            else:
                if thread.data:
                    dataset.update(thread.data)
                    for d in thread.data:
                        helpers.save_object(thread.data[d], tmp_dir.joinpath(str(d + ".pkl")))

    helpers.save_object(dataset, tmp_dir.joinpath("dataset_full.pkl"))

    # write metadata to csv file
    print("Write dataset to csv")
    f = open(tmp_dir.joinpath("dataset.csv"), "w+")
    f.write("category,name,path,length,length_raw,avgcolumnsize,charratioA,charratioB,year,eyear,lyear,type\r\n")
    for d in dataset:
        for item in dataset[d]:
            f.write("\"" + d + "\",\"" + str(item["metadata"]["name"]) + "\",\"" +
                    str(item["metadata"]["path"]) + "\"," +
                    str(item["metadata"]["length"]) + "," +
                    str(item["metadata"]["length_raw"]) + "," +
                    str(item["metadata"]["avgcolumnsize"]) + "," +
                    str(item["metadata"]["charratioA"]) + "," +
                    str(item["metadata"]["charratioB"]) + ",\"" +
                    str(item["metadata"]["year"]) + "\"," +
                    str(item["metadata"]["eyear"]) + "," +
                    str(item["metadata"]["lyear"]) + ",\"" +
                    str(item["metadata"]["type"]) + "\"\r\n")
    f.close()

    shutil.copyfile(tmp_dir.joinpath("dataset.csv"),
                    models_dir.joinpath("dataset.csv")) # copy dataset.csv to models

    print("data cleaned successfully")
elif "metadata-filtering" not in skip_steps:
    # load dataset_full.pkl because it was not generate during runtime
    dataset = helpers.load_object(tmp_dir.joinpath("dataset_full.pkl"))
    print("data loaded from dataset_full.pkl")

if "metadata-filtering" not in skip_steps:
    for key in data_names_exclude:
        if key in dataset:
            del dataset[key]

    for key in dataset:
        d_tmp = []
        for x in dataset[key]:
            if key == "declaration" and eval(metadata_file_filter_declaration):
                d_tmp.append(x)
            elif eval(metadata_file_filter):
                d_tmp.append(x)
        dataset[key] = d_tmp

    helpers.save_object(dataset, tmp_dir.joinpath("dataset_filtered.pkl"))
    shutil.copyfile(tmp_dir.joinpath("dataset_filtered.pkl"),
                    models_dir.joinpath("dataset_filtered.pkl")) # copy dataset_filtered.pkl to models
elif "modeling" not in skip_steps:
    # load dataset_filtered.pkl because it was not generate during runtime
    dataset = helpers.load_object(tmp_dir.joinpath("dataset_filtered.pkl"))
    print("data loaded from dataset_filtered.pkl")

if "modeling" not in skip_steps:
```

```
    t0 = time.time()
    # calculate size of dataset
    dataset_size = 0
    for key in dataset:
        dataset_size = dataset_size + len(dataset[key])

    print("start building corpus")
    BaseManager.register('PoolCorpus', nlp_pool.PoolCorpus)

    if __name__ == '__main__':
        with BaseManager() as manager:
            corp = manager.PoolCorpus()
            corp.set_totalFilesTarget(dataset_size)
            with Pool(processes=number_of_threads) as pool:
                for key in dataset:
                    pool.map(corp.add, ((d["content"], d["metadata"]) for d in dataset[key]))
            corpus = corp.get()
            print("corpus loaded")
            corpus.save(tmp_dir.joinpath("corpus.bin.gz"))
    print("end building corpus")
    print("Time elapsed: ", time.time() - t0, "s")  # CPU seconds elapsed (floating point)

    shutil.copyfile(tmp_dir.joinpath("corpus.bin.gz"),
                    models_dir.joinpath("corpus.bin.gz")) # copy dataset_filtered.pkl to models

# load corpus.bin.gz always because freq cannot handle vanilla corpus
corpus = textacy.Corpus.load("en_core_web_sm", tmp_dir.joinpath("corpus.bin.gz"))
print("data loaded from corpus.bin.gz")

if "analysis_freq" not in skip_steps:
    print("start wordcount")
    # get lowercased and filtered corpus vocabulary (R3.3.1)
    vocab = corpus.word_counts(by='lemma_', filter_stops=True, filter_punct=True, filter_nums=True)
    vocab_doc = corpus.word_doc_counts(by='lemma_', filter_stops=True, filter_punct=True, filter_nums=True)

    # sort vocabulary by descending frequency
    vocab_sorted = sorted(vocab.items(), key=lambda x: x[1], reverse=True)
    vocab_sorted_doc = sorted(vocab_doc.items(), key=lambda x: x[1], reverse=True)

    # write to file, one word and its frequency per line
    with open(tmp_dir.joinpath('vocab_frq.txt'), 'w') as f:
        for word, frq in vocab_sorted:
            line = f"{word}\t{frq}\n"
            f.write(line)
    with open(tmp_dir.joinpath('vocab_frq_doc.txt'), 'w') as f:
        for word, frq in vocab_sorted_doc:
            line = f"{word}\t{frq}\n"
            f.write(line)

    shutil.copyfile(tmp_dir.joinpath("vocab_frq.txt"),
                    analysis_dir.joinpath("vocab_frq.txt")) # copy dataset_filtered.pkl to analysis
    shutil.copyfile(tmp_dir.joinpath("vocab_frq_doc.txt"),
                    analysis_dir.joinpath("vocab_frq_doc.txt"))  # copy dataset_filtered.pkl to analysis

    en = textacy.load_spacy_lang("en_core_web_sm")
    stats = []
    for cat in data_names + ["declaration"]:
        ctmp = corpus.get(lambda doc: doc._.meta["category"] == cat)
        dtmp = list(ctmp)
        ndocs = len(dtmp)
        nvocab = 0
        nlength = 0
        for d in dtmp:
            nvocab = nvocab + d.vocab.length
            nlength = nlength + d._.meta["length"]
        stats.append({"category": cat, "ndocs":  ndocs, "nvocab": nvocab, "nlength": nlength})
        if ndocs > 0:
            tmpcorpus = textacy.corpus.Corpus(en, data=dtmp)
            tmpvocab = tmpcorpus.word_counts(by='lemma_', filter_stops=True, filter_punct=True, filter_nums=True)
            tmpvocab_doc = tmpcorpus.word_doc_counts(by='lemma_', filter_stops=True, filter_punct=True, filter_nums=True)
            tmpvocab_sorted = sorted(tmpvocab.items(), key=lambda x: x[1], reverse=True)
            tmpvocab_sorted_doc = sorted(tmpvocab_doc.items(), key=lambda x: x[1], reverse=True)
            with open(tmp_dir.joinpath('vocab_frq_' + cat + '.txt'), 'w') as f:
                for word, frq in tmpvocab_sorted:
                    line = f"{word}\t{frq}\n"
                    f.write(line)
            with open(tmp_dir.joinpath('vocab_frq_doc_' + cat + '.txt'), 'w') as f:
                for word, frq in tmpvocab_sorted_doc:
                    line = f"{word}\t{frq}\n"
                    f.write(line)
            shutil.copyfile(tmp_dir.joinpath('vocab_frq_' + cat + '.txt'),
                            analysis_dir.joinpath('vocab_frq_' + cat + '.txt'))  # copy dataset_filtered.pkl to analysis
            shutil.copyfile(tmp_dir.joinpath('vocab_frq_doc_' + cat + '.txt'),
                            analysis_dir.joinpath('vocab_frq_doc_' + cat + '.txt'))  # copy dataset_filtered.pkl to analysis
    with open(tmp_dir.joinpath('stats.csv'), 'w') as csv_file:
        writer = csv.DictWriter(csv_file, stats[0].keys())
        writer.writeheader()
        writer.writerows(stats)
    shutil.copyfile(tmp_dir.joinpath('stats.csv'),
                    analysis_dir.joinpath('stats.csv'))  # copy stats.csv to analysis

    print("end wordcount")

if "analysis_advance_preparation" not in skip_steps:
    print("start advance preparation")
    # merge metadata and actual content for each document in the corpus
    # ugly, verbose syntax to merge two dictionaries
    data = [{**doc._.meta, **{'text': doc.text}} for doc in corpus]

    # create panda dataframe
```

```
    df = pd.DataFrame(data)

    df_sub = df[(df['text'].str.len() > 10)]

