Corpus der Entscheidungen des Bundesverwaltungsgerichts

(CE-BVerwG-Source)

Compilation Report

Version 2024-03-13

DOI: 10.5281/zenodo.10809041

Titel	Source Code des »Corpus der Entscheidungen des Bundesverwaltungsgerichts«
Abkürzung	CE-BVerwG-Source
Autor	Seán Fobbe
Version	2024-03-13
Download	https://doi.org/10.5281/zenodo.10809041
Lizenz	MIT No Attribution (MIT-0)

Zitiervorschlag

Seán Fobbe (2024). Source Code des »Corpus der Entscheidungen des Bundesverwaltungsgerichts« (CE-BVerwG-Source). Version 2024-03-13. Zenodo. DOI: 10.5281/zenodo.10809041.

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 2024-03-13. 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. Sie lautet 10.5281/zenodo.4625134. Die »Concept DOI« verlinkt immer die aktuellste Version.

Lizenz: MIT No Attribution (MIT-0)

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 README: Corpus der Entscheidungen des Bundesverwaltungsgerichts (CE-BVerwG)

1.1 Überblick

Das **Corpus der Entscheidungen des Bundesverwaltungsgerichts (CE-BVerwG)** ist eine möglichst vollständige Sammlung der vom Bundesverwaltungsgericht veröffentlichten Entscheidungen. Der Datensatz nutzt als seine Datenquelle die amtliche Entscheidungsdatenbank des Bundesverwaltungsgerichts und wertet diese vollständig aus.

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.3911067

1.2 Funktionsweise

Primäre Endprodukte des Skripts sind folgende ZIP-Archive:

- Der volle Datensatz im CSV-Format
- Die reinen Metadaten im CSV-Format (wie unter 1, nur ohne Entscheidungstexte)
- (Optional) Linguistisch annotierte Fassung aller Entscheidungen im CSV-Format
- Alle Entscheidungen im TXT-Format (reduzierter Umfang an Metadaten)
- Alle Entscheidungen im PDF-Format (reduzierter Umfang an Metadaten)
- Alle Analyse-Ergebnisse (Tabellen als CSV, Grafiken als PDF und PNG)
- Der Source Code und alle weiteren Quelldaten

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.

1.3 Systemanforderungen

- Docker
- Docker Compose
- 12 GB Speicherplatz auf Festplatte
- 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.

1.4 Anleitung

1.4.1 Schritt 1: Ordner vorbereiten

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

```
$ git clone https://github.com/seanfobbe/ce-bverwge
```

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.

1.4.2 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
```

1.4.3 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
```

1.4.4 Ergebnis

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

1.5 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
```

1.6 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
```

1.7 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
├ compose.yaml               # Konfiguration für Docker
├ 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
├ Dockerfile                 # Definition des Docker Images
├ docker-run-project.sh      # Docker Image und Datensatz kompilieren
├ functions                  # Wichtige Schritte der Pipeline
├ gpg                        # Persönlicher Public GPG-Key für Seán Fobbe
├ old                        # Alter Code aus früheren Versionen
├ pipeline.Rmd               # Zentrale Definition der Pipeline
├ README.md                  # Bedienungsanleitung
├ reports                    # Markdown-Dateien
├ requirements-python.txt    # Benötigte Python packages
├ requirements-R.R           # Benötigte R packages
├ requirements-system.txt    # Benötigte system dependencies
├ run_project.R              # Kompiliert den gesamten Datensatz
└ tex                       # LaTeX-Templates
```

1.8 Weitere Open Access Veröffentlichungen (Fobbe)

Website — https://www.seanfobbe.de

Open Data — https://zenodo.org/communities/sean-fobbe-data/

Source Code — https://zenodo.org/communities/sean-fobbe-code/

Volltexte regulärer Publikationen — https://zenodo.org/communities/sean-fobbe-publications/

1.9 Kontakt

Fehler gefunden? Anregungen? Kommentieren Sie gerne im Issue Tracker auf GitHub oder schreiben Sie mir eine E-Mail an fobbe-data@posteo.de

2 Packages laden

```
library(targets)
library(tarchetypes)
library(RcppTOML)
library(future)
library(data.table)
library(quanteda)
#> Package version: 3.2.4
#> Unicode version: 14.0
#> ICU version: 70.1
#> Parallel computing: 16 of 16 threads used.
#> See https://quanteda.io for tutorials and examples.
library(knitr)
library(kableExtra)
library(igraph)
#>
#> Attaching package: 'igraph'
#> The following objects are masked from 'package:future':
#>
#>     %->%, %<-%
#> The following objects are masked from 'package:stats':
#>
#>     decompose, spectrum
#> The following object is masked from 'package:base':
#>
#>     union
library(ggraph)
#> Loading required package: ggplot2

tar_unscript()
```

3 Vorbereitung

3.1 Definitionen

```
## Datum
datestamp <- Sys.Date()
print(datestamp)
#> [1] "2024-03-13"

