Package 'writeAlizer'

May 8, 2026

Type Package

Title Generate Predicted Writing Quality Scores

Version 1.7.3

Date 2026-2-16

Description Imports variables from 'ReaderBench' (Dascalu et al., 2018)<doi:10.1007/978-3-319-66610-5_48>, 'Coh-Metrix' (McNamara et al., 2014)<doi:10.1017/CBO9780511894664>, and/or 'GAMET' (Crossley et al., 2019) <doi:10.17239/jowr-2019.11.02.01> output files; downloads predictive scoring models described in Mercer & Cannon (2022)<doi:10.31244/jero.2022.01.03> and Mercer et al.(2021)<doi:10.1177/0829573520987753>; and generates predicted writing quality and curriculum-based measurement (McMaster & Espin, 2007)<doi:10.1177/00224669070410020301> scores.

License MIT + file LICENSE

URL https://github.com/shmercer/writeAlizer/, https://shmercer.github.io/writeAlizer/

BugReports https://github.com/shmercer/writeAlizer/issues

Depends R (>= 2.10)

Imports caret, cli, digest, dplyr, glue, magrittr, rlang (>= 1.1.0), stats, tidyselect

Suggests caretEnsemble, Cubist, curl, earth, gbm, glmnet, kernlab, knitr, pls, randomForest, rmarkdown, testthat (>= 3.1.0), withr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

VignetteBuilder knitr

NeedsCompilation no

Author Sterett H. Mercer [aut, cre] (ORCID: <https://orcid.org/0000-0002-7940-4221>)

Maintainer Sterett H. Mercer <sterett.mercer@ubc.ca>

Repository CRAN

Date/Publication 2026-02-17 06:10:13 UTC

Contents

`writeAlizer-package` *writeAlizer: An R Package to Generate Automated Writing Quality and Curriculum-Based Measurement (CBM) Scores.*

Description

Package-level documentation for writeAlizer.

Details

Detailed documentation on writeAlizer is available in the GitHub README file and wiki.

The writeAlizer R package (a) imports ReaderBench, Coh-Metrix, and GAMET output files into R, and (b) uses research-developed scoring models to generate predicted writing quality scores or Correct Word Sequences and Correct Minus Incorrect Word Sequences scores from those files.

The writeAlizer package includes functions to do two types of tasks: (1) importing ReaderBench, Coh-Metrix, and/or GAMET output files into R; and (2) generating predicted quality scores using the imported output files. There are also additional functions to help with (3) installation of package dependencies and (4) cache management.

1. Import output files

- `import_rb`
- `import_coh`
- `import_gamet`
- `import_merge_gamet_rb`

2. Generate predicted quality scores

- predict_quality

3. Identify necessary packages

- model_deps

4. Cache management

- wa_cache_dir
- wa_cache_clear

Author(s)

Maintainer: Sterett H. Mercer <sterett.mercer@ubc.ca> (ORCID)

See Also

Useful links:

- https://github.com/shmercer/writeAlizer/
- https://shmercer.github.io/writeAlizer/
- Report bugs at https://github.com/shmercer/writeAlizer/issues

---

`import_coh` *Import a Coh-Metrix output file (.csv) into R.*

Description

Import a Coh-Metrix output file (.csv) into R.

Usage

```
import_coh(path)
```

Arguments

`path`	A string giving the path and filename to import.

Value

A base `data.frame` with one row per record and the following columns:

- `ID` (`character`): unique identifier of the text/essay.
- One column per retained Coh-Metrix feature, kept by original feature name (`numeric`). Feature names mirror the Coh-Metrix output variables.

The object has class `data.frame` (or `tibble` if converted by the user).

See Also

predict_quality

Examples

```
# Example with package sample data
file_path <- system.file("extdata", "sample_coh.csv", package = "writeAlizer")
coh_file  <- import_coh(file_path)
head(coh_file)
```

`import_gamet` *Import a GAMET output file into R.*

Description

Import a GAMET output file into R.

Usage

```
import_gamet(path)
```

Arguments

`path`	A string giving the path and filename to import.

Value

A base `data.frame` with one row per record and the following columns:

- `ID` (`character`): unique identifier of the text/essay.
- One column per retained GAMET error/category variable (`numeric`; typically counts or rates). Column names follow the GAMET output variable names.

The object has class `data.frame` (or `tibble` if converted by the user).

See Also

predict_quality

Examples

```
# Example with package sample data
file_path   <- system.file("extdata", "sample_gamet.csv", package = "writeAlizer")
gamet_file  <- import_gamet(file_path)
head(gamet_file)
```

`import_merge_gamet_rb` *Import a ReaderBench output file (.csv) and GAMET output file (.csv), and merge the two files on ID.*

Description

Import a ReaderBench output file (.csv) and GAMET output file (.csv), and merge the two files on ID.