    # make new column containing all relevant metadata (showing in plot later on)
    df_sub['descripton'] = df_sub[['name', 'year', 'charratioB', 'avgcolumnsize']].astype(str).agg(', '.join, axis=1)

    helpers.save_object(df, tmp_dir.joinpath("df.pkl"))
    helpers.save_object(df_sub, tmp_dir.joinpath("df_sub.pkl"))
    print("end advance preparation")

    shutil.copyfile(tmp_dir.joinpath("df.pkl"),
                    models_dir.joinpath("df.pkl")) # copy df.pkl to models
    shutil.copyfile(tmp_dir.joinpath("df_sub.pkl"),
                    models_dir.joinpath("df_sub.pkl")) # copy df_sub.pkl to models
else:
    # load df.pkl and df_sub.pkl because it was not generate during runtime
    df = helpers.load_object(tmp_dir.joinpath("df.pkl"))
    df_sub = helpers.load_object(tmp_dir.joinpath("df_sub.pkl"))
    print("data loaded from df.pkl and df_sub.pkl")

if "analysis_scattertext" not in skip_steps:
    print("start scattertext")
    censor_tags = set(['CARD'])  # tags to ignore in corpus, e.g. numbers

    en = textacy.load_spacy_lang("en_core_web_sm")
    # stop words to ignore in corpus
    en_stopwords = spacy.lang.en.stop_words.STOP_WORDS  # default stop words
    custom_stopwords = set(['[', ']', '%'])
    en_stopwords = en_stopwords.union(custom_stopwords)  # extend with custom stop words

    # create corpus from dataframe
    # lowercased terms, no stopwords, no numbers
    # use lemmas for English only, German quality is too bad
    corpus_speeches = st.CorpusFromPandas(df_sub,  # dataset
                                          category_col='type',  # index differences by ...
                                          text_col='text',
                                          nlp=en,  # EN model
                                          feats_from_spacy_doc=st.FeatsFromSpacyDoc(tag_types_to_censor=censor_tags,
                                                                                    use_lemmas=True),
                                          ).build().get_stoplisted_unigram_corpus(en_stopwords)

    # produce visualization (interactive html)
    html = st.produce_scattertext_explorer(corpus_speeches,
                                           category='declaration',  # set attribute to divide corpus into two parts
                                           category_name='declaration',
                                           not_category_name='textfiles',
                                           metadata=df_sub['descripton'],
                                           width_in_pixels=1000,
                                           minimum_term_frequency=5,  # drop terms occurring less than 5 times
                                           save_svg_button=True,
                                           )

    # write visualization to html file
    fname = tmp_dir.joinpath("viz_declaration_textfiles.html")
    open(fname, 'wb').write(html.encode('utf-8'))
    print("end scattertext")
    shutil.copyfile(tmp_dir.joinpath("viz_declaration_textfiles.html"),
                    analysis_dir.joinpath("viz_declaration_textfiles.html")) # copy viz_declaration_textfiles.html to analysis

if "analysis_year" not in skip_steps:
    print("start year")

    df_sub = df[(df['text'].str.len() > 10)]

    # make new column containing all relevant metadata (showing in plot later on)
    df_sub['descripton'] = df_sub[['name', 'year', 'charratioB', 'avgcolumnsize']].astype(str).agg(', '.join, axis=1)

    dtmp = df_sub.groupby('eyear').agg({'text': "count"}).reset_index().rename(columns={'text': 'count'})
    dtmp = dtmp.rename(columns={"eyear": "year"})
    dtmp.insert(2, "type", "eyear")
    docs_per_year = dtmp

    dtmp = df_sub.groupby('lyear').agg({'text': "count"}).reset_index().rename(columns={'text': 'count'})
    dtmp = dtmp.rename(columns={"lyear": "year"})
    dtmp.insert(2, "type", "lyear")
    docs_per_year = docs_per_year.append(dtmp, ignore_index=True)

    # manual year was only available in top100 analysis
    # dtmp = pd.read_csv('top100_years.txt', delimiter=",").groupby('myear').agg({'text': "count"}).reset_index().rename(
    #    columns={'text': 'count'})
    # dtmp = dtmp.rename(columns={"myear": "year"})
    # dtmp.insert(2, "type", "myear")
    # docs_per_year = docs_per_year.append(dtmp, ignore_index=True)

    docs_per_year = docs_per_year[docs_per_year["year"] != "NA"]
    docs_per_year['year'] = pd.to_numeric(docs_per_year['year'])

    p = (ggplot(docs_per_year, aes('year', 'count', color='type', group='type'))
           + geom_point(alpha=0.5, stroke=0)
           + geom_line()
           + theme_classic()
           + labs(x="Year",
                  y="absolute number",
                  color="Legend")
           + theme(axis_text_x=element_text(angle=90, hjust=1))
           + scale_x_continuous(limits=(1960, 1999))
           )

