Entscheidungen des Bundesgerichtshofs in Strafsachen aus dem 20. Jahrhundert

(BGH-Strafsachen-20Jhd-Source)

Compilation Report

Version 1.0.0

DOI: 10.5281/zenodo.7847575

Titel	Source Code des »Entscheidungen des Bundesgerichtshofs in Strafsachen aus dem 20. Jahrhundert«
Abkürzung	BGH-Strafsachen-20Jhd-Source
Autor	Seán Fobbe und Tilko Swalve
Version	1.0.0
Download	https://doi.org/10.5281/zenodo.7847575
Lizenz	GNU General Public License Version 3 (GPLv3)

Zitiervorschlag

Seán Fobbe und Tilko Swalve (2024). Source Code des »Entscheidungen des Bundesgerichtshofs in Strafsachen aus dem 20. Jahrhundert« (BGH-Strafsachen-20Jhd-Source). Version 1.0.0. Zenodo. DOI: 10.5281/zenodo.7847575.

Digital Object Identifier (DOI): Concept DOI und Version DOI

Soweit nicht anders angegeben ist die DOI immer eine »Version DOI« und bezieht sich nur auf eine bestimmte Version der Software. Sie verlinkt daher nur Version 1.0.0. Für das Gesamtkonzept der Software steht eine »Concept DOI« zur Verfügung, die auf der Zenodo-Seite jeder Version unter »Cite all versions?« zu finden ist. Die »Concept DOI« verlinkt immer die aktuellste Version.

GNU General Public License Version 3 (GPLv3)

Disclaimer

Dieser Datensatz ist eine private wissenschaftliche Initiative und steht in keiner Verbindung zu Behörden, Gerichten oder anderen amtlichen Stellen der Bundesrepublik Deutschland.

Inhaltsverzeichnis

1 Entscheidungen des Bundesgerichtshofs in Strafsachen aus dem 20. Jahrhundert (BGH-Strafsachen-20Jhd) 6
1.1 Überblick . . . 6
1.2 Features . . . 6
1.3 Ergebnisse . . . 6
1.4 Systemanforderungen . . . 6
1.5 Anleitung . . . 7
1.5.1 Schritt 1: Ordner vorbereiten . . . 7
1.5.2 Schritt 2: Docker Image erstellen . . . 7
1.5.3 Schritt 3: Datensatz kompilieren . . . 7
1.5.4 Ergebnis . . . 7
1.6 Pipeline visualisieren . . . 8
1.7 Troubleshooting . . . 8
1.8 Projektstruktur . . . 8
1.9 Persönliche Webseiten der Autor:innen . . . 8
1.10 Kontakt . . . 9

2 Packages laden 10

3 Vorbereitung 11
3.1 Definitionen . . . 11
3.2 Aufräumen . . . 12
3.3 Ordner erstellen . . . 12
3.4 Vollzitate statistischer Software schreiben . . . 12

4 Globale Variablen 13
4.1 Packages definieren . . . 13
4.2 Konfiguration . . . 13
4.3 Funktionen definieren . . . 14
4.4 Metadaten für TXT-Dateien definieren . . . 14
4.5 ZIP-Datei für Source definieren . . . 14

5 Pipeline: Konstruktion 15
5.1 File Tracking Targets . . . 15
5.1.1 Source Code . . . 15
5.1.2 Changelog . . . 15
5.1.3 Datenmodell . . . 15
5.1.4 Aktenzeichen der Bundesrepublik Deutschland (AZ-BRD) . . . 15
5.1.5 Presidents and Vice-Presidents of the Federal Courts of Germany (PVP-FCG) . . . 16
5.1.6 Problematische OCR-Dateien . . . 16
5.1.7 Tabelle mit OCR-Korrekturen . . . 17
5.2 Vorbereitung des Korpus . . . 17
5.2.1 Entpacken und Standardisieren von Dateinamen . . . 17
5.2.2 Testen auf Querformat . . . 17
5.2.3 Querformat zu Hochformat rotieren . . . 17
5.2.4 Originale und rotierte PDF-Dateien vereinigen . . . 18
5.3 Konvertieren . . . 18

5.3.1 Problematische PDF-Dateien entfernen . . . 18
5.3.2 Optical Character Recognition (OCR) . . . 18
5.3.3 PDF Text Layer extrahieren . . . 18
5.4 Enhance . . . 19
5.4.1 OCR-Fehler im Text bereinigen . . . 19
5.4.2 Variable erstellen: »datum« . . . 19
5.4.3 Variable erstellen: »verfahrensart« . . . 20
5.4.4 Variable erstellen: »aktenzeichen« . . . 20
5.4.5 Variable erstellen: »praesi« . . . 20
5.4.6 Variable erstellen: »vpraesi« . . . 20
5.4.7 Variablen erstellen: »bghst, bghz, bghr, nachschlagewerk« . . . 20
5.4.8 Variablen erstellen: »zeichen, token, typen, saetze« . . . 21
5.4.9 Konstanten erstellen . . . 21
5.4.10 Zusätzliche Variablen zusammenführen . . . 21
5.4.11 Finalen Datensatz erstellen . . . 21
5.4.12 Variante erstellen: Nur Metadaten . . . 22
5.5 Write Targets . . . 22
5.5.1 CSV schreiben: Voller Datensatz . . . 22
5.5.2 CSV schreiben: Metadaten . . . 22
5.5.3 TXT schreiben: Voller Datensatz (reduziert) . . . 22
5.6 Report Targets . . . 23
5.6.1 LaTeX-Definitionen schreiben . . . 23
5.6.2 Zusammenfassungen linguistischer Kennwerte berechnen . . . 23
5.6.3 Report erstellen: Quality Control . . . 23
5.6.4 Report erstellen: Codebook . . . 24
5.7 ZIP Targets . . . 24
5.7.1 ZIP erstellen: Analyse-Dateien . . . 24
5.7.2 ZIP erstellen: Source Code . . . 24
5.7.3 ZIP erstellen: CSV-Datei (voller Datensatz) . . . 24
5.7.4 ZIP erstellen: CSV-Datei (nur Metadaten) . . . 25
5.7.5 ZIP erstellen: TXT-Dateien . . . 25
5.7.6 ZIPs erstellen: PDF-Dateien . . . 25
5.7.7 ZIPs erstellen: Problematische PDF-Dateien . . . 25
5.8 Kryptographische Hashes . . . 26
5.8.1 Zu hashende ZIP-Archive definieren . . . 26
5.8.2 Kryptographische Hashes berechnen . . . 26
5.8.3 CSV schreiben: Kryptographische Hashes . . . 26

6 Pipeline: Kompilierung **27**
6.1 Durchführen der Kompilierung . . . 27
6.2 Pipeline archivieren . . . 27
6.3 Visualisierung . . . 27

7 Liste aller Targets (alphabetisch) **29**

8 Gesamte Laufzeit **32**

9 Laufzeit einzelner Targets **33**

10 Warnungen **36**

10.1 dt.ocr . . . 36
10.2 report.codebook . . . 36
10.3 report.quality . . . 36

11 Fehlermeldungen **37**

12 Dateigrößen **38**
12.1 ZIP-Dateien . . . 38
12.2 CSV-Dateien . . . 39
12.3 PDF-Dateien (MB) . . . 39
12.4 TXT-Dateien (MB) . . . 39

13 Kryptographische Signaturen **40**
13.1 Signaturen laden . . . 40
13.2 Leerzeichen hinzufügen um bei SHA3-512 Zeilenumbruch zu ermöglichen . . . 40
13.3 In Bericht anzeigen . . . 40

14 Changelog **43**
14.1 Version 1.0.0 . . . 43

15 Abschluss **44**

16 Parameter für strenge Replikationen **45**

Literaturverzeichnis **47**

1 Entscheidungen des Bundesgerichtshofs in Strafsachen aus dem 20. Jahrhundert (BGH-Strafsachen-20Jhd)

1.1 Überblick

Der Datensatz **Entscheidungen des Bundesgerichtshofs in Strafsachen aus dem 20. Jahrhundert** (BGH-Strafsachen-20Jhd)** ist eine möglichst vollständige Sammlung der durch den Bundesgerichtshof in Strafsachen getroffenen Entscheidungen vom 1. Oktober 1950 (Gründung des BGH) bis zum 1. Januar 2000, dem Zeitpunkt ab dem der BGH digitale Entscheidungen regulär veröffentlicht.

Der Datensatz nutzt als seine Datenquelle eine vom Bundesgerichtshof den Autoren übergebene digitale Sammlung dieser Entscheidungen und bereitet diese wissenschaftlich auf.

Alle mit diesem Skript erstellten Datensätze werden dauerhaft kostenlos und urheberrechtsfrei auf Zenodo, dem wissenschaftlichen Archiv des CERN, veröffentlicht. Alle Versionen sind mit einem separaten und langzeit-stabilen (persistenten) Digital Object Identifier (DOI) versehen.