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

## Konfiguration
config <- RcppTOML::parseTOML("config.toml")
print(config)
#> List of 10
#>  $ annotate:List of 1
#>   ..$ toggle: logi TRUE
#>  $ cores   :List of 2
#>   ..$ max   : logi TRUE
#>   ..$ number: int 8
#>  $ debug   :List of 3
#>   ..$ cleanrun: logi FALSE
#>   ..$ files   : int 300
#>   ..$ toggle  : logi FALSE
#>  $ doi     :List of 4
#>   ..$ aktenzeichen : chr "10.5281/zenodo.4569564"
#>   ..$ data         :List of 2
#>   .. ..$ concept: chr "10.5281/zenodo.3911067"
#>   .. ..$ version: chr "10.5281/zenodo.10809039"
#>   ..$ personendaten: chr "10.5281/zenodo.4568682"
#>   ..$ software     :List of 2
#>   .. ..$ concept: chr "10.5281/zenodo.4625134"
#>   .. ..$ version: chr "10.5281/zenodo.10809041"
#>  $ download:List of 2
#>   ..$ max    : int 28000
#>   ..$ 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 "MIT-0"
#>   ..$ data: chr "Creative Commons Zero 1.0 Universal"
#>  $ parallel:List of 4
#>   ..$ annotate     : logi TRUE
#>   ..$ extractPDF   : logi TRUE
#>   ..$ lingsummarize: logi TRUE
#>   ..$ multihashes  : logi TRUE
#>  $ project :List of 3
#>   ..$ author   : chr "Seán Fobbe"
#>   ..$ fullname : chr "Corpus der Entscheidungen des Bundesverwaltungsgerichts"
#>   ..$ shortname: chr "CE-BVerwG"
#>  $ quanteda:List of 1
```

```
#>   ..$ tokens_locale: chr "de_DE"


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

3.2 Aufräumen

Löscht Dateien im Output-Ordner, die nicht vom heutigen Tag sind.

```
unlink(grep(datestamp,
            list.files("output",
                       full.names = TRUE),
            invert = TRUE,
            value = TRUE))
```

3.3 Ordner erstellen

```
#unlink("output", recursive = TRUE)
dir.create("output", showWarnings = FALSE)
dir.create("temp", showWarnings = FALSE)

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

3.4 Vollzitate statistischer Software schreiben

```
knitr::write_bib(renv::dependencies()$Package,
                 "temp/packages.bib")
#> Finding R package dependencies ... Done!
#> Warning in knitr::write_bib(renv::dependencies()$Package, "temp/packages.bib")
    :
#> package(s) doParallel not found
```

4 Globale Variablen

4.1 Packages definieren

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

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 | DOI:",
                 config$doi$data$version)


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

## File prefix
prefix.files <- paste0(config$project$shortname,
```

```
                        "_",
                        datestamp)


if (config$cores$max == TRUE){
    fullCores <- future::availableCores() - 1
}


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

#> 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("gericht",
                 "datum",
                 "entscheidung_typ",
                 "spruchkoerper_az",
                 "registerzeichen",
                 "eingangsnummer",
                 "eingangsjahr_az",
                 "verzoegerung",
                 "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 und eingelesen. 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 Variablen

Die Variablen des Datensatzes, inklusive ihrer Erläuterung.

```
list(
    tar_target(file.variables.codebook,
               "data/CE-BVerwG_Variables.csv",
               format = "file"),
    tar_target(variables.codebook,
               fread(file.variables.codebook))
)
#> Establish _targets.R and _targets_r/targets/tar.filevars.R.
```

5.1.2 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.azbrd.R.
```

5.1.3 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_2023-03-21_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_2023-03-21_GermanFederalCourts_VicePresidents.csv",
           format = "file"),
     tar_target(vpresidents,
                fread(file.vpresidents))
     )
#> Establish _targets.R and _targets_r/targets/tar.file.vpresi.R.
```

5.1.4 Source Code

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

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

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

5.1.5 Changelog

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

5.2 Download Targets

5.2.1 URLs zu PDF-Dateien abrufen

```
tarchetypes::tar_age(url.pdf,
                     f.pdflinks_bverwg(download.max = config$download$max),
                     age = as.difftime(1, units = "days"))
#> Establish _targets.R and _targets_r/targets/tar.download.url.R.
```

5.2.2 Dateinamen erstellen

```
tar_target(filenames.pdf,
           f.filenames.bverwg(url.pdf))
#> Establish _targets.R and _targets_r/targets/tar.download.filename.R.
```

5.2.3 Finale Download-Tabelle erstellen

```
tar_target(dt.download.final,
           f.download_table_final(url = url.pdf,
                                  doc_id = filenames.pdf,
                                  debug.toggle = config$debug$toggle,
                                  debug.files = config$debug$files))
#> Establish _targets.R and _targets_r/targets/tar.download.table.R.
```

5.2.4 Download durchführen

```
tar_target(files.pdf,
                f.download(url = dt.download.final$url,
                           filename = dt.download.final$doc_id,
                           dir = "files/pdf",
                           sleep.min = 0.3,
                           sleep.max = 1,
                           retries = 3,
                           retry.sleep.min = 2,
                           retry.sleep.max = 5,
                           timeout = config$download$timeout,
                           debug.toggle = config$debug$toggle,
                           debug.files = config$debug$files),
                format = "file")

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

5.3 Convert Targets

Durch diesen Abschnitt der Pipeline werden die PDF-Dateien in TXT konvertiert und mitsamt den Variablen in ihren Dateinamen eingelesen. Beim Einlesen werden die in PDF-Dateien üblichen über Zeilen gebrochene Wörter wieder zusammengefügt.

```
list(tar_target(files.txt,
                f.tar_pdf_extract(files.pdf,
                                  outputdir = "files/txt",
                                  cores = fullCores),
                format = "file"),
     tar_target(dt.bverwg,
                f.readtext(x = files.txt,
                           docvarnames = docvarnames))
     )

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

5.4 Enhance Targets

Dieser Abschnitt der Pipeline berechnet diverse Verbesserungen für den Datensatz und führt diese am Ende zusammen.

5.4.1 Daten standardisieren

Das Datum wird im ISO-Format standardisiert und die Variablen »entscheidungsjahr« und »eingangsjahr_iso« hinzugefügt.