Usage

```
import_merge_gamet_rb(rb_path, gamet_path)
```

Arguments

`rb_path`	A string giving the path and ReaderBench filename to import.
`gamet_path`	A string giving the path and GAMET filename to import.

Value

A base `data.frame` created by joining the ReaderBench and GAMET tables by `ID`, with one row per matched ID and the following columns:

- `ID` (`character`): identifier present in both sources.
- All retained ReaderBench feature columns (`numeric`).
- All retained GAMET error/category columns (`numeric`).

By default, only IDs present in both inputs are kept (inner join). If a feature name appears in both sources, standard merge suffixes (e.g., `.x`/`.y`) may be applied by the join implementation. The object has class `data.frame` (or `tibble` if converted by the user).

See Also

`predict_quality`

Examples

```
# Example with package sample data
rb_path   <- system.file("extdata", "sample_rb.csv", package = "writeAlizer")
gam_path  <- system.file("extdata", "sample_gamet.csv", package = "writeAlizer")
rb_gam    <- import_merge_gamet_rb(rb_path, gam_path)
head(rb_gam)
```

`import_rb` *Import a ReaderBench output file (.csv) into R.*

Description

When available, the function reads the header of the packaged sample (`inst/extdata/sample_rb.csv`) and keeps the first 404 columns by NAME (plus the `File.name/ID` column), excluding any columns with names appearing after position 404 in that header. If the sample is unavailable, it falls back to keeping the first 404 columns by position.

Usage

```
import_rb(path)
```

Arguments

`path`	A string giving the path and filename to import.

Value

A base `data.frame` with one row per record and the following columns:

- `ID` (`character`): unique identifier of the text/essay.
- One column per retained ReaderBench feature, kept by original feature name (`numeric`). Feature names mirror the ReaderBench output variables.

The object has class `data.frame` (or `tibble` if converted by the user).

See Also

`predict_quality`

Examples

```
# Fast, runnable example with package sample data
file_path <- system.file("extdata", "sample_rb.csv", package = "writeAlizer")
rb_file   <- import_rb(file_path)
head(rb_file)
```

`keep_stem_before_txt` *Extract the filename stem before ".txt"*

Description

Removes any directory path and optional '.txt' extension from filenames or file paths. This function standardizes text identifiers across Coh-Metrix, GAMET, and other text analysis outputs that may include full paths or extensions in their ID fields.

Usage

```
keep_stem_before_txt(x)
```

Arguments

`x`	A character vector (or coercible) containing file paths or filenames. Elements may or may not include a '.txt' suffix or any directory path.

Details

The function handles both forward ('/') and backward ('\') slashes in file paths. If a value has no path and/or no '.txt' suffix, it is returned unchanged (aside from coercion to character).

Value

A character vector where each element is reduced to the final path component, with any trailing '.txt' (case-insensitive) removed. 'NA' values are preserved as 'NA_character_'.

Examples

```
keep_stem_before_txt(c(
  "C:/data/3401.txt",
  "E:\\\\samples\\\\1002.TXT",
  "plain_id",
  NA
))
#> [1] "3401" "1002" "plain_id" NA
```

`model_deps` *Report optional model dependencies (no installation performed)*

Description

Discovers package dependencies for model fitting from the package 'Suggests' field. This function **never installs** packages. It reports which packages are required and which are currently missing, and prints a ready-to-copy command you can run to install the missing ones manually.

Usage

```
model_deps()
```

Details

You can add or override discovered packages for testing or CI with 'options(writeAlizer.required_pkgs = c("pkgA", "pkgB (>= 1.2.3)"))'. Any version qualifiers you include are preserved in the 'required' output, but stripped for the availability check in 'missing'.

Value

A named list:

required Character vector of discovered package tokens (may include version qualifiers), e.g. `c("glmnet (>= 4.1)", "ranger")`. This is the union of the package *Suggests* field and the optional `writeAlizer.required_pkgs` override.

missing Character vector of base package names that are not installed, e.g. `c("glmnet", "ranger")`.

The function also emits a message. If nothing is missing, it reports that all required packages are installed. Otherwise, it lists the missing packages and prints a copy-paste `install.packages()` command.

Examples

```
md <- model_deps()
md$missing
```

predict_quality *Predict writing quality*

Description

Run the specified model(s) on preprocessed data and return predictions. Apply scoring models to ReaderBench, Coh-Metrix, and/or GAMET files. Holistic writing quality can be generated from ReaderBench (model = 'rb_mod3all') or Coh-Metrix files (model = 'coh_mod3all'). Also, Total Words Written, Words Spelled Correctly, Correct Word Sequences, and Correct Minus Incorrect Word Sequences can be generated from a GAMET file (model = 'gamet_cws1').