Aktuellster, funktionaler und zitierfähiger Release des Datensatzes: https://doi.org/10.5281/zenodo.4540377

1.2 Features

- Bereinigung der Dateinamen
- Korrektur falscher Rotationen, Standardisierung im Hochformat
- Optische Zeichenerkennung (OCR)
- Automatisierte Bereinigung von OCR-Fehlern mit Ersetzungstabelle
- Extraktion zusätzlicher Variablen
- Erstellung nutzungsfertiger ZIP-Archive
- Umfangreiche Dokumentation
- Automatisierte Unit Tests und statistisches Reporting
- Kryptographische Signaturen

1.3 Ergebnisse

Primäre Endprodukte des Skripts sind folgende ZIP-Archive:

- Der volle Datensatz im CSV-Format (mit zusätzlichen Metadaten)
- Die reinen Metadaten im CSV-Format (wie unter 1, nur ohne Entscheidungsinhalte)
- Alle Entscheidungen im TXT-Format
- Alle Entscheidungen im PDF-Format
- Alle Analyse-Ergebnisse (Tabellen als CSV, Grafiken als PDF und PNG)

Alle Ergebnisse werden im Ordner `output` abgelegt. Zusätzlich werden für alle ZIP-Archive kryptographische Signaturen (SHA2-256 und SHA3-512) berechnet und in einer CSV-Datei hinterlegt.

1.4 Systemanforderungen

- Docker

- Docker Compose
- 34 GB Speicherplatz auf Festplatte
- 32 GB Arbeitsspeicher (RAM)
- Multi-core CPU empfohlen (8 cores/16 threads für die Referenzdatensätze).

In der Standard-Einstellung wird das Skript vollautomatisch die maximale Anzahl an Rechenkernen/Threads auf dem System zu nutzen. Die Anzahl der verwendeten Kerne kann in der Konfigurationsatei angepasst werden. Wenn die Anzahl Threads auf 1 gesetzt wird, ist die Parallelisierung deaktiviert.

1.5 Anleitung

1.5.1 Schritt 1: Ordner vorbereiten

Kopieren Sie bitte den gesamten Source Code in einen leeren Ordner (!), beispielsweise mit:

```
$ git clone https://github.com/seanfobbe/bgh-strafrecht
```

Verwenden Sie immer einen separaten und *leeren* Ordner für die Kompilierung. Die Skripte löschen innerhalb von bestimmten Unterordnern (`files/`, `temp/`, `analysis` und `output/`) alle Dateien die den Datensatz verunreinigen könnten — aber auch nur dort.

1.5.2 Schritt 2: Docker Image erstellen

Ein Docker Image stellt ein komplettes Betriebssystem mit der gesamten verwendeten Software automatisch zusammen. Nutzen Sie zur Erstellung des Images einfach:

```
$ bash docker-build-image.sh
```

1.5.3 Schritt 3: Datensatz kompilieren

Falls Sie zuvor den Datensatz schon einmal kompiliert haben (ob erfolgreich oder erfolglos), können Sie mit folgendem Befehl alle Arbeitsdaten im Ordner löschen:

```
$ Rscript delete_all_data.R
```

Den vollständigen Datensatz kompilieren Sie mit folgendem Skript:

```
$ bash docker-run-project.sh
```

1.5.4 Ergebnis

Der Datensatz und alle weiteren Ergebnisse sind nun im Ordner `output/` abgelegt.

1.6 Pipeline visualisieren

Sie können die Pipeline visualisieren, aber nur nachdem sie die zentrale .Rmd-Datei mindestens einmal gerendert haben:

```
> targets::tar_glimpse()     # Nur Datenobjekte
> targets::tar_visnetwork()  # Alle Objekte
```

1.7 Troubleshooting

Hilfreiche Befehle um Fehler zu lokalisieren und zu beheben.

```
> tar_progress()  # Zeigt Fortschritt und Fehler an
> tar_meta()      # Alle Metadaten
> tar_meta(fields = "warnings", complete_only = TRUE)  # Warnungen
> tar_meta(fields = "error", complete_only = TRUE)  # Fehlermeldungen
> tar_meta(fields = "seconds")  # Laufzeit der Targets
```

1.8 Projektstruktur

Die folgende Struktur erläutert die wichtigsten Bestandteile des Projekts. Während der Kompilierung werden weitere Ordner erstellt (`files/`, `temp/` `analysis` und `output/`). Die Endergebnisse werden alle in `output/` abgelegt.

```
.
├ buttons                    # Buttons (nur optische Bedeutung)
├ CHANGELOG.md               # Alle Änderungen
├ config.toml                # Zentrale Konfigurations-Datei
├ data                       # Datensätze, auf denen die Pipeline aufbaut
├ delete_all_data.R          # Löscht den Datensatz und Zwischenschritte
├ docker-build-image.sh      # Docker Image erstellen
├ docker-compose.yaml        # Konfiguration für Docker
├ docker-delete-all-data.sh  # Löschen aller Datein aus Docker heraus
├ Dockerfile                 # Definition des Docker Images
├ docker-run-project.sh      # Docker Image und Datensatz kompilieren
├ etc                        # Zusätzliche Konfigurationsdateien
├ functions                  # Wichtige Schritte der Pipeline
├ gpg                        # Persönlicher Public GPG-Key für Seán Fobbe
├ pipeline.Rmd               # Zentrale Definition der Pipeline
├ README.md                  # Bedienungsanleitung
├ reports                    # Markdown-Dateien
├ run_project.R              # Kompiliert den gesamten Datensatz
└ tex                       # LaTeX-Templates
```

1.9 Persönliche Webseiten der Autor:innen

Seán Fobbe — https://www.seanfobbe.de

Tilko Swalve — https://tilkoswalve.netlify.app/

1.10 Kontakt

Fehler gefunden? Anregungen? Kommentieren Sie gerne im Issue Tracker auf GitHub oder kontaktieren sie mich via https://www.seanfobbe.de

2 Packages laden

```
library(targets)
library(tarchetypes)
library(RcppTOML)
library(future)
library(data.table)
library(quanteda)
library(knitr)
library(kableExtra)
library(igraph)
library(ggraph)

tar_unscript()
```

3 Vorbereitung

3.1 Definitionen

```
## Datum
datestamp <- Sys.Date()
print(datestamp)
#> [1] "2024-11-25"

## Datum und Uhrzeit (Beginn)
begin.script <- Sys.time()

## Konfiguration
config <- RcppTOML::parseTOML("config.toml")
print(config)
#> List of 10
#>  $ cores   :List of 3
#>   ..$ max     : logi FALSE
#>   ..$ number  : int 15
#>   ..$ tessjobs: int 5
#>  $ debug   :List of 3
#>   ..$ cleanrun       : logi FALSE
#>   ..$ skipzip        : logi TRUE
#>   ..$ tesseractresume: logi TRUE
#>  $ doi     :List of 4
#>   ..$ aktenzeichen : chr "10.5281/zenodo.4569564"
#>   ..$ data         :List of 2
#>   .. ..$ concept: chr "10.5281/zenodo.4540376"
#>   .. ..$ version: chr "10.5281/zenodo.4540377"
#>   ..$ personendaten: chr "10.5281/zenodo.4568682"
#>   ..$ software     :List of 2
#>   .. ..$ concept: chr "10.5281/zenodo.7847574"
#>   .. ..$ version: chr "10.5281/zenodo.7847575"
#>  $ download:List of 1
#>   ..$ timeout: int 600
#>  $ fig     :List of 3
#>   ..$ align : chr "center"
#>   ..$ dpi   : int 300
#>   ..$ format: chr [1:2] "pdf" "png"
#>  $ license :List of 2
#>   ..$ code: chr "GNU General Public License Version 3 (GPLv3)"
#>   ..$ data: chr "Creative Commons Zero 1.0 Universal"
#>  $ parallel:List of 3
#>   ..$ extractPDF   : logi TRUE
#>   ..$ lingsummarize: logi TRUE
#>   ..$ multihashes  : logi TRUE
#>  $ project :List of 3
#>   ..$ author   : chr "Seán Fobbe und Tilko Swalve"
#>   ..$ fullname : chr "Entscheidungen des Bundesgerichtshofs in Strafsachen aus
    dem 20. Jahrhundert"
#>   ..$ shortname: chr "BGH-Strafsachen-20Jhd"
#>  $ quanteda:List of 1
#>   ..$ tokens_locale: chr "de_DE"
#>  $ version :List of 2
```

```
#>   ..$ dash    : chr "1-0-0"
#>   ..$ semantic: chr "1.0.0"


# Analyse-Ordner
dir.analysis <- paste0(getwd(),
                       "/analysis")
```

3.2 Aufräumen

Löscht nicht mehr aktuelle Dateien im Output-Ordner.

```
unlink("temp")
##unlink("output", recursive = TRUE)


if(config$debug$tesseractresume == FALSE){

    unlink("temp_tesseract")

    }
```

3.3 Ordner erstellen

```
dirs <- c("output",
          "temp",
          "files")

lapply(dirs, dir.create, showWarnings = FALSE, recursive = TRUE)
#> [[1]]
#> [1] FALSE
#>
#> [[2]]
#> [1] FALSE
#>
#> [[3]]
#> [1] FALSE

dir.create(dir.analysis, showWarnings = FALSE)
```