```
tar_target(dt.bverwg.datecleaned,
                f.clean_dates(dt.bverwg))
#> Establish _targets.R and _targets_r/targets/tar.enhance.dates.R.
```

5.4.2 Variable erstellen: »verfahrensart«

Die Variable »verfahrensart« wird aus den Registerzeichen berechnet.

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

5.4.3 Variable erstellen: »aktenzeichen«

Das Aktenzeichen wird aus seinen Komponenten berechnet.

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

5.4.4 Variable erstellen: »ecli«

Die ECLI wird aus ihren Komponenten berechnet.

```
tar_target(var_ecli,
           f.var_ecli_bverwg(x = dt.bverwg.datecleaned,
                             dt.download.final = dt.download.final,
                             files.pdf = files.pdf))
#> Establish _targets.R and _targets_r/targets/tar.enhance.ecli.R.
```

5.4.5 Variable erstellen: »praesi«

```
tar_target(var_praesi,
           f.presidents(datum = dt.bverwg.datecleaned$datum,
                        gericht = "BVerwG",
                        pvp.fcg = presidents))
#> Establish _targets.R and _targets_r/targets/tar.enhance.presi.R.
```

5.4.6 Variable erstellen: »vpraesi«

```
tar_target(var_vpraesi,
           f.presidents(datum = dt.bverwg.datecleaned$datum,
                        gericht = "BVerwG",
                        pvp.fcg = vpresidents))
#> Establish _targets.R and _targets_r/targets/tar.enhance.vpresi.R.
```

5.4.7 Variablen erstellen: »bverwge, uebersetzung, leitsatz, fachpresse«

```
tar_target(var_sammlungen,
                f.var_sammlungen(dt.bverwg.datecleaned$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(dt.bverwg.datecleaned,
                            multicore = config$parallel$lingsummarize,
                            cores = fullCores,
                            germanvars = TRUE,
                            tokens_locale = "de_DE"))
#> 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 = as.character(datestamp),
                      doi_concept = config$doi$data$concept,
                      doi_version = config$doi$data$version,
                      lizenz = as.character(config$license$data))[rep(1,
                                                                      nrow(dt.
    bverwg.datecleaned)),])
#> Establish _targets.R and _targets_r/targets/tar.enhance.constants.R.
```

5.4.10 Zusätzliche Variablen zusammenführen

```
tar_target(vars_additional,
           data.table(verfahrensart = var_verfahrensart,
                      aktenzeichen = var_aktenzeichen,
```

```
                        ecli = var_ecli,
                        praesi = var_praesi,
                        v_praesi = var_vpraesi,
                        var_lingstats,
                        var_sammlungen,
                        var_constants))

#> Establish _targets.R and _targets_r/targets/tar.enhance.combine.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.bverwg.full,
           f.dataset_finalize(x = dt.bverwg.datecleaned,
                              download.table = dt.download.final,
                              vars.additional = vars_additional,
                              varnames = variables.codebook$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.bverwg.meta,
               dt.bverwg.full[, !"text"])

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

5.4.13 Variante erstellen: Linguistische Annotationen

```
tar_target(dt.annotated,
           f.future_spacyparse(dt.bverwg.full,
                               chunksperworker = 1,
                               chunksize = NULL,
                               model = "de_core_news_sm",
                               pos = TRUE,
                               tag = TRUE,
                               lemma = TRUE,
                               entity = TRUE,
                               dependency = TRUE,
                               nounphrase = TRUE,
                               multicore = config$parallel$annotate,
                               cores = fullCores))

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

5.5 Write Targets

Dieser Abschnitt der Pipeline schreibt alle CSV-Varianten auf die Festplatte.

5.5.1 CSV schreiben: Voller Datensatz

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

5.5.2 CSV schreiben: Metadaten

```
tar_target(csv.meta,
           f.tar_fwrite(x = dt.bverwg.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 CSV schreiben: Metadaten

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

5.6 ZIP Targets

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

5.6.1 ZIP erstellen: PDF-Dateien (alle Entscheidungen)

```
tar_target(zip.pdf.all,
           f.tar_zip(files.pdf,
                     filename = paste(prefix.files,
```

```
                                        "DE_PDF_Datensatz.zip",
                                        sep = "_"),
                        dir = "output",
                        mode = "cherry-pick"),
            format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.pdf.R.
```

5.6.2 ZIP erstellen: TXT-Dateien

```
tar_target(zip.txt,
           f.tar_zip(files.txt,
                     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.
```

5.6.3 ZIP erstellen: CSV-Datei (voller Datensatz)

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

5.6.4 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.
```

5.6.5 ZIP erstellen: CSV-Datei (Linguistische Annotationen)

```
tar_target(zip.csv.annotated,
                f.tar_zip(csv.annotated,
                                filename = gsub("\\.csv", "\\.zip", basename(csv.
    annotated)),
```

```
                                dir = "output",
                                mode = "cherry-pick"),
                format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.csv.annotated.R.
```

5.6.6 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.robustness), # manually enforced dependency
    relationship
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.analysis.R.
```

5.6.7 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 Report Targets

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

5.7.1 LaTeX-Definitionen schreiben

Um 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 = datestamp),
           format = "file")

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

5.7.2 Zusammenfassungen linguistischer Kennwerte berechnen

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

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

5.7.3 Report erstellen: Robustness Checks

```
tarchetypes::tar_render(report.robustness,
                        file.path("reports",
                                  "RobustnessChecks.Rmd"),
                        output_file = file.path("../output",
                                                paste0(config$project$shortname,
                                                       "_",
                                                       datestamp,
                                                       "_RobustnessChecks.pdf")))

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

5.7.4 Report erstellen: Codebook

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

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

5.8 Kryptographische Hashes

5.8.1 Zu hashende ZIP-Archive definieren