    ggsave(plot=p, filename="docs_per_year", path=tmp_dir)
```

```
398
399     shutil.copyfile(tmp_dir.joinpath("docs_per_year.png"),
400                     analysis_dir.joinpath("docs_per_year.png"))  # copy docs_per_year.png to analysis
401
402     try:
403         dummy = pd.DataFrame(years.from_1960_to_1999)
404
405         e = dummy.append(docs_per_year[(docs_per_year['type'] == "eyear") &
406                                        (docs_per_year['year'] >= 1960)][["year", "count"]])\
407             .groupby('year').agg({'count': "sum"})
408         l = dummy.append(docs_per_year[(docs_per_year['type'] == "lyear") &
409                                        (docs_per_year['year'] >= 1960)][["year", "count"]])\
410             .groupby('year').agg({'count': "sum"})
411
412         print("r_{eyear mit lyear} = ", np.corrcoef(e["count"], l["count"])[0, 1])
413     except Exception as e:
414         print("something went wrong while calculating eyear / lyear variaty.")
415         pass
416
417     print("end year")
418
419 if "analysis_octis" not in skip_steps:
420     print("start octis")
421
422     df_sub.to_csv(tmp_dir.joinpath("corpus.txt"), "\t", columns = ["text"])
423
424     # Initialize preprocessing
425     preprocessor = Preprocessing(vocabulary=None, max_features=None,
426                                  remove_punctuation=True, punctuation=string.punctuation,
427                                  lemmatize=True, stopword_list='english',
428                                  min_chars=1, min_words_docs=0)
429
430     # preprocess
431     octis_dataset = preprocessor.preprocess_dataset(documents_path=tmp_dir.joinpath("corpus.txt"))
432
433     # save the preprocessed dataset
434     octis_dataset.save(str(tmp_dir.joinpath('octis_dataset')))
435
436     if os.path.exists(octis_dataset_dir.joinpath('octis_dataset')):
437         shutil.rmtree(octis_dataset_dir.joinpath('octis_dataset'))
438     if os.path.exists(models_dir.joinpath('octis_dataset')):
439         shutil.rmtree(models_dir.joinpath('octis_dataset'))
440
441     shutil.copytree(tmp_dir.joinpath('octis_dataset'),
442                     octis_dataset_dir.joinpath('octis_dataset'))  # copy octis_dataset to datasets
443     shutil.copytree(tmp_dir.joinpath('octis_dataset'),
444                     models_dir.joinpath('octis_dataset'))  # copy octis_dataset to models
445
446     # octis will be processed using dashboard
447     # model = LDA(num_topics=5)  # Create model
448     # model_output = model.train_model(octis_dataset)  # Train the model
449
450     # metric = TopicDiversity(topk=10)  # Initialize metric
451     # topic_diversity_score = metric.score(model_output)  # Compute score of the metric
452
453     # print("topic diversity score:", topic_diversity_score)
454
455     # wordcloud will be generated outside of python
456     # wocl = WordCloud(mode="RGBA", background_color="white").generate(" ".join(model_output["topics"][0]))
457     # wocl2 = WordCloud(mode="RGBA", background_color="white", relative_scaling=1, scale=10, max_words=9999,
458     #                   min_font_size=1, max_font_size=18, collocations=False).generate(" ".join(model_output["topics"][0]))
459     # image = wocl.to_image()
460     # image.save(tmp_dir.joinpath("wordcloud.png"))
461     # image2 = wocl2.to_image()
462     # image2.save(tmp_dir.joinpath("wordcloud2.png"))
463
464     print("end octis")
465
466 if "analysis_entities" not in skip_steps:
467     entities = []
468
469     for doc in corpus.docs:
470         for ent in textacy.extract.entities(doc):
471             try:
472                 entities += [{"text": ent.text, "label": ent.label_, "explain": spacy.explain(ent.label_)}]
473             except:
474                 print("Problem with:", doc._.meta["name"])
475
476     # export corpus as csv
477     f_csv = tmp_dir.joinpath('entities.csv')
478     textacy.io.csv.write_csv(entities, f_csv, fieldnames=entities[0].keys())
479
480     shutil.copyfile(tmp_dir.joinpath('entities.csv'),
481                     analysis_dir.joinpath('entities.csv'))  # copy entities.csv to analysis
482
483     df_entities = pd.DataFrame(entities, columns=['text', 'label', 'explain'])
484     df_entities_count = df_entities.groupby('text').agg({'label': "count"}).rename(
485         columns={'label': 'count'}).sort_values(by=['count'], ascending=False).reset_index()
486
487     # write to file, one word and its frequency per line
488     fname = tmp_dir.joinpath('entities_frq.csv')
489     with open(fname, 'w') as f:
490         for i, d in df_entities_count.iterrows():
491             line = d["text"] + "," + str(d["count"]) + "\n"
492             f.write(line)
493
494     shutil.copyfile(tmp_dir.joinpath('entities_frq.csv'),
495                     analysis_dir.joinpath('entities_frq.csv'))  # copy entities_frq.csv to analysis
496
497     print("entities: \n", df_entities_count[:25])
498
499 print("everything done.")
```

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

############################################################
# title:  datawrangler
# author: Josias Bruderer
# date:   27.08.2021
# desc:   this module takes care of all tasks that are
#         related to juggling files and datasets.
############################################################

import os
import sys
from threading import Thread
from pathlib import Path
import requests
import zipfile
import codecs
import re


def averageLen(lst, excludeEmpty=True):
    if excludeEmpty:
        lengths = [len(i) for i in lst if i != "" and i != " "]
    else:
        lengths = [len(i) for i in lst]
    return 0 if len(lengths) == 0 else round((float(sum(lengths)) / len(lengths)), 2)


def daterange(lst, t="r"):
    ltmp = []
    ltmp2 = []
    if len(lst) > 0:
        for l in lst:
            ltmp += list(filter(None, l))
        for l in ltmp:
            if len(l) == 2:
                ltmp2 += ["19" + l]
            else:
                ltmp2 += [l]
        if len(ltmp2) > 2:
            if t == "e":
                return str(min(ltmp2))
            elif t == "l":
                return str(max(ltmp2))
            else:
                return str(min(ltmp2) + "-" + max(ltmp2))
        else:
            return str(ltmp2[0])
    else:
        return "NA"

class cleaner(Thread):

    def __init__(self, data_dir, data_names, file_filter):
        Thread.__init__(self)
        self.data_dir = data_dir
        self.data_names = data_names
        self.file_filter = file_filter
        self.data = {}

    def run(self):
        for data_name in self.data_names:
            self.data[data_name] = self.get_texts(Path(self.data_dir, data_name), data_name)

    def get_texts(self, dir_texts, data_name):
        """
        Sequentially stream all documents from a given folder,
        including metadata.
        """
        data = []

        # iterate over all documents
        for fname in dir_texts.glob('**/*'):  # ** = all subdirectories
            if Path(fname).is_file():
                print("processing in " + str(dir_texts.stem) + " file: " + str(fname))

                if re.match(self.file_filter, fname.name):
                    print("skip in " + str(dir_texts.stem) + " because of file_filter: " + str(fname))
                    continue
                # Read file content and replace encoding erros
                content_raw = codecs.open(fname, 'r', encoding='utf-8', errors='replace').read()

                # join lines as there are hard line-breaks
                content = content_raw.replace('\r\n', ' ')
                content = content.replace('\r', ' ')
                content = content.replace('\n', ' ')
```

```
                content = content.replace('\t', ' ')
                content = content.replace('\x1a', ' ')
                content = re.sub('[^A-z0-9\ \.\'\,\!]', ' ', content)
                content = re.sub('[\\\\\^\[\]]', ' ', content)

                # add more metadata here if needed
                # charratio: 0 = no character is "text", 1 = every character is "text"
                if len(content_raw) == 0:
                    charratioA = 0
                    charratioB = 0
                else:
                    charratioA = round(1 / len(content_raw) * len(re.findall("[A-z]", content_raw)), 2)
                    charratioB = round(1 / len(content_raw) * len(re.findall("[A-z\ \.\"\,\!]", content_raw)), 2)

                typ = "textfile"
                if fname.name == "declarationbarlow1996.txt" or data_name == "declaration":
                    typ = "declaration"

                rxdate = re.compile(
                    'copyright.{0,3}(19[6-9][0-9])|updated.{0,3}[0-1]?[0-9]?-[0-3]?[0-9]?-([6-9][0-9])|'
                    'Date\:.*([6-9][0-9]).*,|(?:jan(?:uary)?|feb(?:ruary)?|mar(?:ch)?|apr(?:il)?|may|'
                    'june|july|aug(?:ust)?|sept(?:ember)?|oct(?:ober)?|nov(?:ember)?|dec(?:ember)?)'
                    '.{0,8}(1?9?[6-9][0-9])|[0-1]?[0-9]?\/[0-3]?[0-9]?\/([6-9][0-9])|[0-1]?[0-9]?-[0-3]?[0-9]?'
                    '-([6-9][0-9])|[^-](19[6-9][0-9])')

                matches = rxdate.findall(content, re.IGNORECASE)

                metadata = {'name': fname.name,
                            'path': str(fname),
                            'length_raw': len(content_raw),
                            'length': len(content),
                            'avgcolumnsize': averageLen(content_raw.splitlines()),
                            'charratioA': charratioA,
                            'charratioB': charratioB,
                            'year': daterange(matches),
                            'eyear': daterange(matches, "e"),
                            'lyear': daterange(matches, "l"),
                            'type': typ,
                            'category': data_name,
                            }

                # return documents one after another (sequentially)
                data.append({"content": content, "metadata": metadata})
        return data

class loader(Thread):

    def __init__(self, tmp_dir, data_dir, data_url, data_names):
        Thread.__init__(self)
        self.tmp_dir = tmp_dir
        self.data_dir = data_dir
        self.data_url = data_url
        self.data_names = data_names
        self.init()

    def init(self):
        try:
            # create tmp directory if not existing yet
            if not os.path.exists(self.tmp_dir.is_dir()):
                os.mkdir(self.tmp_dir)

            # create data directory if not existing yet
            if not os.path.exists(self.data_dir):
                os.mkdir(self.data_dir)
        except:
            print("Unexpected error: ", sys.exc_info()[0])
            raise

    def run(self):
        for data_name in self.data_names:
            url = self.data_url.replace("[name]", data_name)
            zipdir = Path(self.tmp_dir, str(data_name + ".zip"))
            self.download_zip(url, zipdir)
            self.extract_zip(zipdir)

    def download_zip(self, url, zipdir):
        try:
            if not zipdir.is_file():
                print("Downloading: " + url)
                r = requests.get(url, allow_redirects=True)
                open(zipdir, 'wb').write(r.content)
            else:
                print("Skip downloading, file already downloaded: " + url)
        except:
            print("Unexpected error: ", sys.exc_info()[0])
            raise

    def extract_zip(self, zipdir):
        try:
            if not Path(self.data_dir / zipdir.stem).is_dir():
```

wrangler.py

```
178                 print("Extracting: " + str(zipdir))
179                 with zipfile.ZipFile(zipdir, 'r') as zip_ref:
180                     zip_ref.extractall(self.data_dir)
181             else:
182                 print("Skip extracting, file already extracted: " + str(zipdir))
183         except:
184             print("Unexpected error: ", sys.exc_info()[0])
185             raise
186
```

modules/helpers.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

############################################################
# title:  helpers
# author: Josias Bruderer
# date:   25.08.2021
# desc:   this module provides some useful functions
############################################################

import pickle


def save_object(obj, filename):
    with open(filename, 'wb') as outp:  # Overwrites any existing file.
        pickle.dump(obj, outp, pickle.HIGHEST_PROTOCOL)

def load_object(filename):
    with open(filename, 'rb') as inp:
        return pickle.load(inp)

def chunker_list(seq, size):
    return (seq[i::size] for i in range(size))

```

modules/nlp_pool.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

############################################################
# title:  helpers
# author: Josias Bruderer
# date:   28.08.2021
# desc:   this module provides nlp functions
############################################################

import textacy
import spacy
# run: ./.envs/bin/python -m spacy download en_core_web_sm


class PoolCorpus(object):

    def __init__(self):
        model = spacy.load('en_core_web_sm', disable=["parser"])
        model.max_length = 10000000  # enable utilization of ~ 100GB RAM
        self.corpus = textacy.corpus.Corpus(lang=model)
        self.totalFilesTarget = 1
        self.processedFiles = 0

    def add(self, data):
        self.corpus.add(data)
        self.processedFiles = self.processedFiles + 1
        print("Processed ", self.processedFiles, " of ", self.totalFilesTarget, "files: ",
              round(100/self.totalFilesTarget*self.processedFiles, 4), "%")

    def get(self):
        return self.corpus

    def save(self, path):
        self.corpus.save(path)

    def set_totalFilesTarget(self, n):
        self.totalFilesTarget = n

"""
texts = {
        'key1': 'First text 1.',
        'key2': 'Second text 2.',
        'key3': 'Third text 3.',
        'key4': 'Fourth text 4.',
    }

BaseManager.register('PoolCorpus', PoolCorpus)

if __name__ == '__main__':
    with BaseManager() as manager:
        corpus = manager.PoolCorpus()

        with Pool(processes=2) as pool:
            pool.map(corpus.add, ((v, {'key': k}) for k, v in texts.items()))

        print(corpus.get())
"""
```