3.4 Vollzitate statistischer Software schreiben

```
knitr::write_bib(renv::dependencies()$Package,
                 "temp/packages.bib")
#> Finding R package dependencies ... Done!
```

4 Globale Variablen

4.1 Packages definieren

```
tar_option_set(packages = c("tarchetypes",  # Zusätzliche Targets Funktionen
                            "RcppTOML",     # TOML-Dateien lesen und schreiben
                            "fs",           # Verbessertes File Handling
                            "zip",          # Verbessertes ZIP Handling
                            "testthat",     # Unit tests
                            "mgsub",        # Vektorisiertes Gsub
                            "httr",         # HTTP-Werkzeuge
                            "rvest",        # HTML/XML-Extraktion
                            "knitr",        # Professionelles Reporting
                            "kableExtra",   # Verbesserte Kable Tabellen
                            "pdftools",     # Verarbeitung von PDF-Dateien
                            "ggplot2",      # Fortgeschrittene
    Datenvisualisierung
                            "ggraph",       # Visualisierung von Graphen
                            "scales",       # Skalierung von Diagrammen
                            "magick",       # Image processing
                            "readtext",     # TXT-Dateien einlesen
                            "quanteda",     # Fortgeschrittene Computerlinguistik
                            "future",       # Parallelisierung
                            "future.apply", # Funktionen höherer Ordnung für
    Parallelisierung
                            "data.table"))   # Fortgeschrittene Datenverarbeitung

tar_option_set(workspace_on_error = TRUE) # Save Workspace on Error
tar_option_set(format = "qs")

#> Establish _targets.R and _targets_r/globals/global-packages.R.
```

4.2 Konfiguration

```
datestamp <- Sys.Date()

config <- RcppTOML::parseTOML("config.toml")

dir.analysis <- paste0(getwd(),
                       "/analysis")

## Caption for diagrams
caption <- paste("Fobbe/Swalve | DOI:",
                 config$doi$data$version)


## Prefix for figure titles
prefix.figuretitle <- paste(config$project$shortname,
                            "| Version",
                            config$version$semantic)
```

```
## File prefix
prefix.files <- paste0(config$project$shortname,
                       "_",
                       config$version$dash)


if (config$cores$max == TRUE){
    fullCores <- parallel::detectCores()
    tessJobs <- round(fullCores / 3)
}


if (config$cores$max == FALSE){
    fullCores <- as.integer(config$cores$number)
    tessJobs <- as.integer(config$cores$tessjobs)
}

#> Establish _targets.R and _targets_r/globals/global-config.R.
```

4.3 Funktionen definieren

```
lapply(list.files("functions", pattern = "\\.R$", full.names = TRUE), source)

#> Establish _targets.R and _targets_r/globals/global-functions.R.
```

4.4 Metadaten für TXT-Dateien definieren

```
docvarnames <- c("spruchkoerper_az",
                 "registerzeichen",
                 "eingangsnummer",
                 "eingangsjahr_az",
                 "zusatz_az",
                 "name",
                 "kollision")

#> Establish _targets.R and _targets_r/globals/global-txtvars.R.
```

4.5 ZIP-Datei für Source definieren

```
files.source.raw <-  c(system2("git", "ls-files", stdout = TRUE), ".git")

#> Establish _targets.R and _targets_r/globals/global-sourcefiles.R.
```

5 Pipeline: Konstruktion

5.1 File Tracking Targets

Mit diesem Abschnitt der Pipeline werden Input-Dateien getrackt. Mit der Option »format = "file"« werden für Input-Dateien Prüfsummen berechnet. Falls sich diese verändern werden alle von ihnen abhängigen Pipeline-Schritte als veraltet markiert und neu berechnet.

5.1.1 Source Code

Dies sind alle Dateien, die den Source Code für den Datensatz bilden.

```
tar_target(files.source,
           files.source.raw,
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.file.source.R.
```

5.1.2 Changelog

```
tar_target(changelog,
           "CHANGELOG.md",
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.file.changelog.R.
```

5.1.3 Datenmodell

Dieses Target liest das maschinenlesbare Datenmodell für den Datensatz ein.

```
list(
    tar_target(file.datamodel,
               "data/BGH-Strafrecht_Variables.csv",
               format = "file"),
    tar_target(datamodel,
               fread(file.datamodel))
)
#> Establish _targets.R and _targets_r/targets/tar.file.datamodel.R.
```

5.1.4 Aktenzeichen der Bundesrepublik Deutschland (AZ-BRD)

Die Tabelle der Registerzeichen und der ihnen zugeordneten Verfahrensarten stammt aus dem folgenden Datensatz: »Seán Fobbe (2021). Aktenzeichen der Bundesrepublik Deutschland (AZ-BRD). Version 1.0.1. Zenodo. DOI: 10.5281/zenodo.4569564.«

```
list(
    tar_target(file.az.brd,
           "data/AZ-BRD_1-0-1_DE_Registerzeichen_Datensatz.csv",
```

```
            format = "file"),
     tar_target(az.brd,
                fread(file.az.brd))
     )
#> Establish _targets.R and _targets_r/targets/tar.file.az.R.
```

5.1.5 Presidents and Vice-Presidents of the Federal Courts of Germany (PVP-FCG)

Die Personendaten stammen aus folgendem Datensatz: »Seán Fobbe and Tilko Swalve (2021). Presidents and Vice-Presidents of the Federal Courts of Germany (PVP-FCG). Version 2021-04-08. Zenodo. DOI: 10.5281/zenodo.4568682«.

```
list(
    tar_target(file.presidents,
           "data/PVP-FCG_2021-04-08_GermanFederalCourts_Presidents.csv",
           format = "file"),
     tar_target(presidents,
                fread(file.presidents))
     )
#> Establish _targets.R and _targets_r/targets/tar.file.presi.R.
```

```
list(
    tar_target(file.vpresidents,
           "data/PVP-FCG_2021-04-08_GermanFederalCourts_VicePresidents.csv",
           format = "file"),
     tar_target(vpresidents,
                fread(file.vpresidents))
     )
#> Establish _targets.R and _targets_r/targets/tar.file.vpresi.R.
```

5.1.6 Problematische OCR-Dateien

Diese Dateien verursachen seltsames Verhalten in der OCR-Pipeline. In der Regel reichen 10-15 GB Arbeitsspeicher aus um 5-6 Tesseract Jobs parallel laufen zu lassen, diese Dateien verbrauchen über 64GB Arbeitsspeicher selbst bei nur sequentiellen Tesseract Jobs.

```
list(
    tar_target(file.ocrprob,
               "data/ocr-problem-cases.txt",
               format = "file"),
    tar_target(pdf.ocrprob,
               readLines(file.ocrprob))
)
#> Establish _targets.R and _targets_r/targets/tar.file.ocrprob.R.
```

5.1.7 Tabelle mit OCR-Korrekturen

Diese Tabelle enthält Korrekturen für OCR-Fehler.

```
list(
    tar_target(file.replacements,
           "data/BGH-Strafrecht_ReplacementTable.csv",
           format = "file"),
     tar_target(replacements,
                fread(file.replacements))
     )
#> Establish _targets.R and _targets_r/targets/tar.file.replacements.R.
```

5.2 Vorbereitung des Korpus

5.2.1 Entpacken und Standardisieren von Dateinamen

```
tar_target(pdf.original,
           f.unzip_rename(dir.in = "zip_original",
                          dir.out = "files/pdf_original"),
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.unzip.rename.R.
```

5.2.2 Testen auf Querformat

```
tar_target(dt.landscape,
           f.landscape(x = pdf.original))

#> Establish _targets.R and _targets_r/targets/tar.landscape.R.
```

5.2.3 Querformat zu Hochformat rotieren

```
tar_target(pdf.rotated,
           f.rotate(dt.landscape$x[dt.landscape$landscape],
                    files.opposite = c("2_StR_481_84_NA_NA_NA.pdf",
                                       "4_StR_131_60_NA_NA_NA.pdf",
                                       "4_StR_190_70_NA_NA_NA.pdf",
                                       "4_StR_512_89_NA_NA_NA.pdf"),
                    angle = -90,
                    dir.output = "files/pdf_rotated",
                    clean = TRUE))

#> Establish _targets.R and _targets_r/targets/tar.rotate.R.
```

5.2.4 Originale und rotierte PDF-Dateien vereinigen

```
tar_target(pdf.cleaned,
           c(pdf.original[basename(pdf.original) %notin% basename(pdf.rotated)],
             pdf.rotated))

#> Establish _targets.R and _targets_r/targets/tar.original.rotate.unite.R.
```

5.3 Konvertieren

5.3.1 Problematische PDF-Dateien entfernen

```
tar_target(pdf.cleaned.noprob,
           pdf.cleaned[basename(pdf.cleaned) %notin% basename(pdf.ocrprob)])

#> Establish _targets.R and _targets_r/targets/tar.convert.removeprob.R.
```