```
tar_target(zip.all,
                c(zip.pdf.all,
                  zip.txt,
                  zip.csv.full,
                  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.robustness[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

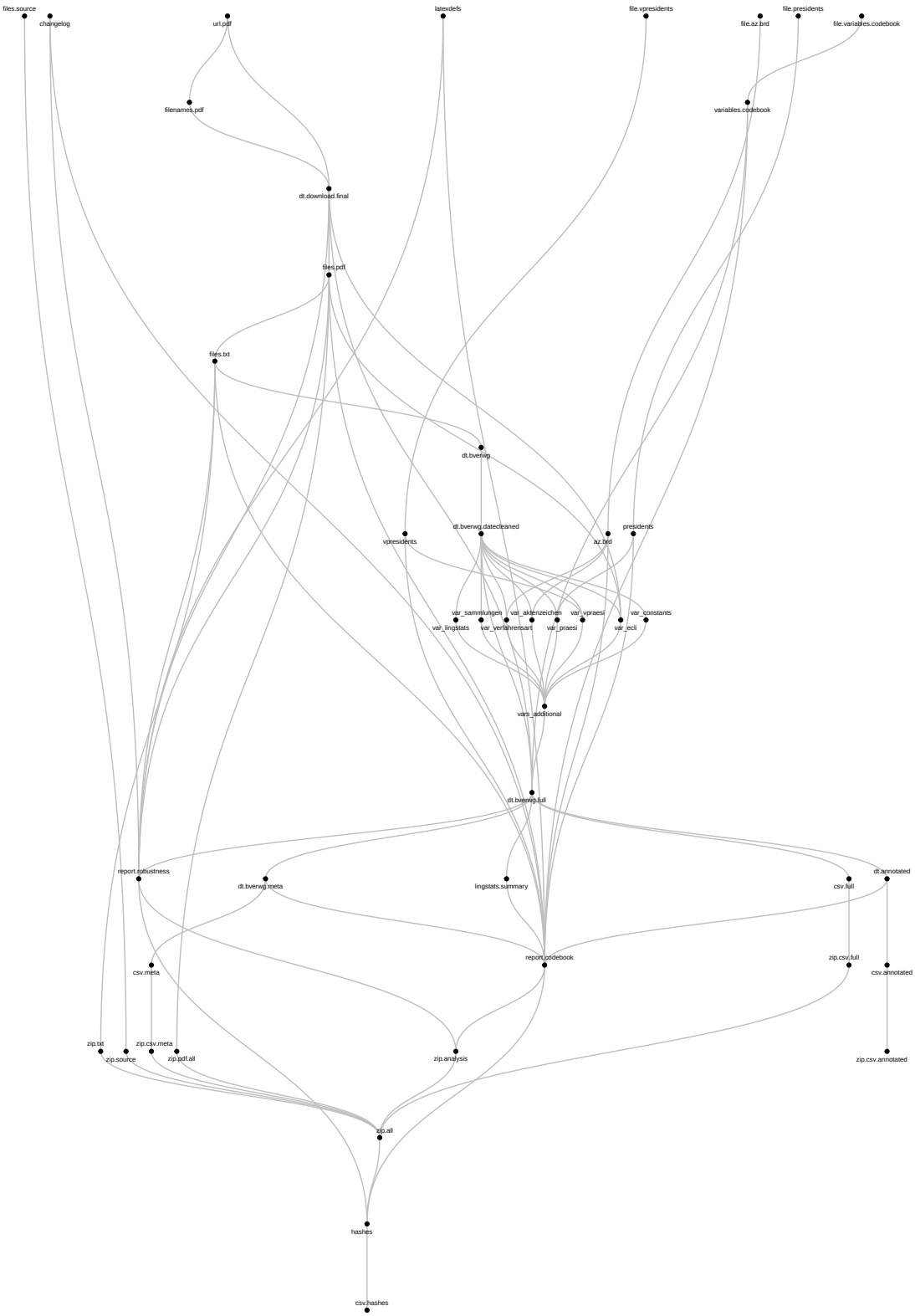
```
zip(paste0("output/",
           paste0(config$project$shortname,
                  "_",
                  datestamp),
           "_Targets_Storage.zip"),
    "_targets/")
```

6.3 Visualisierung

```
edgelist <- tar_network(targets_only = TRUE)$edges
setDT(edgelist)

g  <- igraph::graph.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()
#> Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2
    3.4.0.
#> i Please use `linewidth` in the `default_aes` field and elsewhere instead.
```


files.source
changelog
url.pdf
latexdefs
file.vpresidents
file.az.brd
file.presidents
file.variables.codebook
filenames.pdf
variables.codebook
dt.download.final
files.pdf
files.txt
dt.bverwg
dt.bverwg.datecleaned
vpresidents
az.brd
presidents
var_sammlungen
var_aktenzeichen
var_vpraesi
var_constants
var_lingstats
var_verfahrensart
var_praesi
var_ecli
vars_additional
dt.bverwg.full
report.robustness
dt.bverwg.meta
lingstats.summary
csv.full
dt.annotated
report.codebook
csv.meta
zip.csv.full
csv.annotated
zip.txt
zip.csv.meta
zip.source
zip.pdf.all
zip.analysis
zip.csv.annotated
zip.all
hashes
csv.hashes