modules/years.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

############################################################
# title:  helpers
# author: Josias Bruderer
# date:   29.08.2021
# desc:   this module provides years variable
############################################################

from_1960_to_1999 = [{"year": 1960, "count": 0},
                     {"year": 1961, "count": 0},
                     {"year": 1962, "count": 0},
                     {"year": 1963, "count": 0},
                     {"year": 1964, "count": 0},
                     {"year": 1965, "count": 0},
                     {"year": 1966, "count": 0},
                     {"year": 1967, "count": 0},
                     {"year": 1968, "count": 0},
                     {"year": 1969, "count": 0},
                     {"year": 1970, "count": 0},
                     {"year": 1971, "count": 0},
                     {"year": 1972, "count": 0},
                     {"year": 1973, "count": 0},
                     {"year": 1974, "count": 0},
                     {"year": 1975, "count": 0},
                     {"year": 1976, "count": 0},
                     {"year": 1977, "count": 0},
                     {"year": 1978, "count": 0},
                     {"year": 1979, "count": 0},
                     {"year": 1980, "count": 0},
                     {"year": 1981, "count": 0},
                     {"year": 1982, "count": 0},
                     {"year": 1983, "count": 0},
                     {"year": 1984, "count": 0},
                     {"year": 1985, "count": 0},
                     {"year": 1986, "count": 0},
                     {"year": 1987, "count": 0},
                     {"year": 1988, "count": 0},
                     {"year": 1989, "count": 0},
                     {"year": 1990, "count": 0},
                     {"year": 1991, "count": 0},
                     {"year": 1992, "count": 0},
                     {"year": 1993, "count": 0},
                     {"year": 1994, "count": 0},
                     {"year": 1995, "count": 0},
                     {"year": 1996, "count": 0},
                     {"year": 1997, "count": 0},
                     {"year": 1998, "count": 0},
                     {"year": 1999, "count": 0}]

```

8.5 Anhang 5: Resultate

Dieser Anhang dient der Nachvollziehbarkeit. Die vollständigen Resultate inkl. Korpus sind zu finden unter:
https://github.com/josiasbruderer/bbs-for-independence/tree/main/03_workspace/states (zugegriffen 30.08.2021)

8.5.1 Dokumente pro Jahr

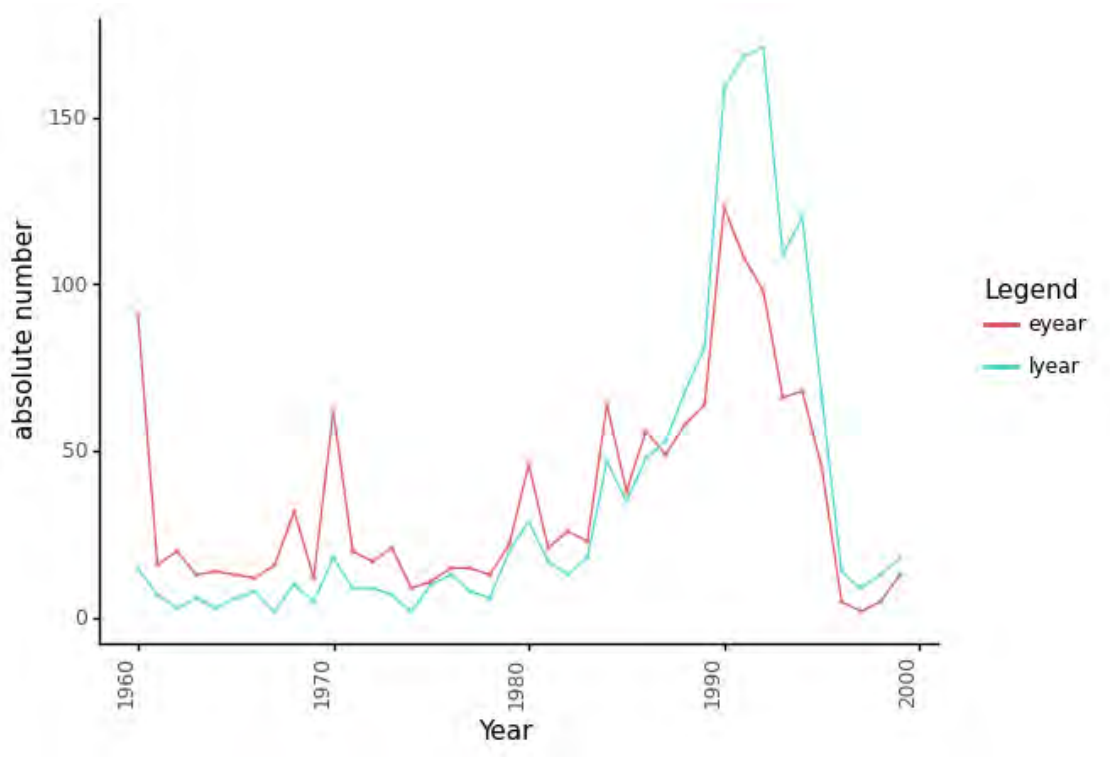


Abbildung 4: Dokumente pro Jahr: Gesamtdatensatz ohne Kategorie «sex»