5.3.2 Optical Character Recognition (OCR)

```
list(tar_target(txt.ocr,
                f.tar_pdf_ocr(pdf.cleaned.noprob,
                              dpi = 300,
                              lang = "deu",
                              output = "txt",
                              resume = config$debug$tesseractresume,
                              crop.firstpage = 0,
                              crop.lastpage = 0,
                              dir.out.pdf = "files/pdf_tesseract",
                              dir.out.txt = "files/txt_tesseract",
                              tempfile = TRUE,
                              chunksperworker = 1,
                              chunksize = 1,
                              quiet = TRUE,
                              jobs = tessJobs),
                format = "file"),
     tar_target(dt.ocr,
                f.readtext(x = txt.ocr,
                           docvarnames = docvarnames))
     )

#> Establish _targets.R and _targets_r/targets/tar.convert.tesseract.R.
```

5.3.3 PDF Text Layer extrahieren

```
list(tar_target(txt.extracted,
                f.tar_pdf_extract(x = pdf.original,
                                  outputdir = "files/txt_extracted",
```

```
                                    multicore = config$parallel$extractPDF,
                                    cores = fullCores),
                format = "file"),
    tar_target(dt.extracted,
                f.readtext(x = txt.extracted,
                           docvarnames = docvarnames))
    )

#> Establish _targets.R and _targets_r/targets/tar.convert.pdfextract.R.
```

5.4 Enhance

5.4.1 OCR-Fehler im Text bereinigen

Dieser Schritt bereinigt einige häufige OCR-Fehler. Der Fokus liegt hierbei auf juristischen Fachbegriffen und nicht auf dem allgemeinen Wortschatz. Die Liste erhebt keinen Anspruch auf Vollständigkeit, sollteaber künftige NLP-Anwendungen etwas verbessern, insbesondere Netzwerkanalysen, die auf die Erkennung von Gesetzesbezeichnungen abstellen.

Die Auswahl der Korrekturen erfolgte durch das genaue Lesen einzelner mit `ls | shuf -n 1` zufällig ausgewählter Entscheidungen und der Notierung entsprechender Korrekturen in einer CSV-Tabelle im Source Code.

```
tar_target(var_text,
           f.clean_text(x = dt.ocr$text,
                        replacements = replacements))

#> Establish _targets.R and _targets_r/targets/tar.enhance.cleantext.R.
```

5.4.2 Variable erstellen: »datum«

Dieser Schritt versucht das Datum der Entscheidung aus dem Text zu extrahieren. Erkannt werden Daten ähnlich folgendem Beispiel “am/vom 13. September 1966”. Die längeren Zeichenfolgen “Beschluss/Urteil/Sitzung vom” ergeben zuviele falsch-negative Ergebnisse, da beim Auslesen von PDFs oft das Aktenzeichen zwischen “Beschluss” und dem Datum landet.

Das Limit legt fest bis zu wievielen Zeichen vom Anfang des Textes an gesucht werden soll. Eine vernünftige Einstellung beugt falsch-positiven Ergebnissen vor und beschleunigt die Extraktion

```
tar_target(var_date,
           f.var_date(x = dt.ocr$text,
                      limit = 2000,
                      date.min = "1950-10-1",
                      date.max = "2000-01-01"))

#> Establish _targets.R and _targets_r/targets/tar.enhance.date.R.
```

5.4.3 Variable erstellen: »verfahrensart«

Die Variable »verfahrensart« wird aus den Registerzeichen berechnet.

```
tar_target(var_verfahrensart,
           f.var_verfahrensart(dt.ocr$registerzeichen,
                               az.brd = az.brd,
                               gericht = "BGH"))
#> Establish _targets.R and _targets_r/targets/tar.enhance.verfahrensart.R.
```

5.4.4 Variable erstellen: »aktenzeichen«

Das Aktenzeichen wird aus seinen Komponenten berechnet.

```
tar_target(var_aktenzeichen,
           f.var_aktenzeichen(x = dt.ocr,
                              az.brd = az.brd,
                              gericht = "BGH"))
#> Establish _targets.R and _targets_r/targets/tar.enhance.az.R.
```

5.4.5 Variable erstellen: »praesi«

```
tar_target(var_praesi,
           f.presidents(datum = var_date,
                        gericht = "BGH",
                        pvp.fcg = presidents))
#> Establish _targets.R and _targets_r/targets/tar.enhance.praesi.R.
```

5.4.6 Variable erstellen: »vpraesi«

```
tar_target(var_vpraesi,
           f.presidents(datum = var_date,
                        gericht = "BGH",
                        pvp.fcg = vpresidents))
#> Establish _targets.R and _targets_r/targets/tar.enhance.vpraesi.R.
```

5.4.7 Variablen erstellen: »bghst, bghz, bghr, nachschlagewerk«

```
tar_target(var_sammlungen,
           f.var_sammlungen(var_text))

#> Establish _targets.R and _targets_r/targets/tar.enhance.sammlungen.R.
```

5.4.8 Variablen erstellen: »zeichen, token, typen, saetze«

Berechnung klassischer linguistischer Kennzahlen.

```
tar_target(var_lingstats,
                f.lingstats(data.table(doc_id = 1:length(var_text), text = var_
    text),
                            multicore = config$parallel$lingsummarize,
                            cores = fullCores,
                            germanvars = TRUE))
#> Establish _targets.R and _targets_r/targets/tar.enhance.lingstats.R.
```

5.4.9 Konstanten erstellen

Konstanten die dem Datensatz wichtige Herkunftsinformationen hinzufügen. Darunter sind die Versionsnummer, die Version DOI, die Concept DOI und die Lizenz.

```
tar_target(var_constants,
           data.frame(version = config$version$semantic,
                      doi_concept = config$doi$data$concept,
                      doi_version = config$doi$data$version,
                      lizenz = as.character(config$license$data))[rep(1,
                                                                      nrow(dt.ocr
    )),])
#> Establish _targets.R and _targets_r/targets/tar.enhance.constants.R.
```

5.4.10 Zusätzliche Variablen zusammenführen

```
tar_target(vars_additional,
           data.table(datum = var_date,
                      text = var_text,
                      praesi = var_praesi,
                      v_praesi = var_vpraesi,
                      verfahrensart = var_verfahrensart,
                      aktenzeichen = var_aktenzeichen,
                      var_lingstats,
                      var_sammlungen,
                      var_constants))

#> Establish _targets.R and _targets_r/targets/tar.enhance.unify.R.
```

5.4.11 Finalen Datensatz erstellen

Die Verbesserungen der vorherigen Schritte werden in dieser Funktion zusammengefügt um den finalen Datenatz herzustellen.

```
tar_target(dt.final,
           f.finalize(x = dt.ocr,
```

```
                        vars.additional = vars_additional,
                        varnames = datamodel$varname))

#> Establish _targets.R and _targets_r/targets/tar.enhance.final.R.
```

5.4.12 Variante erstellen: Nur Metadaten

Hier wird die Text-Variable entfernt, um eine deutlich platzsparendere Variante des Datensatzes zu erstellen. Enthalten sind nur noch die Metadaten.

```
tar_target(dt.meta,
           dt.final[, !c("text", "text_raw")])

#> Establish _targets.R and _targets_r/targets/tar.enhance.meta.R.
```

5.5 Write Targets

Dieser Abschnitt der Pipeline schreibt den Datensatz und alle Hash-Prüfsummen auf die Festplatte.

5.5.1 CSV schreiben: Voller Datensatz

```
tar_target(csv.final,
           f.tar_fwrite(x = dt.final,
                        filename = file.path("output",
                                             paste0(prefix.files,
                                                    "_DE_CSV_Datensatz.csv"))
                        )
           )
#> Establish _targets.R and _targets_r/targets/tar.write.final.R.
```

5.5.2 CSV schreiben: Metadaten

```
tar_target(csv.meta,
           f.tar_fwrite(x = dt.meta,
                        filename = file.path("output",
                                             paste0(prefix.files,
                                                    "_DE_CSV_Metadaten.csv"))
                        )
           )
#> Establish _targets.R and _targets_r/targets/tar.write.meta.R.
```

5.5.3 TXT schreiben: Voller Datensatz (reduziert)

```
tar_target(txt_best,
           f.tar_write_txt(text = dt.final$text,
```

```
                            doc_id = dt.final$doc_id,
                            dir = "files/txt_best",
                            cleandir = TRUE),
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.write.txt.best.R.
```

5.6 Report Targets

Dieser Abschnitt der Pipeline erstellt die finalen Berichte (Codebook und Robustness Checks).