7 Pipeline: Profiling

7.1 Gesamte Liste

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] 5201.44

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	3808	file	0.00
csv.annotated	stem	88	qs	0.00
csv.full	stem	88	qs	0.00
csv.hashes	stem	94	qs	0.00
csv.meta	stem	88	qs	0.00
dt.annotated	stem	417063702	qs	417.06
dt.bverwg	stem	95067250	qs	95.07
dt.bverwg.datecleaned	stem	95069068	qs	95.07
dt.bverwg.full	stem	95668825	qs	95.67
dt.bverwg.meta	stem	815471	qs	0.82
dt.download.final	stem	389426	qs	0.39
file.az.brd	stem	36533	file	0.04
file.presidents	stem	10300	file	0.01
file.variables.codebook	stem	7342	file	0.01
file.vpresidents	stem	11295	file	0.01

(continued)

name	type	bytes	format	MB
filenames.pdf	stem	195484	qs	0.20
files.pdf	stem	1774347052	file	1774.35
files.source	stem	796320	file	0.80
files.txt	stem	408608343	file	408.61
hashes	stem	1260	qs	0.00
latexdefs	stem	1295	file	0.00
lingstats.summary	stem	392	qs	0.00
presidents	stem	2346	qs	0.00
report.codebook	stem	618495	file	0.62
report.robustness	stem	361749	file	0.36
url.pdf	stem	192750	qs	0.19
var_aktenzeichen	stem	78918	qs	0.08
var_constants	stem	15670	qs	0.02
var_ecli	stem	141425	qs	0.14
var_lingstats	stem	171362	qs	0.17
var_praesi	stem	253	qs	0.00
var_sammlungen	stem	7723	qs	0.01
var_verfahrensart	stem	40656	qs	0.04
var_vpraesi	stem	404	qs	0.00
variables.codebook	stem	2771	qs	0.00
vars_additional	stem	442982	qs	0.44
vpresidents	stem	2775	qs	0.00
zip.all	stem	147	qs	0.00
zip.analysis	stem	2710655	file	2.71
zip.csv.annotated	stem	702305910	file	702.31
zip.csv.full	stem	105902571	file	105.90
zip.csv.meta	stem	982752	file	0.98
zip.pdf.all	stem	1353661574	file	1353.66
zip.source	stem	542010	file	0.54

(continued)

	name	type	bytes	format	MB
	zip.txt	stem	145150817	file	145.15

7.2 Timing

7.2.1 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] 31338.41

## Minuten
runtime.sum / 60
#> [1] 522.3069

## Stunden
runtime.sum / 3600
#> [1] 8.705114
```

7.2.2 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
files.pdf	2024-03-13 10:11:35	29015.445	483.59
dt.annotated	2024-03-13 10:43:14	1018.008	16.97
files.txt	2024-03-13 10:17:44	343.303	5.72
zip.csv.annotated	2024-03-13 10:48:17	228.829	3.81
url.pdf	2024-03-13 02:08:25	178.208	2.97
var_lingstats	2024-03-13 10:23:27	157.484	2.62
lingstats.summary	2024-03-13 10:25:45	126.725	2.11
dt.bverwg	2024-03-13 10:20:04	75.043	1.25
zip.pdf.all	2024-03-13 10:18:47	59.199	0.99
report.codebook	2024-03-13 10:44:23	42.719	0.71

(continued)

name	time	seconds	mins
zip.txt	2024-03-13 10:20:31	26.639	0.44
zip.csv.full	2024-03-13 10:43:40	26.194	0.44
var_aktenzeichen	2024-03-13 10:20:49	12.641	0.21
report.robustness	2024-03-13 10:23:38	8.502	0.14
hashes	2024-03-13 10:48:24	6.231	0.10
csv.annotated	2024-03-13 10:44:28	4.615	0.08
var_sammlungen	2024-03-13 10:20:36	3.710	0.06
files.source	2024-03-13 12:37:58	1.370	0.02
file.vpresidents	2023-05-09 12:40:42	1.354	0.02
filenames.pdf	2024-03-13 02:08:26	0.921	0.02
zip.csv.meta	2024-03-13 10:44:28	0.324	0.01
csv.full	2024-03-13 10:23:29	0.232	0.00
var_ecli	2024-03-13 10:20:49	0.178	0.00
zip.analysis	2024-03-13 10:44:28	0.134	0.00
dt.bverwg.full	2024-03-13 10:23:29	0.120	0.00
var_vpraesi	2024-03-13 10:20:37	0.077	0.00
var_praesi	2024-03-13 10:23:27	0.074	0.00
zip.source	2024-03-13 12:38:23	0.051	0.00
csv.meta	2024-03-13 10:43:41	0.019	0.00
latexdefs	2024-03-13 02:05:27	0.017	0.00
dt.bverwg.datecleaned	2024-03-13 10:20:33	0.015	0.00
var_constants	2024-03-13 10:20:37	0.012	0.00
dt.bverwg.meta	2024-03-13 10:25:45	0.004	0.00
dt.download.final	2024-03-13 02:08:26	0.003	0.00
var_verfahrensart	2024-03-13 10:23:27	0.002	0.00
vars_additional	2024-03-13 10:23:27	0.002	0.00
vpresidents	2024-03-13 02:08:25	0.002	0.00
az.brd	2024-03-13 02:08:26	0.001	0.00
changelog	2024-03-13 01:49:30	0.001	0.00

(continued)

name	time	seconds	mins
csv.hashes	2024-03-13 10:48:24	0.001	0.00
file.presidents	2023-05-09 12:40:42	0.001	0.00
presidents	2024-03-13 02:08:25	0.001	0.00
zip.all	2024-03-13 12:38:24	0.001	0.00
file.az.brd	2023-05-09 12:40:42	0.000	0.00
file.variables.codebook	2023-05-09 12:40:42	0.000	0.00
variables.codebook	2024-03-13 02:08:26	0.000	0.00

8 Pipeline: Warnungen

```
meta <- tar_meta(fields = "warnings", complete_only = TRUE)
setDT(meta)
meta$warnings <- gsub("(\\.pdf|\\.html?|\\.txt)", "\\1 \n\n", meta$warnings)
meta <- meta[name != "dt.annotated"] # de_core_news_sm does not work correctly
    with lemmatization; warnings are not reported because they flood the document
    ; should be inspected directly

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.")