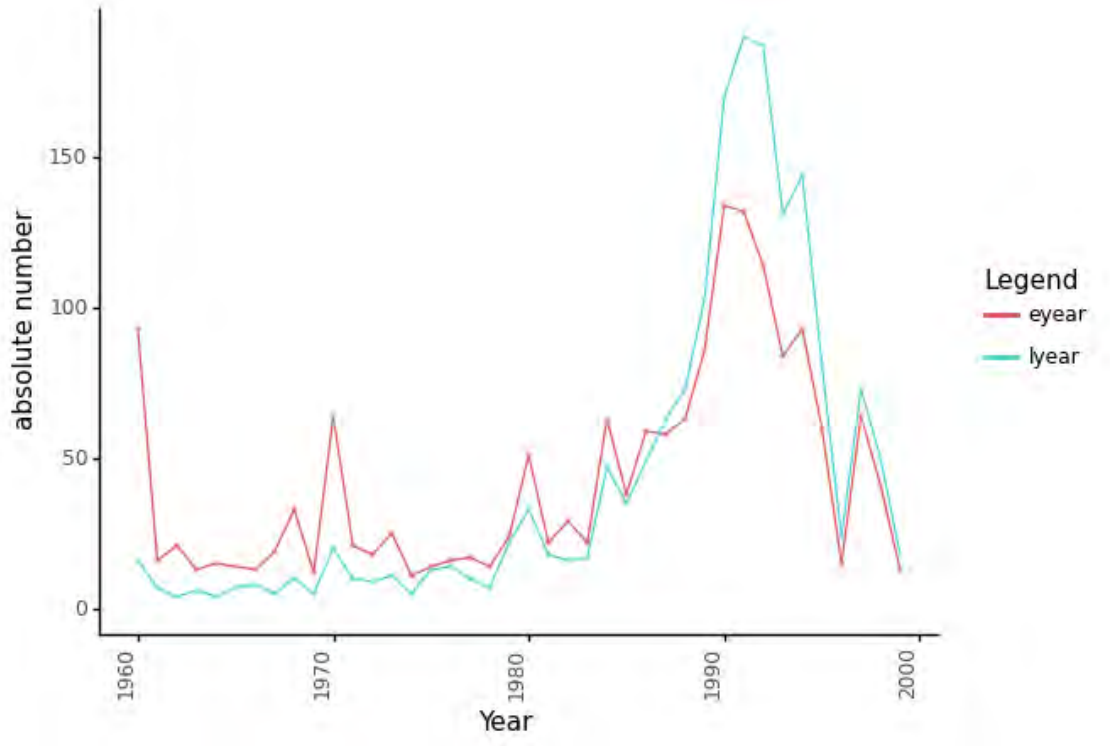


Abbildung 5: Dokumente pro Jahr: Gesamtdatensatz

8.5.2 Kategoriengrösse

Anzahl Dokumente pro Kategorie nach Bereinigung und Filterung:

- state_095_300_30000_categories/analysis/stats.csv

category	ndocs	nvocab	nlength	ndocp	nvocabp	ndocp
art	0	0	0	0.00%	0.00%	0.00%
artifacts	0	0	0	0.00%	0.00%	0.00%
digest	0	0	0	0.00%	0.00%	0.00%
exhibits	0	0	0	0.00%	0.00%	0.00%
floppies	0	0	0	0.00%	0.00%	0.00%
magazines	0	0	0	0.00%	0.00%	0.00%
piracy	0	0	0	0.00%	0.00%	0.00%
tap	0	0	0	0.00%	0.00%	0.00%

category	ndocs	nvocab	nlength	ndocp	nvocabp	ndocp
fidonet-on-the-internet	0	0	0	0.00%	0.00%	0.00%
messages	1	186697	3832	0.02%	0.02%	0.01%
declaration	1	186697	5089	0.02%	0.02%	0.01%
100	2	373394	15490	0.04%	0.04%	0.02%
media	5	933485	70127	0.09%	0.09%	0.11%
holiday	7	1306879	35331	0.13%	0.13%	0.06%
science	8	1493576	36846	0.15%	0.15%	0.06%
programming	14	2613758	196115	0.25%	0.25%	0.32%
rpg	15	2800455	128295	0.27%	0.27%	0.21%
food	15	2800455	139286	0.27%	0.27%	0.22%
reports	17	3173849	102885	0.31%	0.31%	0.17%
hamradio	17	3173849	119696	0.31%	0.31%	0.19%
virus	17	3173849	180867	0.31%	0.31%	0.29%
computers	22	4107334	222865	0.40%	0.40%	0.36%
internet	29	5414213	311059	0.53%	0.53%	0.50%
survival	29	5414213	322363	0.53%	0.53%	0.52%
history	30	5600910	326665	0.54%	0.54%	0.53%
fun	35	6534395	272652	0.64%	0.64%	0.44%
music	39	7281183	273950	0.71%	0.71%	0.44%
news	43	8027971	426174	0.78%	0.78%	0.69%
hacking	48	8961456	393191	0.87%	0.87%	0.63%
games	49	9148153	456712	0.89%	0.89%	0.74%
phreak	53	9894941	397485	0.96%	0.96%	0.64%

category	ndocs	nvocab	nlength	ndocp	nvocabp	ndocp
groups	59	11015123	769780	1.07%	1.07%	1.24%
conspiracy	63	11761911	767596	1.14%	1.14%	1.24%
ufo	71	13255487	816498	1.29%	1.29%	1.31%
law	76	14188972	862562	1.38%	1.38%	1.39%
bbs	79	14749063	525500	1.43%	1.43%	0.85%
adventure	79	14749063	716661	1.43%	1.43%	1.15%
anarchy	107	19976579	941863	1.94%	1.94%	1.52%
sf	123	22963731	1652889	2.23%	2.23%	2.66%
stories	138	25764186	1033873	2.50%	2.50%	1.66%
humor	152	28377944	724643	2.76%	2.76%	1.17%
uploads	158	29498126	875748	2.87%	2.87%	1.41%
apple	166	30991702	1723696	3.01%	3.01%	2.77%
drugs	187	34912339	1510700	3.39%	3.39%	2.43%
etext	240	44807280	2918710	4.36%	4.36%	4.70%
occult	373	69637981	3608148	6.77%	6.77%	5.81%
politics	758	141516326	7874026	13.76%	13.76%	12.67%
sex	2185	407932945	30368320	39.66%	39.66%	48.88%

8.5.3 Worthäufigkeiten

Worthäufigkeit pro Datenset nach Bereinigung und Filterung

- state_080_300_30000_declaration/analysis/vocab_frq.txt
- state_095_300_30000_categories/analysis/vocab_frq.txt
- state_095_300_30000_categories/analysis/vocab_frq_no-sex.txt