5.6.1 LaTeX-Definitionen schreiben

Um gewisse Variablen aus der Pipeline in die LaTeX-Kompilierung einzuführen müssen diese als .tex-Datei auf die Festplatte geschrieben werden.

```
tar_target(latexdefs,
                f.latexdefs(config,
                            dir = "temp",
                            version = config$version$semantic),
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.report.latexdefs.R.
```

5.6.2 Zusammenfassungen linguistischer Kennwerte berechnen

```
tar_target(lingstats.summary,
                f.lingstats_summary(dt.final,
                                    germanvars = TRUE))

#> Establish _targets.R and _targets_r/targets/tar.report.lingstat.summ.R.
```

5.6.3 Report erstellen: Quality Control

```
tarchetypes::tar_render(report.quality,
                        file.path("reports",
                                  "quality.Rmd"),
                        output_file = file.path("../output",
                                                paste0(config$project$shortname,
                                                       "_",
                                                       config$version$dash,
                                                       "_QualityControlReport.pdf
    ")))

#> Establish _targets.R and _targets_r/targets/tar.report.quality.R.
```

5.6.4 Report erstellen: Codebook

```
tarchetypes::tar_render(report.codebook,
                        file.path("reports",
                                  "codebook.Rmd"),
                        output_file = file.path("../output",
                                                paste0(config$project$shortname,
                                                       "_",
                                                       config$version$dash,
                                                       "_Codebook.pdf")))

#> Establish _targets.R and _targets_r/targets/tar.report.codebook.R.
```

5.7 ZIP Targets

Diese Abschnitt der Pipeline erstellt ZIP-Archive für alle zentralen Rechenergebnisse und speichert diese im Ordner »output«.

5.7.1 ZIP erstellen: Analyse-Dateien

```
tar_target(zip.analysis,
           f.tar_zip("analysis/",
                     filename = paste(prefix.files,
                                      "DE_Analyse.zip",
                                      sep = "_"),
                     dir = "output",
                     mode = "cherry-pick",
                     report.codebook,    # manually enforced dependency
    relationship
                     report.quality), # manually enforced dependency relationship
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.analysis.R.
```

5.7.2 ZIP erstellen: Source Code

```
tar_target(zip.source,
                f.tar_zip(files.source,
                                filename = paste0(prefix.files,
                                                  "_Source_Code.zip"),
                                dir = "output",
                                mode = "mirror"),
                format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.source.R.
```

5.7.3 ZIP erstellen: CSV-Datei (voller Datensatz)

```
tar_target(zip.csv.final,
```

```
                f.tar_zip(csv.final,
                              filename = gsub("\\.csv", "\\.zip", basename(csv.
    final)),
                              dir = "output",
                              mode = "cherry-pick"),
                format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.csv.full.R.
```

5.7.4 ZIP erstellen: CSV-Datei (nur Metadaten)

```
tar_target(zip.csv.meta,
                f.tar_zip(csv.meta,
                              filename = gsub("\\.csv", "\\.zip", basename(csv.
    meta)),
                              dir = "output",
                              mode = "cherry-pick"),
                format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.csv.meta.R.
```

5.7.5 ZIP erstellen: TXT-Dateien

```
tar_target(zip.txt,
           f.tar_zip(x = txt_best,
                     filename = paste(prefix.files,
                                      "DE_TXT_Datensatz.zip",
                                      sep = "_"),
                     dir = "output",
                     mode = "cherry-pick"),
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.txt.R.
```

5.7.6 ZIPs erstellen: PDF-Dateien

```
tar_target(zip.pdf,
           f.tar_zip_bgh_custompacker(pdf = pdf.cleaned.noprob,
                                      dt.final = dt.final,
                                      dir = "output",
                                      prefix.files = prefix.files,
                                      skip = config$debug$skipzip),
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.pdf.regular.R.
```

5.7.7 ZIPs erstellen: Problematische PDF-Dateien

```
tar_target(zip.pdf.problem,
           f.tar_zip(x = pdf.cleaned[basename(pdf.cleaned) %in% basename(pdf.
    ocrprob)],
```

```
                      filename = paste(prefix.files,
                                       "DE_PDF_OCR-Probleme.zip",
                                       sep = "_"),
                      dir = "output",
                      mode = "cherry-pick"),
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.pdf.problem.R.
```

5.8 Kryptographische Hashes

5.8.1 Zu hashende ZIP-Archive definieren

```
tar_target(zip.all,
           c(zip.pdf,
             zip.pdf.problem,
             zip.txt,
             zip.csv.final,
             zip.csv.meta,
             zip.analysis,
             zip.source))
#> Establish _targets.R and _targets_r/targets/tar.hashes.all.R.
```

5.8.2 Kryptographische Hashes berechnen

```
tar_target(hashes,
           f.tar_multihashes(c(zip.all,
                               report.codebook[1],
                               report.quality[1]),
                             multicore = config$parallel$multihashes,
                             cores = fullCores))
#> Establish _targets.R and _targets_r/targets/tar.hashes.calc.R.
```

5.8.3 CSV schreiben: Kryptographische Hashes

```
tar_target(csv.hashes,
           f.tar_fwrite(x = hashes,
                        filename = file.path("output",
                                             paste0(prefix.files,
                                                    "_KryptographischeHashes.csv"
    ))
                        )
           )
#> Establish _targets.R and _targets_r/targets/tar.hashes.csv.R.
```

6 Pipeline: Kompilierung

6.1 Durchführen der Kompilierung

```
tar_make()
```

6.2 Pipeline archivieren

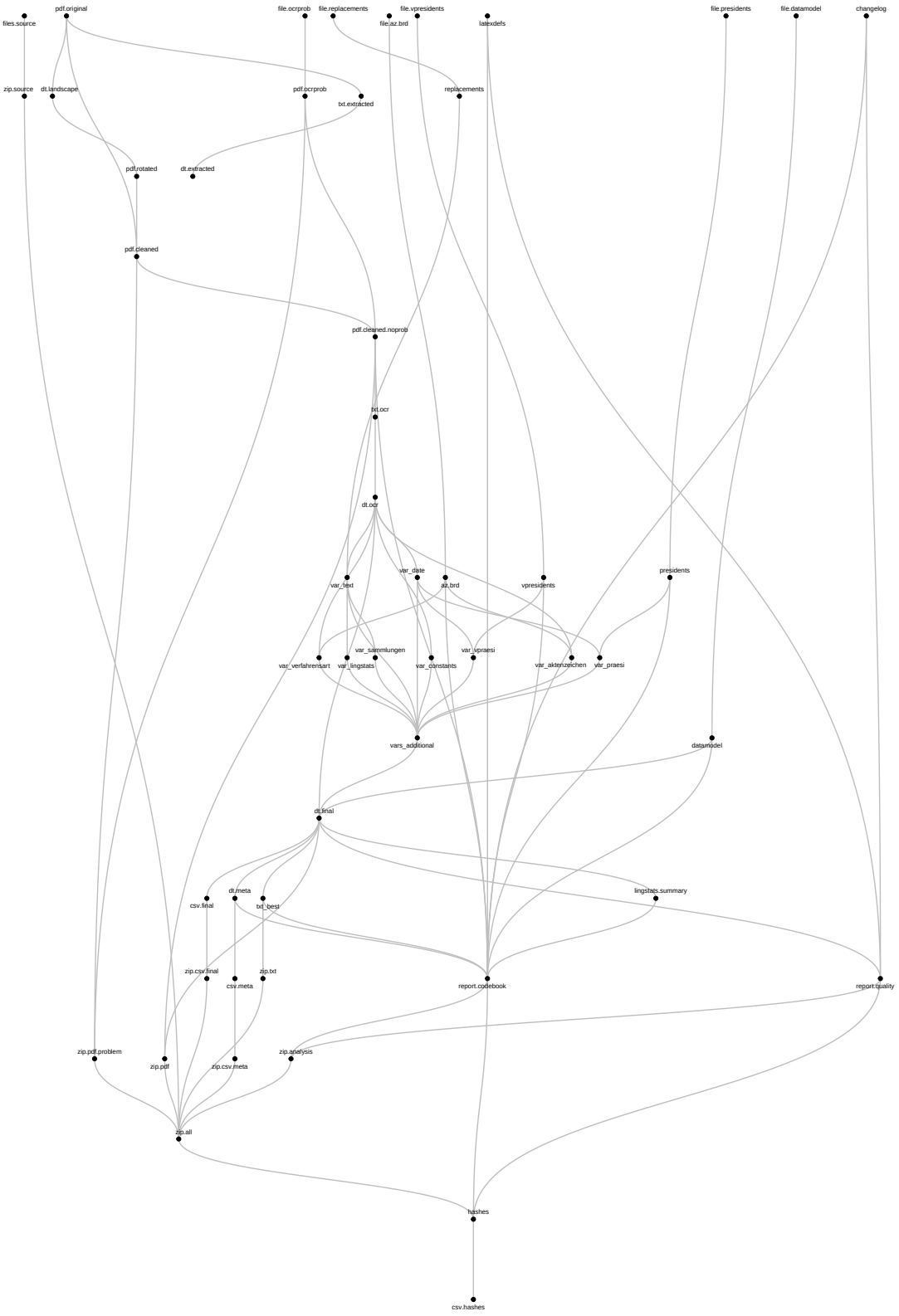
```
zip(paste0("output/",
           paste0(config$project$shortname,
                  "_",
                  config$version$dash),
           "_Targets_Storage.zip"),
    "_targets/")
```