}
```

8.1 files.pdf

downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf030804U1C

HTTP status was 404 Not Found. downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf080610B1WB49.09.0.pdf

HTTP status was 404 Not Found. downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf030804U1C30.02.0.pdf

HTTP status was 404 Not Found. downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf080610B1WB49.09.0.pdf

HTTP status was 404 Not Found. downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf030804U1C30.02.0.pdf

HTTP status was 404 Not Found. downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf080610B1WB49.09.0.pdf

HTTP status was 404 Not Found. downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf030804U1C30.02.0.pdf

HTTP status was 404 Not Found. downloaded length 0 reported length 259. cannot open URL httpswww.bverwg.deentscheidungenpdf080610B1WB49.09.0.pdf

HTTP status was 404 Not Found. Missing file BVerwG_20040803_U_1_C_30_02_NA_0.pdf

Missing file BVerwG_20100608_B_1_WB_49_09_NA_0.pdf

8.2 report.codebook

Package microtype Warning Unable to apply patch footnote on input line 193.

8.3 report.robustness

The file CEBVerwG_20240313_RobustnessChecks.log is not encoded in UTF8. These lines contain invalid UTF8 characters 1190, 1195, 1205. Package microtype Warning Unable to apply patch footnote on input line 202.

9 Pipeline: 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.
```

10 Dateigrößen

10.1 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
CE-BVerwG_2024-03-13_DE_Analyse.zip	2.71
CE-BVerwG_2024-03-13_DE_CSV_Annotiert.zip	702.31
CE-BVerwG_2024-03-13_DE_CSV_Datensatz.zip	105.90
CE-BVerwG_2024-03-13_DE_CSV_Metadaten.zip	0.98
CE-BVerwG_2024-03-13_DE_PDF_Datensatz.zip	1,353.66
CE-BVerwG_2024-03-13_DE_TXT_Datensatz.zip	145.15
CE-BVerwG_2024-03-13_Source_Code.zip	0.54
CE-BVerwG_2024-03-13_Targets_Storage.zip	705.19

10.2 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
CE-BVerwG_2024-03-13_DE_CSV_Annotiert.csv	6,157.84
CE-BVerwG_2024-03-13_DE_CSV_Datensatz.csv	409.58
CE-BVerwG_2024-03-13_DE_CSV_Metadaten.csv	11.36
CE-BVerwG_2024-03-13_KryptographischeHashes.csv	0.00