Declaration_Word	Declaration_n	All_Word	All_n	no-sex_Word	no-sex_n
file	48	time	25326	Time	11434
like	44	like	24493	People	10004
board	44	hand	24483	like	8757
think	33	say	24107	know	8281
world	31	feel	22968	say	7354
time	30	look	22044	work	7256
know	30	come	21704	FIND	7088
start	29	know	20771	use	7010
go	27	go	20255	come	6840
write	25	cock	18508	good	6647
government	24	get	17421	Way	6538
good	24	want	16383	man	6149
new	24	think	15538	new	6092
textfile	23	begin	15214	go	6057
people	22	take	14443	thing	5683
work	22	tell	14323	right	5673
call	21	way	14068	look	5638
number	21	start	13978	System	5506
look	20	good	13852	think	5397
come	20	little	13829	year	5146
line	20	head	13416	need	5084
great	20	long	13110	state	4930
day	19	work	12686	long	4923

Declaration_Word	Declaration_n	All_Word	All_n	no-sex_Word	no-sex_n
want	18	find	12431	world	4916
message	18	hard	12265	DAY	4894
law	17	pull	12182	want	4890
thing	17	man	12152	life	4861
stuff	17	mouth	12045	take	4705
use	16	face	11880	computer	4651
get	16	right	11856	place	4457
long	16	body	11839	get	4439
person	15	turn	11503	leave	4355
life	15	leg	10987	Power	4161
computer	15	let	10898	tell	4142
year	15	people	10761	call	4041
public	14	finger	10335	END	3844
state	14	eye	10332	try	3837
begin	14	thing	10165	Great	3822
find	14	try	10140	Number	3812
right	13	leave	10048	little	3727
Jason	13	ask	10034	government	3714
leech	13	see	10030	see	3714
try	12	pussy	10007	point	3705
matter	12	open	9304	give	3697
friend	12	hold	9069	Group	3635
hard	12	place	8729	law	3527

Declaration_Word	Declaration_n	All_Word	All_n	no-sex_Word	no-sex_n
fact	12	ass	8688	information	3467
BBS	12	stand	8553	ASK	3437
program	12	room	8502	hand	3433
shall	11	day	8493	Turn	3425
take	11	girl	8311	File	3410
phone	11	use	8309	mean	3405
send	11	move	8295	form	3347
tell	11	give	8210	feel	3286
system	11	year	8173	Old	3267
word	11	need	8065	high	3256
abuse	11	woman	7905	let	3193
technology	11	tongue	7878	run	3187
pq	11	watch	7862	start	3175
fish	11	lip	7768	change	3167
free	10	away	7650	Order	3162
key	10	sit	7560	OPEN	3150
power	10	run	7510	Control	3102
copy	10	reach	7486	set	3079
way	10	suck	7466	game	3055
need	10	fuck	7461	case	3047
post	10	close	7460		
information	10				
guy	10				

Declaration_Word	Declaration_n	All_Word	All_n	no-sex_Word	no-sex_n
disk	10				
raise	10				

8.5.4 Entitäten

Worthäufigkeit pro Datenset nach Bereinigung und Filterung

- state_080_300_30000_declaration/analysis/entities_frq.csv
- state_095_300_30000_categories/analysis/entities_frq.csv
- state_095_300_30000_categories/analysis/entities_frq_no-sex.csv

Declaration_Word	Declaration_n	All_Word	All_n	no-sex_Word	no-sex_n
one	11	first	10343	one	4813
two	10	two	8921	first	4708