6.3 Visualisierung

```
edgelist <- tar_network(targets_only = TRUE)$edges
#> -\|/-\|/-\|/-\|/-\|/-\|/-\|/-\|/-\|/-\|/-\|/-\|/-\|
setDT(edgelist)

g  <- igraph::graph_from_data_frame(edgelist,
                                    directed = TRUE)


ggraph(g,
       'sugiyama') +
    geom_edge_diagonal(colour = "grey")+
    geom_node_point()+
    geom_node_text(aes(label = name),
                   size = 2,
                   repel = TRUE)+
    theme_void()
```


pdf.original
file.ocrprob
file.replacements
file.vpresidents
file.presidents
file.datamodel
changelog
files.source
file.az.brd
latexdefs
zip.source
dt.landscape
pdf.ocrprob
txt.extracted
replacements
pdf.rotated
dt.extracted
pdf.cleaned
pdf.cleaned.noprob
txt.ocr
dt.ocr
var_date
presidents
var_text
az.brd
vpresidents
var_sammlungen
var_vpraesi
var_verfahrensart
var_lingstats
var_constants
var_aktenzeichen
var_praesi
vars_additional
datamodel
dt.final
dt.meta
lingstats.summary
csv.final
txt_best
zip.csv.final
zip.txt
csv.meta
report.codebook
report.quality
zip.pdf.problem
zip.pdf
zip.csv.meta
zip.analysis
zip.all
hashes
csv.hashes

7 Liste aller Targets (alphabetisch)

Die vollständige Liste aller Targets, inklusive ihres Types und ihrer Größe. Targets die auf Dateien verweisen (z.B. alle PDF-Dateien) geben die Gesamtgröße der Dateien auf der Festplatte an.

```
meta <- tar_meta(fields = c("type", "bytes", "format"), complete_only = TRUE)
setDT(meta)
meta$MB <- round(meta$bytes / 1e6, digits = 2)

# Gesamter Speicherplatzverbrauch
sum(meta$MB, na.rm = TRUE)
#> [1] 22087.25

kable(meta[order(type, name)],
      format = "latex",
      align = "r",
      booktabs = TRUE,
      longtable = TRUE) %>% kable_styling(latex_options = "repeat_header")
```

name	type	bytes	format	MB
az.brd	stem	5509	qs	0.01
changelog	stem	58	file	0.00
csv.final	stem	95	qs	0.00
csv.hashes	stem	101	qs	0.00
csv.meta	stem	95	qs	0.00
datamodel	stem	2908	qs	0.00
dt.extracted	stem	4199136	qs	4.20
dt.final	stem	168413700	qs	168.41
dt.landscape	stem	108837	qs	0.11
dt.meta	stem	662702	qs	0.66
dt.ocr	stem	85597424	qs	85.60
file.az.brd	stem	36533	file	0.04
file.datamodel	stem	8253	file	0.01
file.ocrprob	stem	667	file	0.00
file.presidents	stem	7249	file	0.01
file.replacements	stem	1133	file	0.00
file.vpresidents	stem	9193	file	0.01

(continued)

name	type	bytes	format	MB
files.source	stem	466676	file	0.47
hashes	stem	1647	qs	0.00
latexdefs	stem	1369	file	0.00
lingstats.summary	stem	378	qs	0.00
pdf.cleaned	stem	108690	qs	0.11
pdf.cleaned.noprob	stem	108665	qs	0.11
pdf.ocrprob	stem	151	qs	0.00
pdf.original	stem	10849479010	file	10849.48
pdf.rotated	stem	462	qs	0.00
presidents	stem	1959	qs	0.00
replacements	stem	716	qs	0.00
report.codebook	stem	580596	file	0.58
report.quality	stem	374860	file	0.37
txt.extracted	stem	14646461	file	14.65
txt.ocr	stem	255334125	file	255.33
txt_best	stem	253631993	file	253.63
var_aktenzeichen	stem	90597	qs	0.09
var_constants	stem	18220	qs	0.02
var_date	stem	85649	qs	0.09
var_lingstats	stem	219377	qs	0.22
var_praesi	stem	30295	qs	0.03
var_sammlungen	stem	10310	qs	0.01
var_text	stem	85483063	qs	85.48
var_verfahrensart	stem	1776	qs	0.00
var_vpraesi	stem	17287	qs	0.02
vars_additional	stem	85998986	qs	86.00
vpresidents	stem	2450	qs	0.00
zip.all	stem	196	qs	0.00
zip.analysis	stem	2529561	file	2.53

(continued)

name	type	bytes	format	MB
zip.csv.final	stem	100319807	file	100.32
zip.csv.meta	stem	936968	file	0.94
zip.pdf	stem	10025415059	file	10025.42
zip.pdf.problem	stem	31477032	file	31.48
zip.source	stem	1220490	file	1.22
zip.txt	stem	119594826	file	119.59

8 Gesamte Laufzeit

```
meta <- tar_meta(fields = c("time", "seconds"), complete_only = TRUE)
setDT(meta)
meta$mins <- round(meta$seconds / 60, digits = 2)

runtime.sum <- sum(meta$seconds)

## Sekunden
print(runtime.sum)
#> [1] 1142.295

## Minuten
runtime.sum / 60
#> [1] 19.03825

## Stunden
runtime.sum / 3600
#> [1] 0.3173042
```

9 Laufzeit einzelner Targets

Der Zeitpunkt an dem die Targets berechnet wurden und ihre jeweilige Laufzeit in Sekunden.

```
kable(meta[order(-seconds)],
      format = "latex",
      align = "r",
      booktabs = TRUE,
      longtable = TRUE) %>% kable_styling(latex_options = "repeat_header")
```

name	time	seconds	mins
zip.pdf	2024-11-25 20:31:47	415.303	6.92
var_lingstats	2024-11-25 16:44:12	198.483	3.31
dt.landscape	2024-11-12 00:09:19	110.341	1.84
lingstats.summary	2024-11-25 16:45:45	83.344	1.39
pdf.original	2017-05-21 12:29:10	71.141	1.19
dt.ocr	2024-11-13 11:38:29	57.463	0.96
var_aktenzeichen	2024-11-13 11:39:01	31.005	0.52
var_text	2024-11-25 16:40:53	22.953	0.38
zip.csv.final	2024-11-25 20:32:31	21.266	0.35
report.codebook	2024-11-25 20:48:21	21.188	0.35
report.quality	2024-11-25 20:48:00	20.819	0.35
zip.txt	2024-11-25 20:38:56	18.786	0.31
hashes	2024-11-25 20:48:39	18.229	0.30
dt.extracted	2024-11-12 00:09:45	13.986	0.23
txt.extracted	2024-11-18 16:49:21	11.840	0.20
txt_best	2024-11-25 20:38:12	11.416	0.19
var_sammlungen	2024-11-25 16:44:14	2.402	0.04
dt.final	2024-11-25 16:44:21	2.074	0.03
latexdefs	2024-11-18 16:49:10	1.425	0.02
files.source	2024-11-25 20:37:45	1.381	0.02
file.replacements	2024-11-25 16:32:55	1.365	0.02
csv.final	2024-11-25 16:46:41	1.242	0.02
zip.pdf.problem	2024-11-25 20:24:50	1.219	0.02

(continued)

name	time	seconds	mins
txt.ocr	2024-11-13 11:37:25	1.048	0.02
var_date	2024-11-13 11:38:30	0.868	0.01
pdf.rotated	2024-11-12 00:09:31	0.597	0.01
var_praesi	2024-11-13 11:39:07	0.315	0.01
var_vpraesi	2024-11-13 11:39:07	0.248	0.00
zip.csv.meta	2024-11-25 20:32:31	0.228	0.00
zip.analysis	2024-11-25 20:48:21	0.116	0.00
zip.source	2024-11-25 20:37:59	0.083	0.00
vars_additional	2024-11-25 16:44:16	0.037	0.00
csv.meta	2024-11-25 16:46:41	0.023	0.00
var_constants	2024-11-18 16:49:31	0.018	0.00
pdf.cleaned	2024-11-12 00:09:45	0.011	0.00
pdf.cleaned.noprob	2024-11-12 00:09:45	0.008	0.00
var_verfahrensart	2024-11-13 11:38:29	0.007	0.00
replacements	2024-11-25 16:40:26	0.006	0.00
dt.meta	2024-11-25 16:44:22	0.004	0.00
az.brd	2024-11-12 00:09:30	0.001	0.00
csv.hashes	2024-11-25 20:48:39	0.001	0.00
datamodel	2024-11-18 18:22:07	0.001	0.00
file.az.brd	2024-11-11 22:58:49	0.001	0.00
pdf.ocrprob	2024-11-12 00:09:30	0.001	0.00
presidents	2024-11-12 00:09:30	0.001	0.00
vpresidents	2024-11-12 00:09:30	0.001	0.00
changelog	2024-11-11 22:58:49	0.000	0.00
file.datamodel	2024-11-18 18:21:51	0.000	0.00
file.ocrprob	2024-11-11 22:58:49	0.000	0.00
file.presidents	2024-11-11 22:58:49	0.000	0.00
file.vpresidents	2024-11-11 22:58:49	0.000	0.00

zip.all	2024-11-25 20:48:21	0.000	0.00

10 Warnungen

```
meta <- tar_meta(fields = "warnings", complete_only = TRUE)
setDT(meta)
meta$warnings <- gsub("(\\.pdf|\\.html?|\\.txt)", "\\1 \n\n", meta$warnings)


if (meta[,.N > 0]){

    for(i in 1:meta[,.N]){

        cat(paste("##", meta[i]$name), "\n\n")
        cat(paste(meta[i]$warnings, "\n\n"))

    }

}else{

    cat("No warnings to report.")

}
```

10.1 dt.ocr

Fewer docnames supplied than existing docvars last 1 docvar given generic names.