10.3 PDF-Dateien (MB)

```
tar_load(files.pdf)
pdf.MB <- file.size(files.pdf) / 10^6
sum(pdf.MB)
#> [1] 1774.347
```

10.4 TXT-Dateien (MB)

```
tar_load(files.txt)
txt.MB <- file.size(files.txt) / 10^6
sum(txt.MB)
#> [1] 408.6083
```

11 Kryptographische Signaturen

11.1 Signaturen laden

```
tar_load(hashes)
```

11.2 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))
```

11.3 In Bericht anzeigen

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

index	filename
1	output/CE-BVerwG_2024-03-13_DE_PDF_Datensatz.zip
2	output/CE-BVerwG_2024-03-13_DE_TXT_Datensatz.zip
3	output/CE-BVerwG_2024-03-13_DE_CSV_Datensatz.zip
4	output/CE-BVerwG_2024-03-13_DE_CSV_Metadaten.zip
5	output/CE-BVerwG_2024-03-13_DE_Analyse.zip
6	output/CE-BVerwG_2024-03-13_Source_Code.zip
7	output/CE-BVerwG_2024-03-13_Codebook.pdf
8	output/CE-BVerwG_2024-03-13_RobustnessChecks.pdf

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

index	sha2.256
1	0ab96c2513c6a297188721273ce53ffc54e20cfe411f46b8c1fa0b5b7631d234
2	062b037c5c118acd2aa441b5762e63fbdf19c5e8f034a0fe3cace96c02672031
3	e9977ea5a8914e189b0f75e03a75397430f5cc12105905a884d30f4782d15315
4	99d90226c47393499ed63ffe6f0f819114e6822ae0005d861d88f106092ac8fb
5	dc22f3602128aa561cf889b2134ce58dc91127e65228f49a4e8404d7d413a0c9
6	2918fe2199f8f638caa4fb9ae27e9627c382624c0f5ecae150965e97ccb22c2c
7	dac4731e2ae74e82b1f87540b60e7654ace1620f9839c7316932f5d72e14a4e6
8	7d1515a68037d8230a5f67ecf36a5f416a0e4a08e1b5d84014ddaef8e2a131e3

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

index	sha3.512
1	4e0c4d7df524381de79da9f5c093377d936f8d1f5c73ef5b6de2c2e929335a97 0ebc7361cdc8ecd3fbdf6f6db99d39a80b9ef81bddaab07c0b708f1efa02e6d0
2	2a37b64ffd08385a6d6e7dcbc79cb5390007e1ae368280c765a9348577b4102e fe16a72c110482dfb8851f1bae3c4cca00e9ba8cabbeb891220274c3f8410e3d
3	4de8bf26112e747d23278644da562c2b4698cfa1623f8a538ae9c97ac7b5449e 36aca1634cebc57718174ccd4cf883d99d11a59687b2972ba2b9a861b80230f1
4	ea95f0d4bbfff89dbd371e4bbd9bc7344cd8dd5585a2f24608333bf2ad089a7b 1580755672c20663446af169d624e9a55fcd4d61d2cb7edac09ec3956f1bdaea
5	fc907da4df4289389fdb511f493c9006032eb01fb06b87ea2d52248230660a1f 3a988735d69b45afd176e247e861dfba55f4d9b0249598a6185a6d40a088b225

6	30162966662f98b73fd2cda36bba697aa2dc0d8aa24271298d37fc0bf1ec7198 f45a23da4957a946638539248e5da956b6b1c0ba3eb40e23c8f2a8889e09f020
7	0d739b64616b8a3e126ea65e29f3413ed30487dbe0d8b4d45066ed22cbf165f6 26429e8a693a2a97f06eed9fdd2ee9da6d9a311f656b03c81aff6f6e40daeaac
8	ab3ac413499bc0b03a682ddab28b170033bbda69da155c2fd480723f81b0da2c 9ae98e2ae16a55c6fd8e583fb6f5100e2b500425077b87b13a8222c7040dbd9a

12 Changelog

12.1 Version 2024-03-13

- Vollständige Aktualisierung der Daten
- Die Pipeline mit allen Zwischenergebnissen wird nun automatisch in "output/" archiviert
- Gruppierung von Dateinamen der Diagramme nach Typ
- Vereinfachung der Repository-Struktur
- Anpassung von Docker Compose an Debian 11
- Update GPG Public Key in Repository

12.2 Version 2023-03-21

- Vollständige Aktualisierung der Daten
- Gesamte Laufzeitumgebung mit Docker versionskontrolliert
- 4 neue Variablen: BVerwGE, Leitsatz, Fachpresse und Übersetzung (jeweils binär)
- Linguistisch annotierte Variante ist wieder verfügbar
- Aktenzeichen aus dem Eingangszeitraum 2000 bis 2009 nun korrekt mit führender Null formatiert (z.B. 1 BvR 44/02 statt 1 BvR 44/2)
- Aktenzeichen in Verzögerungsverfahren nun korrekt formatiert
- Aktenzeichen verwenden nun richigerweise den Punkt als Trenner (statt den Schrägstrich wie bei anderen Gerichten)
- Berücksichtigung von Präsident Korbmacher und Vize-Präsidentin Rublack in den jeweiligen Variablen
- Update des Run-Skripts und des Delete-Skripts
- Proto-Package Mono-Repo entfernt, alle Funktionen nun fest projektbasiert versionskontrolliert
- Vereinfachung der Konfigurations-Datei
- Neue Funktion für automatischen clean run (Löschung aller Zwischenergebnisse)
- Update der Download-Funktion
- Überflüssige Warnung in f.future_lingsummarize-Funktion entfernt
- Alle Roh-Dateien werden nun im Unterordner "files" gespeichert
- Vom BVerwG nicht veröffentlichte Entscheidungstypen im Codebook genauer beschrieben
- Verbesserte Formatierung von Profiling, Warnungen und Fehlermeldungen im Compilation Report
- Zusätzliche Unit-Tests
- Verbesserung des Robustness Check Reports
- README im Hinblick auf Docker überarbeitet

12.3 Version 2022-08-07

- Vollständige Aktualisierung der Daten
- Neuentwurf des Source Codes im *targets* framework
- Einführung von separater Konfigurations-Datei
- Strenge Versionskontrolle aller R packages mit *renv*
- Variante mit linguistischen Annotationen ist temporär nicht mehr verfügbar
- Neue Visualisierung der gesamten Daten-Pipeline
- Vielzahl zusätzlicher Unit Tests (inklusive type safety) für wichtige Funktionen

12.4 Version 2021-10-19

- Vollständige Aktualisierung der Daten
- Neue Variante mit linguistische Annotationen
- Neue Variable für Lizenz
- Strenge Kontrolle und semantische Sortierung aller Variablen-Namen
- Source Code des Codebooks deutlich vereinfacht (insbes. Diagramme und Changelog)
- Erweiterung der Dokumentation
- In den linguistischen Kennzahlen werden jetzt auch die Anzahl Typen bezogen auf den Gesamtdatensatz berechnet
- Standardisierung der Diagramme auf 6:9 Zoll (Breite/Höhe)

12.5 Version 2021-04-15

- Vollständige Aktualisierung der Daten
- Veröffentlichung des vollständigen Source Codes
- Deutliche Erweiterung des inhaltlichen Umfangs des Codebooks
- Einführung der vollautomatischen Erstellung von Datensatz und Codebook
- Einführung von Compilation Reports um den Erstellungsprozess exakt zu dokumentieren
- Einführung von Variablen für Versionsnummer, Concept DOI, Version DOI, ECLI, Präsident:in, Vize-Präsident:in, Verfahrensart und linguistische Kennzahlen (Zeichen, Tokens, Typen, Sätze)
- Zusammenfügung von über Zeilengrenzen getrennten Wörtern
- Automatisierung und Erweiterung der Qualitätskontrolle
- Einführung von Diagrammen zur Visualisierung von Prüfergebnissen
- Einführung kryptographischer Signaturen
- Variable »Suffix« in »kollision« umbenannt.
- Variable »Ordinalzahl« in »eingangsnummer« umbenannt.
- Variable »Entscheidungsart« in »entscheidung_typ« umbenannt.
- Alle übrigen Variablen sind nun in Kleinschreibung und Snake Case gehalten

12.6 Version 2020-06-23

- Erstveröffentlichung

13 Abschluss

```
## Datumsstempel
print(datestamp)
#> [1] "2024-03-13"