first	9	one	8624	two	4051
Megaterm	6	three	3325	three	1656
AE	6	second	2648	One	1313
today	6	One	2156	second	1178
SysOp	5	Linda	1679	1	1169
Cyberspace	5	Kim	1573	2	1100
United States	4	Lisa	1508	American	1011
three	4	four	1506	3	984
10	4	Karen	1448	today	934
Soviet	4	today	1373	four	884
300 1200	4	2	1294	America	644

Declaration_Word	Declaration_n	All_Word	All_n	no-sex_Word	no-sex_n
1	4	1	1261	United States	622
One	3	3	1174	State	569
51	3	half	1160	4	568
Octothorpe	3	Jim	1149	5	557
Jason	3	American	1104	U.S.	548
Chuck Hammill	3	day	1041	10	547
Thomas Covenant	3	Mike	1027	First	525
Soviets	3	Chris	995	third	495
Constitution	3	third	942	CIA	485
2.0	3	Cindy	919	day	475
Sought After	2	Dave	902	Earth	437
50	2	First	885	US	437
200	2	Mary	880	Americans	433
Congress	2	Susan	874	British	430
IBM	2	night	842	half	426
dozen	2	Bill	838	6	415
Washington	2	John	808	English	408
RSA	2	tonight	792	years	400
about 50	2	few minutes	785	FBI	398
Ansi	2	5	730	12	357
about 5	2	Kevin	723	Europe	348
America	2	Jack	716	five	348
8	2	Jeff	705	Mulder	347

Declaration_Word	Declaration_n	All_Word	All_n	no-sex_Word	no-sex_n
914	2	4	699	Picard	346
Digital Dimension	2	10	681	Congress	345
American	2	Carol	667	Christian	337
WAITFOR	2	America	661	year	335
zero	2	Mark	644	German	323
second	2	years	639	Apple	322
OPEC	2	United States	635	20	312
Moscow	2	five	630	Two	309
ftp	2	Tom	628	8	301
Nixon	2	Steve	616	night	297
hundred	2	Bob	602	England	294
PGP	2	Sally	593	seven	293
Jason Scott	2	Sharon	590	six	286

8.5.5 Scattertext

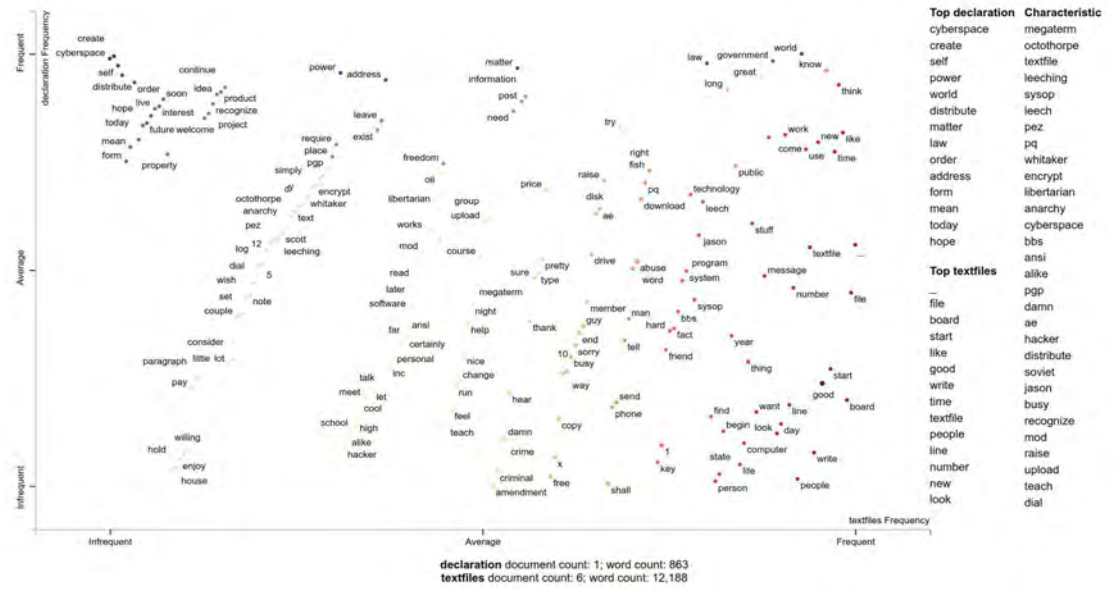


Abbildung 6: Scattertext: Declaration zu Gesamtdatensatz

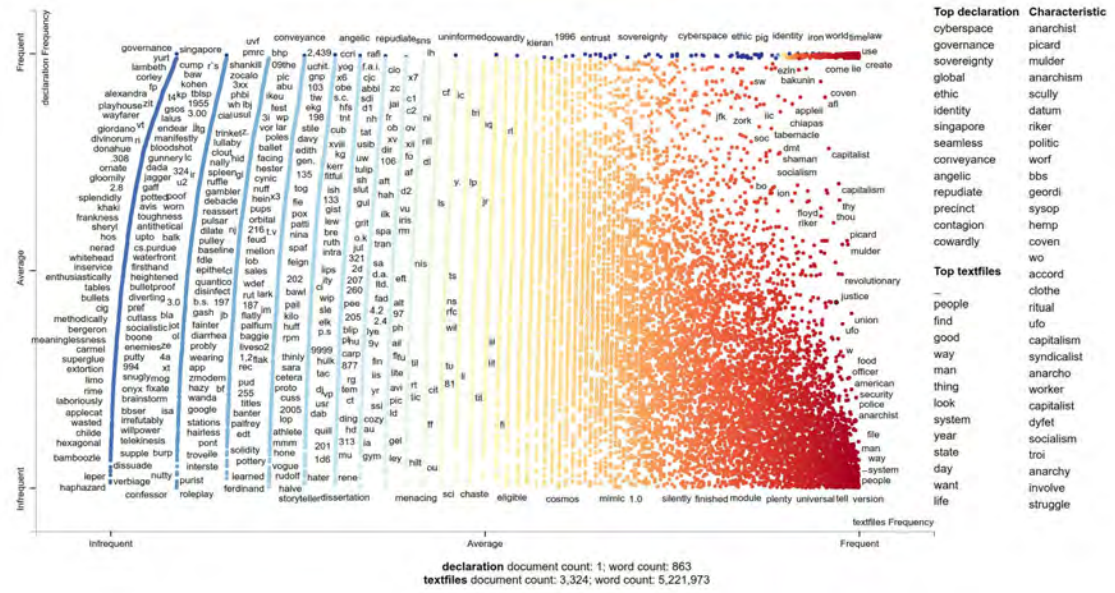


Abbildung 7: Scattertext: Declaration zu Gesamtdatensatz ohne Kategorie «sex»

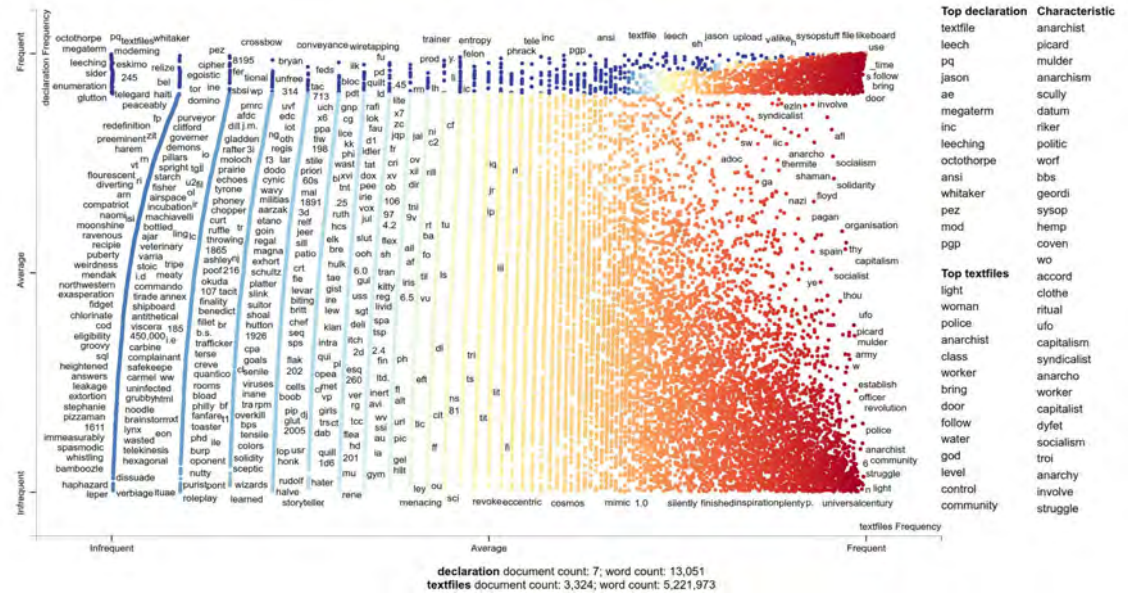


Abbildung 8: Scattertext: Declaration ähnlich zu Gesamtdatensatz ohne Kategorie «sex»

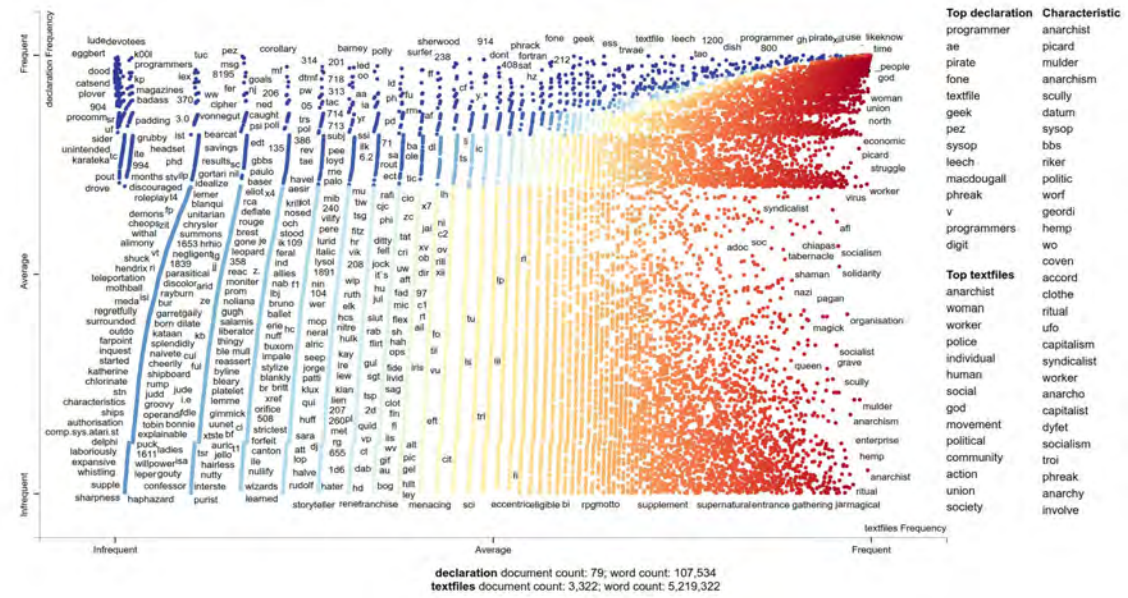


Abbildung 9: Scattertext: Kategorie 100 zu Gesamtdatensatz ohne Kategorie «sex»

8.5.6 Topic Diversity

Diese Grafiken sind eine manuelle Auswahl. Es ist jeweils vermerkt, zu welcher Iteration, Run und Topic die Grafik gehört.

Abbildung 10: Topic Diversity: Declaration LDA - Iteration: 4, Run: 1, Topic: 1

Abbildung 11: Topic Diversity: Declaration NeuralLDA - Iteration: 3, Run: 2, Topic: 11

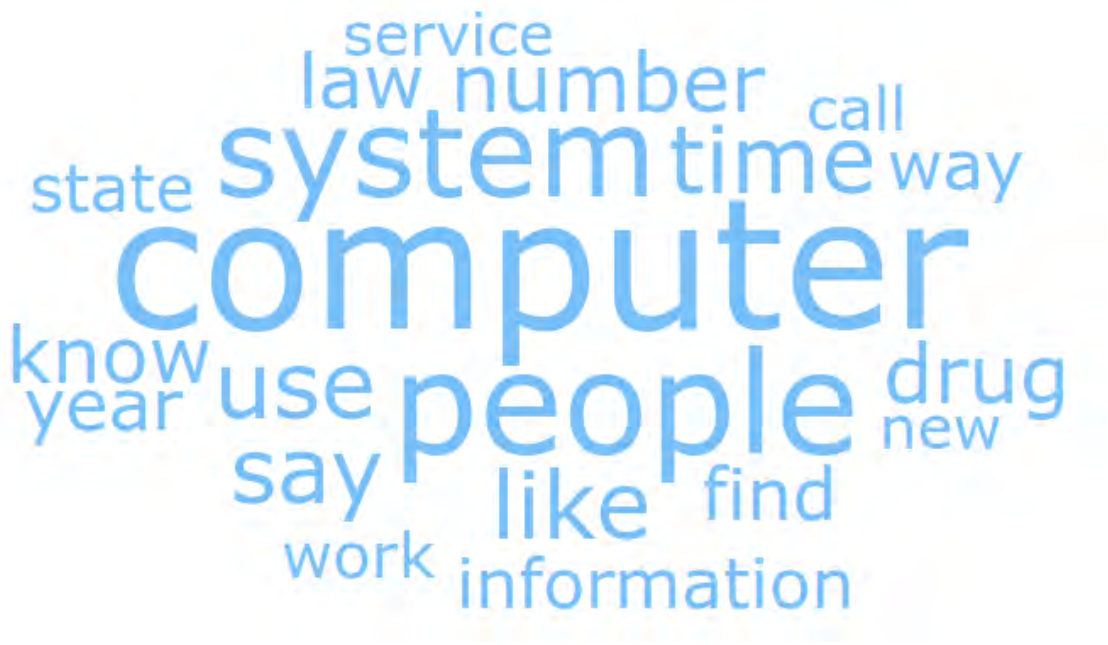


Abbildung 12: Topic Diversity: Gesamtdatensatz LDA - Iteration: 3, Run: 0, Topic: 0

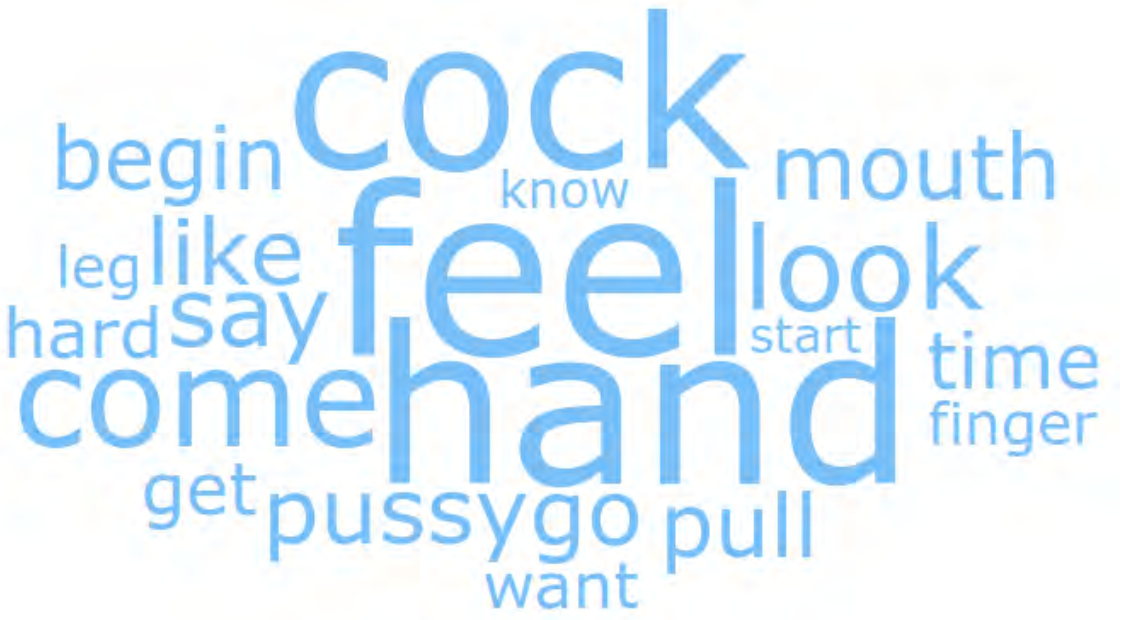


Abbildung 13: Topic Diversity: Gesamtdatensatz LDA - Iteration: 3 Run: 1, Topic: 1

8.6 Anhang 6: Messmethode lyear

Zur Identifikation des Ursprungsjahres dient folgender Python Code:

```
def daterange(lst, t="r"):
    ltmp = []
    ltmp2 = []
    if len(lst) > 0:
        for l in lst:
            ltmp += list(filter(None, l))
        for l in ltmp:
            if len(l) == 2:
                ltmp2 += ["19" + l]
            else:
                ltmp2 += [l]
        if len(ltmp2) > 2:
            if t == "e":
                return str(min(ltmp2))
            elif t == "l":
                return str(max(ltmp2))
            else:
                return str(min(ltmp2) + "-" + max(ltmp2))
        else:
            return str(ltmp2[0])
    else:
        return "NA"

rxdate = re.compile('copyright.{0,3}(19[6-9][0-9])|'
                    'updated.{0,3}[0-1]?[0-9]?-[0-3]?[0-9]?'
                    '-([6-9][0-9])|Date\:.*([6-9][0-9]).*,|'
                    '(?:jan(?:uary)?|feb(?:ruary)?|mar(?:ch)?|'
                    'apr(?:il)?|may|june|july|aug(?:ust)?|'
```

```
                    'sept(?:ember)?|oct(?:ober)?|nov(?:ember)?'
                    '|dec(?:ember)?).{0,8}(1?9?[6-9][0-9])|'
                    '[0-1]?[0-9]?\/[0-3]?[0-9]?\/([6-9][0-9])|'
                    '[0-1]?[0-9]?-[0-3]?[0-9]?-([6-9][0-9])|'
                    '[^-](19[6-9][0-9])')

content = "Test content. New York, 01-01-1992"

matches = rxdate.findall(content, re.IGNORECASE)

metadata = {
'year': daterange(matches),
'eyear': daterange(matches, "e"),
'lyear': daterange(matches, "l")
}
```

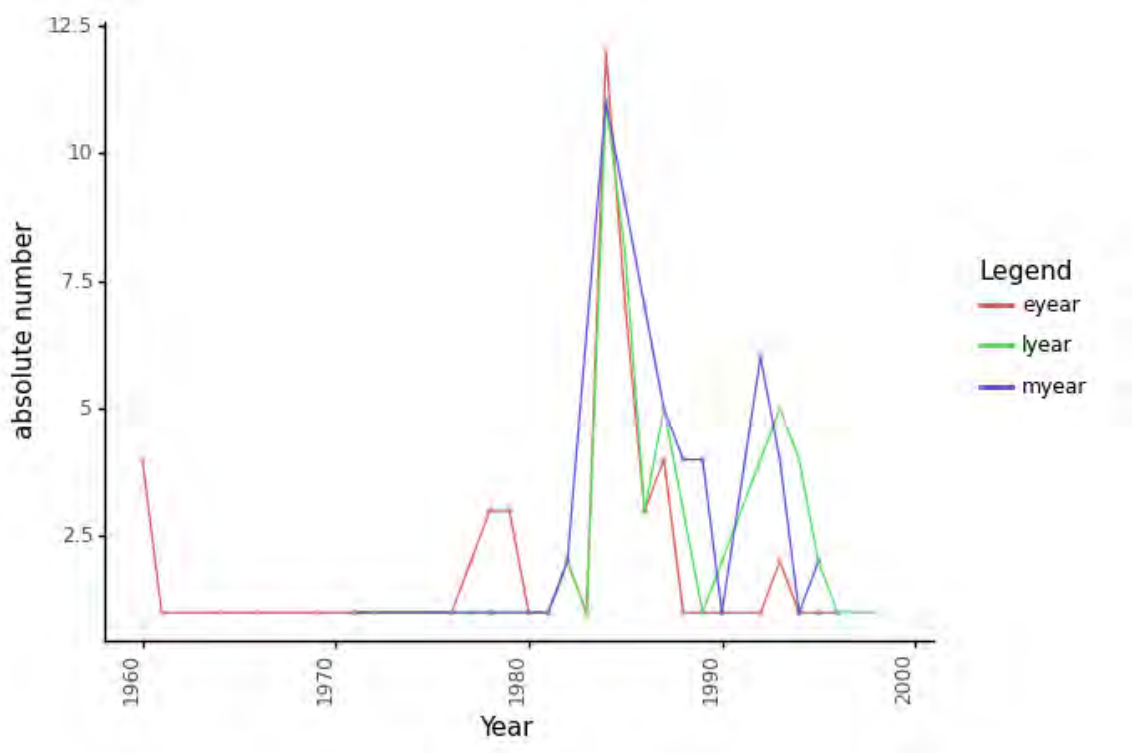


Abbildung 14: Dokumente pro Jahr: Kategorie «100»

Diese Auswertung überprüft, wie akkurat die gewählte Identifikation des Ursprungsjahres eines Textfiles ist. Die manuelle Identifikation des Jahres (myear) stellt den Referenzwert dar. Das älteste respektive frühst genanntes Jahr (eyear = earliest year) ist ungenauer ($r = 0.77$) als das jüngste respektive neuste genannte Jahr (lyear = latest year) ($r = 0.91$).

Bei der Untersuchung des Gesamtdatensatz (bereinigt und gefiltert von textfiles.com) konnte damit die Popularität von BBS gemessen werden. Die Korrelation zwischen lyear und eyear entspricht $r = 0.91$.

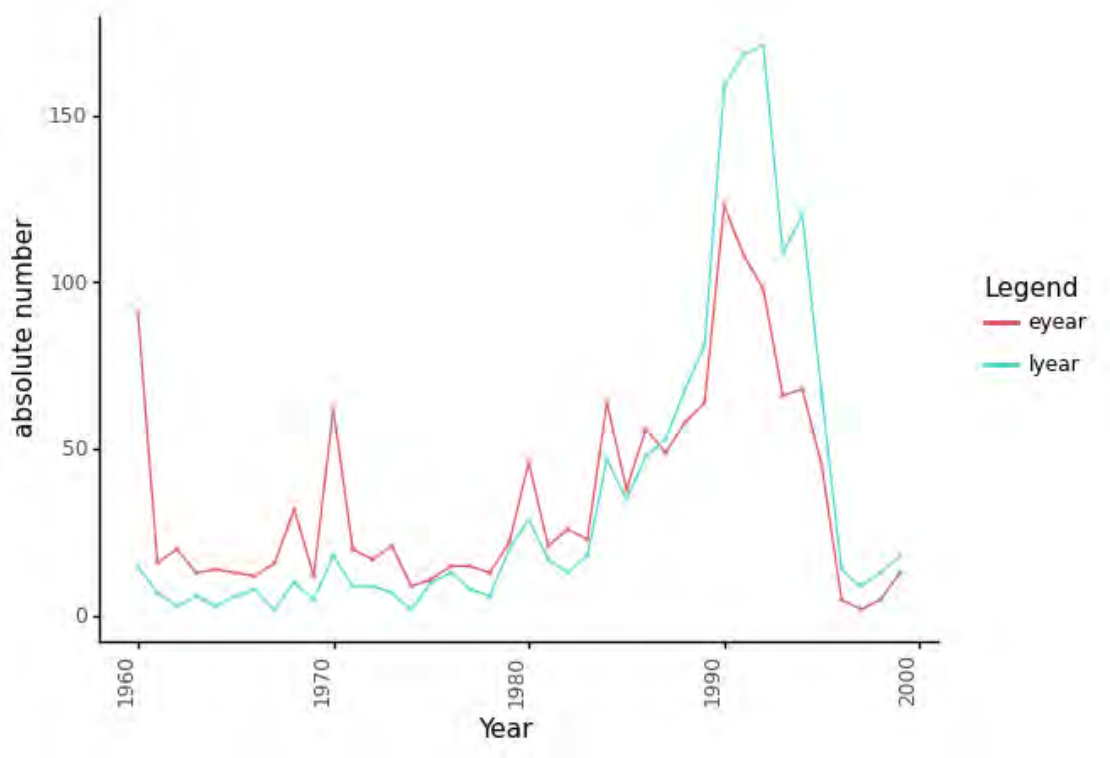


Abbildung 15: Dokumente pro Jahr: Gesamtdatensatz ohne Kategorie «sex»

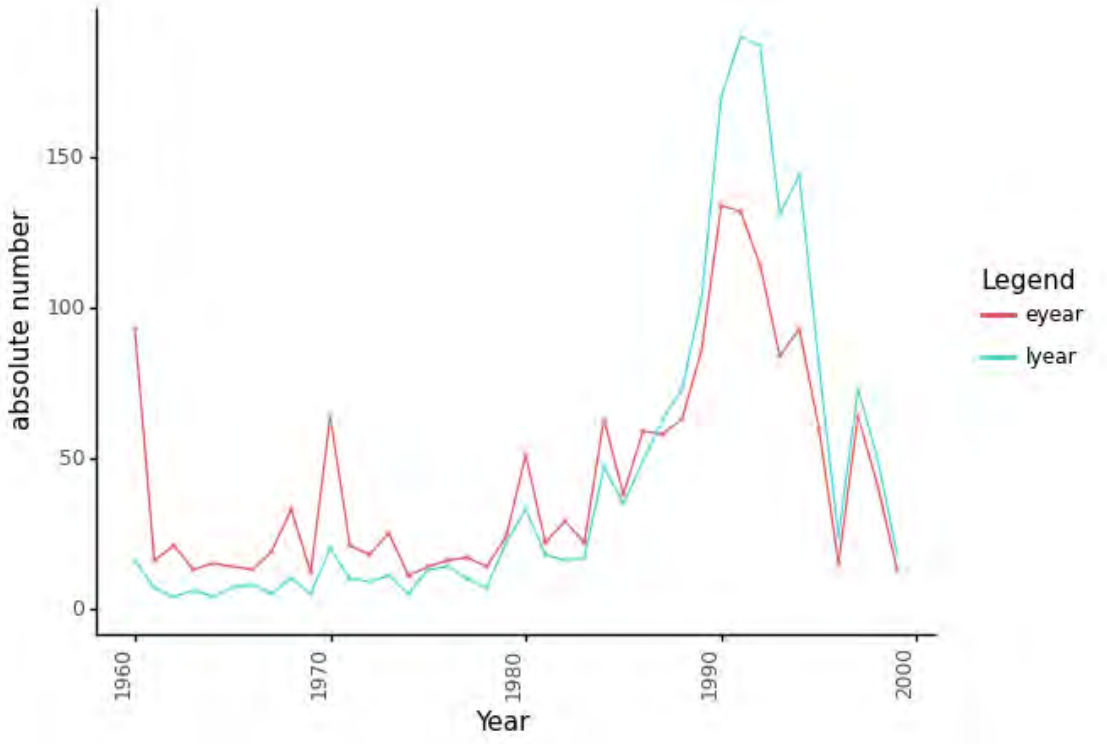


Abbildung 16: Dokumente pro Jahr: Gesamtdatensatz