10.2 report.codebook

LaTeX Warning Label variablen multiply defined.. Package microtype Warning Unable to apply patch footnote on input line 205.. LaTeX Warning There were multiplydefined labels.

10.3 report.quality

Package microtype Warning Unable to apply patch footnote on input line 215.

11 Fehlermeldungen

```
meta <- tar_meta(fields = "error", complete_only = TRUE)
setDT(meta)

if (meta[,.N > 0]){

    for(i in 1:meta[,.N]){

        cat(paste("##", meta[i]$name), "\n\n")
        cat(paste(meta[i]$error, "\n\n"))

    }

}else{

    cat("No errors to report.")

}
#> No errors to report.
```

12 Dateigrößen

12.1 ZIP-Dateien

```
files <- list.files("output", pattern = "\\.zip", full.names = TRUE)

filesize <- round(file.size(files) / 10^6, digits = 2)

table.size <- data.table(basename(files),
                         filesize)


kable(table.size,
      format = "latex",
      align = c("l", "r"),
      format.args = list(big.mark = ","),
      booktabs = TRUE,
      longtable = TRUE,
      col.names = c("Datei",
                    "Größe in MB"))
```

Datei	Größe in MB
BGH-Strafsachen-20Jhd_1-0-0_DE_Analyse.zip	2.53
BGH-Strafsachen-20Jhd_1-0-0_DE_CSV_Datensatz.zip	100.32
BGH-Strafsachen-20Jhd_1-0-0_DE_CSV_Metadaten.zip	0.94
BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_OCR-Probleme.zip	31.48
BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_Senat-1.zip	2,816.45
BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_Senat-2-und-3.zip	3,167.75
BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_Senat-4-und-5.zip	4,041.21
BGH-Strafsachen-20Jhd_1-0-0_DE_TXT_Datensatz.zip	119.59
BGH-Strafsachen-20Jhd_1-0-0_Source_Code.zip	1.22
BGH-Strafsachen-20Jhd_1-0-0_Targets_Storage.zip	431.90

12.2 CSV-Dateien

```
files <- list.files("output", pattern = "\\.csv", full.names = TRUE)

filesize <- round(file.size(files) / 10^6, digits = 2)

table.size <- data.table(basename(files),
                         filesize)


kable(table.size,
      format = "latex",
      align = c("l", "r"),
      format.args = list(big.mark = ","),
      booktabs = TRUE,
      longtable = TRUE,
      col.names = c("Datei",
                    "Größe in MB"))
```

Datei	Größe in MB
BGH-Strafsachen-20Jhd_1-0-0_DE_CSV_Datensatz.csv	522.49
BGH-Strafsachen-20Jhd_1-0-0_DE_CSV_Metadaten.csv	14.79
BGH-Strafsachen-20Jhd_1-0-0_KryptographischeHashes.csv	0.00

12.3 PDF-Dateien (MB)

```
tar_load(pdf.cleaned.noprob)
pdf.MB <- file.size(pdf.cleaned.noprob) / 10^6
sum(pdf.MB)
#> [1] 10816.82
```

12.4 TXT-Dateien (MB)

```
tar_load(txt_best)
txt.MB <- file.size(txt_best) / 10^6
sum(txt.MB)
#> [1] 253.632
```

13 Kryptographische Signaturen

13.1 Signaturen laden

```
tar_load(hashes)
```

13.2 Leerzeichen hinzufügen um bei SHA3-512 Zeilenumbruch zu ermöglichen

Hierbei handelt es sich lediglich um eine optische Notwendigkeit. Die normale 128 Zeichen lange Zeichenfolge von SHA3-512-Signaturen wird ansonsten nicht umgebrochen und verschwindet über die Seitengrenze. Das Leerzeichen erlaubt den automatischen Zeilenumbruch und damit einen für Menschen sinnvoll lesbaren Abdruck im Codebook. Diese Variante wird nur zur Anzeige verwendet und danach verworfen.

```
hashes$sha3.512 <- paste(substr(hashes$sha3.512, 1, 64),
                         substr(hashes$sha3.512, 65, 128))
```

13.3 In Bericht anzeigen

```
kable(hashes[,.(index,filename)],
      format = "latex",
      align = c("p{1cm}",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	filename
1	output/BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_Senat-1.zip
2	output/BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_Senat-2-und-3.zip
3	output/BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_Senat-4-und-5.zip
4	output/BGH-Strafsachen-20Jhd_1-0-0_DE_PDF_OCR-Probleme.zip
5	output/BGH-Strafsachen-20Jhd_1-0-0_DE_TXT_Datensatz.zip
6	output/BGH-Strafsachen-20Jhd_1-0-0_DE_CSV_Datensatz.zip
7	output/BGH-Strafsachen-20Jhd_1-0-0_DE_CSV_Metadaten.zip
8	output/BGH-Strafsachen-20Jhd_1-0-0_DE_Analyse.zip
9	output/BGH-Strafsachen-20Jhd_1-0-0_Source_Code.zip
10	output/BGH-Strafsachen-20Jhd_1-0-0_Codebook.pdf

11	output/BGH-Strafsachen-20Jhd_1-0-0_QualityControlReport.pdf

```
kable(hashes[,.(index,sha2.256)],
      format = "latex",
      align = c("c",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	sha2.256
1	220b384f65be62ad851460f47887cc684cd348bb2fc31b61491888003cac13ed
2	015801d3a92a1d14146cca776c7b5d7c895cde41294daec87886b489955efde9
3	123a33eb72c1fffdc43cb02ced6089540ea42e1360e4f02179341375dfbf6348
4	144645a579d8f70def56e14c8971d01daf9406164fe0eb5fff35e8cf0a8863a6
5	96c1885afe7646ba1a98ec0c45c5f9b3316b4411fb8087981318d87176cc1f98
6	4bdcc28a992b07eee404d56cb46300c56e048cfd3bac3a7262e69e9c59d6456e
7	e671eb569b7faa64445d534f2b92c948e0f14a928fe8e3726ddcf6b8f3d2c388
8	f0d1c8963712bbf6879f9e0ae7c3019eeff435a7ca7c447e62d968cb7c140d67
9	d5537e95fffee1371d6d4e33f24665f99f5799f3d32a32aefd3ef11da2e24402
10	1b1d75601b545b34461b06a28d743db46451e795986364e29f6aa6d023f4d660
11	85840590a752fb2d5d74a179aa0024468097d1bbbad0347a02bd951d31e6ebb8

```
kable(hashes[,.(index,sha3.512)],
      format = "latex",
      align = c("c",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	sha3.512
1	b1a791ab10815cc6be464431853294681c44a8b09c6d33d743a1ec5af566eb14 0969767e19edb707f0e6341f5eafaf6dc7bf78d3f75b8a07509348f08f1a5008
2	0b85eb6f16d5aab53658a907d93aaad6df29e37979ce7dfbed8cfd7f14be94b7 cb5a59e5bab7d23afce899799a5f747f9e64598ff80a3b8ebdc4a8f4b187e941
3	6193ffe81eb040b665163111ae6f1c38e6403561bf6652247450d91b84c51152 2f40091f4249f9710b06a8a562d717822873f4035c7dbfa489ea4811f4cb28b0
4	0db83f1b6fd33559ac859984cf830b6e8af8a0c658df5a4e57a5e42c7bbdfd47 710bc3cd88acf7df2b92659bc74cb5abc377374386717ac7833a0735e5a52749
5	f31c9316583251e6a799984e889d61e51ef860f41a198aeb865de6e4eb1a1257 975e01c1739d22d0ae244683710136c9f66689e0b42c3a6106d6e7a0aa536c17
6	973a733941b0c8ae43811583185aa581b1b6c4b00eb975c3905927aef098f327 addd42c2c8fc572610e33c6f1be460b91e4321a1e9a4b1321be7623dddb53fb7
7	c03b851206da38ae5a27cf150fbfb0b0e81ca205efbd98c9561c96e654c6a11e a12f6370904529d52e7266f5c8d7408987612a986711ce066ca9aaa46b60ace4
8	9a9d1bbba0351f59ab16b765f4304ed4151f4f37e615055eb540a0c068e11717 cd7003e0c3b40fbd88dee8f04843439a7f35689227154ddb934aed3a824003d2
9	7489405283042f915611fe510e4e44aa57666a6bee346d0a550e17d5907ebdc8 30fe81614c5ebc8ef15667d2e5512e0745f071ee7af6b7b0447ec049668d3099
10	f04f7bbcd35e2016930a93ef6539e2cb1961574baadc7d0e128db4dc6adf6adc 70071a1a1f9a13c3f8a7476eda06eea6f2de79f3877d9a8d4c148f628ba11c2c
11	8bcfe3a03790b9d5245119fb42cb685de2504224c4dd3e641da0fb27b0ed4833 e5744663b4859b1c149529d7ee82b1f420dda64390b6743b16bcde813eea70b0

14 Changelog

14.1 Version 1.0.0

- Erstveröffentlichung

15 Abschluss