## Datum und Uhrzeit (Anfang)
print(begin.script)
#> [1] "2024-03-13 12:38:20 CET"


## Datum und Uhrzeit (Ende)
end.script <- Sys.time()
print(end.script)
#> [1] "2024-03-13 12:38:44 CET"


## Laufzeit des gesamten Skriptes
print(end.script - begin.script)
#> Time difference of 24.52974 secs
```

14 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 4.2.2 (2022-10-31)
#> Platform: x86_64-pc-linux-gnu (64-bit)
#> Running under: Ubuntu 22.04.2 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
#>
#> locale:
#>  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
#>  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
#>  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
#>  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#>  [1] ggraph_2.1.0      ggplot2_3.4.1     igraph_1.4.1      kableExtra_1.3.4
#>  [5] knitr_1.42        quanteda_3.2.4    data.table_1.14.8 future_1.32.0
#>  [9] RcppTOML_0.2.2    tarchetypes_0.7.5 targets_0.14.3
#>
#> loaded via a namespace (and not attached):
#>  [1] viridis_0.6.2       httr_1.4.5          tidyr_1.3.0
#>  [4] tidygraph_1.2.3     viridisLite_0.4.1   RcppParallel_5.1.7
#>  [7] highr_0.10          future.callr_0.8.1  base64url_1.4
#> [10] renv_0.17.0         yaml_2.3.7          ggrepel_0.9.3
#> [13] globals_0.16.2      pillar_1.8.1        backports_1.4.1
#> [16] lattice_0.20-45     glue_1.6.2          digest_0.6.31
#> [19] polyclip_1.10-4     rvest_1.0.3         stringfish_0.15.7
#> [22] colorspace_2.1-0    htmltools_0.5.4     Matrix_1.5-1
#> [25] pkgconfig_2.0.3     listenv_0.9.0       purrr_1.0.1
#> [28] scales_1.2.1        webshot_0.5.4       processx_3.8.0
#> [31] svglite_2.1.1       tweenr_2.0.2        RApiSerialize_0.1.2
#> [34] ggforce_0.4.1       tibble_3.2.0        generics_0.1.3
#> [37] farver_2.1.1        withr_2.5.0         furrr_0.3.1
#> [40] cli_3.6.0           magrittr_2.0.3      evaluate_0.20
#> [43] ps_1.7.2            stopwords_2.3       fs_1.6.1
#> [46] fansi_1.0.4         parallelly_1.34.0   MASS_7.3-58.1
#> [49] xml2_1.3.3          tools_4.2.2         lifecycle_1.0.3
#> [52] stringr_1.5.0       munsell_0.5.0       callr_3.7.3
#> [55] compiler_4.2.2      qs_0.25.5           systemfonts_1.0.4
#> [58] rlang_1.0.6         grid_4.2.2          rstudioapi_0.14
#> [61] labeling_0.4.2      rmarkdown_2.20      gtable_0.3.1
#> [64] codetools_0.2-18    graphlayouts_0.8.4  R6_2.5.1
#> [67] gridExtra_2.3       dplyr_1.1.0         fastmap_1.1.1
#> [70] utf8_1.2.3          fastmatch_1.1-3     stringi_1.7.12
```

```
#> [73] parallel_4.2.2      Rcpp_1.0.10         vctrs_0.5.2
#> [76] tidyselect_1.2.0    xfun_0.37
```

Literaturverzeichnis

Allaire, JJ, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, Winston Chang, and Richard Iannone. 2023. *Rmarkdown: Dynamic Documents for R.*

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.

———. 2022. *Future.apply: Apply Function to Elements in Parallel Using Futures.*

———. 2023. *Future: Unified Parallel and Distributed Processing in R for Everyone.*

Benoit, Kenneth, and Akitaka Matsuo. 2020. *Spacyr: Wrapper to the spaCy 'Nlp' Library.* https://spacyr.quanteda.io.

Benoit, Kenneth, and Adam Obeng. 2021. *Readtext: Import and Handling for Plain and Formatted Text Files.* https://github.com/quanteda/readtext.

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. 2022. *Quanteda: Quantitative Analysis of Textual Data.* https://quanteda.io.

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

Csárdi, Gábor, Kuba Podgórski, and Rich Geldreich. 2022. *Zip: Cross-Platform Zip Compression.* https://github.com/r-lib/zip#readme.

Dowle, Matt, and Arun Srinivasan. 2023. *Data.table: Extension of 'Data.frame'.*

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.*

file., See AUTHORS. 2023. *Igraph: Network Analysis and Visualization.*

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. 2023. *Stringi: Fast and Portable Character String Processing Facilities.*

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.

———. 2023a. *Tarchetypes: Archetypes for Targets.*

———. 2023b. *Targets: Dynamic Function-Oriented Make-Like Declarative Pipelines.*

Ooms, Jeroen. 2023a. *Magick: Advanced Graphics and Image-Processing in R.*

———. 2023b. *Pdftools: Text Extraction, Rendering and Converting of Pdf Documents.*

Pedersen, Thomas Lin. 2022. *Ggraph: An Implementation of Grammar of Graphics for Graphs and Networks.*

Ushey, Kevin. 2023. *Renv: Project Environments.* https://rstudio.github.io/renv/.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York. https://ggplot2.tidyverse.org.

———. 2022. *Rvest: Easily Harvest (Scrape) Web Pages.*

———. 2023. *Httr: Tools for Working with Urls and Http.*

Wickham, Hadley, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. 2023. *Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics.*

Wickham, Hadley, and Dana Seidel. 2022. *Scales: Scale Functions for Visualization.*

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/.

———. 2023. *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. 2021. *KableExtra: Construct Complex Table with Kable and Pipe Syntax.*