```
## Datumsstempel
print(datestamp)
#> [1] "2024-11-25"

## Datum und Uhrzeit (Anfang)
print(begin.script)
#> [1] "2024-11-25 20:47:24 CET"


## Datum und Uhrzeit (Ende)
end.script <- Sys.time()
print(end.script)
#> [1] "2024-11-25 20:49:03 CET"


## Laufzeit des gesamten Skriptes
print(end.script - begin.script)
#> Time difference of 1.65118 mins
```

16 Parameter für strenge Replikationen

```
system2("openssl", "version", stdout = TRUE)
#> [1] "OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)"

sessionInfo()
#> R version 4.4.0 (2024-04-24)
#> Platform: x86_64-pc-linux-gnu
#> Running under: Ubuntu 22.04.4 LTS
#>
#> Matrix products: default
#> BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
#> LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so;
    LAPACK version 3.10.0
#>
#> locale:
#>  [1] LC_CTYPE=en_US.utf8       LC_NUMERIC=C
#>  [3] LC_TIME=en_US.utf8        LC_COLLATE=en_US.utf8
#>  [5] LC_MONETARY=en_US.utf8    LC_MESSAGES=en_US.utf8
#>  [7] LC_PAPER=en_US.utf8       LC_NAME=C
#>  [9] LC_ADDRESS=C              LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=en_US.utf8 LC_IDENTIFICATION=C
#>
#> time zone: Europe/Berlin
#> tzcode source: system (glibc)
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#>  [1] ggraph_2.2.1      ggplot2_3.5.1     igraph_2.0.3      kableExtra_1.4.0
#>  [5] knitr_1.48        quanteda_4.1.0    data.table_1.16.0 future_1.34.0
#>  [9] RcppTOML_0.2.2    magrittr_2.0.3    tarchetypes_0.9.0 targets_1.7.1
#>
#> loaded via a namespace (and not attached):
#>  [1] fastmatch_1.1-4     gtable_0.3.5        xfun_0.47
#>  [4] ggrepel_0.9.6       processx_3.8.4      RApiSerialize_0.1.3
#>  [7] lattice_0.22-6      callr_3.7.6         vctrs_0.6.5
#> [10] tools_4.4.0         ps_1.8.0            generics_0.1.3
#> [13] base64url_1.4       parallel_4.4.0      tibble_3.2.1
#> [16] fansi_1.0.6         highr_0.11          pkgconfig_2.0.3
#> [19] Matrix_1.7-0        secretbase_1.0.2    RcppParallel_5.1.9
#> [22] lifecycle_1.0.4     farver_2.1.2        compiler_4.4.0
#> [25] stringr_1.5.1       tinytex_0.53        munsell_0.5.1
#> [28] ggforce_0.4.2       qs_0.26.3           graphlayouts_1.2.0
#> [31] codetools_0.2-20    htmltools_0.5.8.1   yaml_2.3.10
#> [34] pillar_1.9.0        tidyr_1.3.1         MASS_7.3-60.2
#> [37] cachem_1.1.0        viridis_0.6.5       parallelly_1.38.0
#> [40] stopwords_2.3       tidyselect_1.2.1    digest_0.6.37
#> [43] stringi_1.8.4       dplyr_1.1.4         purrr_1.0.2
#> [46] listenv_0.9.1       labeling_0.4.3      polyclip_1.10-7
#> [49] fastmap_1.2.0       grid_4.4.0          colorspace_2.1-1
#> [52] cli_3.6.3           tidygraph_1.3.1     utf8_1.2.4
#> [55] withr_3.0.1         scales_1.3.0        backports_1.5.0
#> [58] rmarkdown_2.28      globals_0.16.3      gridExtra_2.3
```

```
#> [61] stringfish_0.16.0   memoise_2.0.1       evaluate_1.0.0
#> [64] viridisLite_0.4.2   rlang_1.1.4         Rcpp_1.0.13
#> [67] glue_1.7.0          renv_1.0.9          tweenr_2.0.3
#> [70] xml2_1.3.6          svglite_2.1.3       rstudioapi_0.16.0
#> [73] R6_2.5.1            systemfonts_1.1.0   fs_1.6.4
```

Literaturverzeichnis

Allaire, JJ, Yihui Xie, Christophe Dervieux, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, et al. 2024. *Rmarkdown: Dynamic Documents for R*. https://github.com/rstudio/rmarkdown.

Barrett, Tyson, Matt Dowle, Arun Srinivasan, Jan Gorecki, Michael Chirico, Toby Hocking, and Benjamin Schwendinger. 2024. *Data.table: Extension of 'Data.frame'*. https://r-datatable.com.

Bengtsson, Henrik. 2021. "A Unifying Framework for Parallel and Distributed Processing in R Using Futures." *The R Journal* 13 (2): 208–27. https://doi.org/10.32614/RJ-2021-048.

———. 2023. *Future.callr: A Future Api for Parallel Processing Using Callr*. https://future.callr.futureverse.org.

———. 2024a. *Future.apply: Apply Function to Elements in Parallel Using Futures*. https://future.apply.futureverse.org.

———. 2024b. *Future: Unified Parallel and Distributed Processing in R for Everyone*. https://future.futureverse.org.

Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. 2018. "Quanteda: An R Package for the Quantitative Analysis of Textual Data." *Journal of Open Source Software* 3 (30): 774. https://doi.org/10.21105/joss.00774.

Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, Akitaka Matsuo, and William Lowe. 2024. *Quanteda: Quantitative Analysis of Textual Data*. https://quanteda.io.

Csárdi, Gábor. 2024. *Zip: Cross-Platform Zip Compression*. https://github.com/r-lib/zip.

Csardi, Gabor, and Tamas Nepusz. 2006. "The Igraph Software Package for Complex Network Research." *InterJournal* Complex Systems: 1695. https://igraph.org.

Csárdi, Gábor, Tamás Nepusz, Vincent Traag, Szabolcs Horvát, Fabio Zanini, Daniel Noom, and Kirill Müller. 2024. *Igraph: Network Analysis and Visualization*. https://r.igraph.org/.

Eddelbuettel, Dirk. 2023. *RcppTOML: Rcpp Bindings to Parser for "Tom's Obvious Markup Language"*. http://dirk.eddelbuettel.com/code/rcpp.toml.html.

Ewing, Mark. 2021. *Mgsub: Safe, Multiple, Simultaneous String Substitution*.

Gagolewski, Marek. 2022. "stringi: Fast and Portable Character String Processing in R." *Journal of Statistical Software* 103 (2): 1–59. https://doi.org/10.18637/jss.v103.i02.

Gagolewski, Marek, Bartek Tartanus, others; Unicode, Inc., and others. 2024. *Stringi: Fast and Portable Character String Processing Facilities*. https://stringi.gagolewski.com/.

Landau, William Michael. 2021a. *Tarchetypes: Archetypes for Targets*.

———. 2021b. "The Targets R Package: A Dynamic Make-Like Function-Oriented Pipeline Toolkit for Reproducibility and High-Performance Computing." *Journal of Open Source Software* 6 (57): 2959. https://doi.org/10.21105/joss.02959.

———. 2024a. *Tarchetypes: Archetypes for Targets*. https://docs.ropensci.org/tarchetypes/.

———. 2024b. *Targets: Dynamic Function-Oriented Make-Like Declarative Pipelines*. https://docs.ropensci.org/targets/.

Ooms, Jeroen. 2024a. *Magick: Advanced Graphics and Image-Processing in R*. https://docs.ropensci.org/magick/.

———. 2024b. *Pdftools: Text Extraction, Rendering and Converting of Pdf Documents*. https://docs.ropensci.org/pdftools/.

———. 2024c. *Qpdf: Split, Combine and Compress Pdf Files*. https://docs.ropensci.org/qpdf/.

Pedersen, Thomas Lin. 2024. *Ggraph: An Implementation of Grammar of Graphics for Graphs and Networks*. https://ggraph.data-imaginist.com.

Ushey, Kevin, and Hadley Wickham. 2024. *Renv: Project Environments*. https://rstudio.github.io/renv/.

Wickham, Hadley. 2011. "Testthat: Get Started with Testing." *The R Journal* 3: 5–10. https://journal.r-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf.

———. 2024. *Testthat: Unit Testing for R*. https://testthat.r-lib.org.

Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC.

———. 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. https://yihui.org/knitr/.

———. 2024. *Knitr: A General-Purpose Package for Dynamic Report Generation in R*. https://yihui.org/knitr/.

Xie, Yihui, J. J. Allaire, and Garrett Grolemund. 2018. *R Markdown: The Definitive Guide*. Boca Raton, Florida: Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown.

Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. *R Markdown Cookbook*. Boca Raton, Florida: Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown-cookbook.

Zhu, Hao. 2024. *KableExtra: Construct Complex Table with Kable and Pipe Syntax*. http://haozhu233.github.io/kableExtra/.