Usage

```
predict_quality(model, data)
```

Arguments

model	A string telling which scoring model to use. Options are: 'rb_mod1', 'rb_mod2', 'rb_mod3narr', 'rb_mod3exp', 'rb_mod3per', or 'rb_mod3all', for ReaderBench files to generate holistic quality, 'coh_mod1', 'coh_mod2', 'coh_mod3narr', 'coh_mod3exp', 'coh_mod3per', or 'coh_mod3all' for Coh-Metrix files to generate holistic quality, and 'gamet_cws1' to generate Total Words Written (TWW), Words Spelled Correctly (WSC), Correct Word Sequences (CWS) and Correct Minus Incorrect Word Sequences (CIWS) scores from a GAMET file.
data	Data frame returned by import_gamet, import_coh, or import_rb.

Details

Offline/examples: Examples use a built-in 'example' model seeded in a temporary directory via `writeAlizer::wa_seed_example_models("example")`, so no downloads are attempted and checks stay fast. The temporary files created for the example are cleaned up at the end of the `\examples{}`.

Value

A `data.frame` with `ID` and one column per sub-model prediction. If multiple sub-models are used and all predictions are numeric, an aggregate column named `pred_<model>_mean` is added (except for "gamet_cws1").

See Also

import_rb, import_coh, import_gamet

Examples

```
# Offline, CRAN-safe example using a tiny seeded model
if (requireNamespace("withr", quietly = TRUE)) {
  withr::local_options(writeAlizer.offline = TRUE)
  tmp <- withr::local_tempdir()
  withr::local_options(writeAlizer.mock_dir = tmp)

  # Seed the example artifacts into the temp dir and point the loader there
  writeAlizer::wa_seed_example_models("example", dir = tmp)

  coh <- import_coh(system.file("extdata", "sample_coh.csv", package = "writeAlizer"))
  out <- predict_quality("example", coh)
  head(out)
} else {
  # Fallback without 'withr' (still CRAN-safe)
  old <- options(writeAlizer.offline = TRUE)
  on.exit(options(old), add = TRUE)
  ex_dir <- writeAlizer::wa_seed_example_models("example", dir = tempdir())
  old2 <- options(writeAlizer.mock_dir = ex_dir)
  on.exit(options(old2), add = TRUE)

  coh <- import_coh(system.file("extdata", "sample_coh.csv", package = "writeAlizer"))
  out <- predict_quality("example", coh)
  head(out)
}

# Longer, networked demos
## Not run:
if (!isTRUE(getOption("writeAlizer.offline", FALSE))) {
  rb <- import_rb(system.file("extdata", "sample_rb.csv", package = "writeAlizer"))
  print(head(predict_quality("rb_mod3all", rb)))

  coh <- import_coh(system.file("extdata", "sample_coh.csv", package = "writeAlizer"))
  print(head(predict_quality("coh_mod3all", coh)))

 gam <- import_gamet(system.file("extdata", "sample_gamet.csv", package = "writeAlizer"))
  print(head(predict_quality("gamet_cws1", gam)))
}

## End(Not run)
```

`preprocess` *Pre-process data*

Description

Pre-process Coh-Metrix and ReaderBench data files before applying predictive models. Uses the artifact registry to load the correct variable lists and applies centering and scaling per sub-model, preserving the original behavior by model key.

Usage

```
preprocess(model, data)
```

Arguments

model	Character scalar. Which scoring model to use. Supported values include: Reader-Bench: 'rb_mod1','rb_mod2','rb_mod3narr','rb_mod3exp','rb_mod3per','rb_mod3all', 'rb_mod3narr_v2','rb_mod3exp_v2','rb_mod3per_v2','rb_mod3all_v2'; Coh-Metrix: 'coh_mod1','coh_mod2','coh_mod3narr','coh_mod3exp','coh_mod3per','coh_mod3all'; GAMET: 'gamet_cws1'. Legacy keys for RB mod3 (non-v2) are mapped to their v2 equivalents internally.
data	A data.frame produced by import_rb, import_coh, or import_gamet, with an ID column and the expected feature columns.

Details

Offline/examples: Examples use a built-in 'example' model seeded in a temporary directory via writeAlizer::wa_seed_example_models("example"), so no downloads are attempted and checks stay fast.

Value

A list of pre-processed data frames, one per sub-model. For models with no varlists (e.g., 'rb_mod1','coh_mod1'), returns six copies of the input data. For 'gamet_cws1', returns two copies (CWS/CIWS). For 1-part/3-part models, returns a list of length 1/3 with centered & scaled features plus the ID column.

Examples

```
# Minimal, offline example using the built-in 'example' model (no downloads)
rb_path <- system.file("extdata", "sample_rb.csv", package = "writeAlizer")
rb <- import_rb(rb_path)

pp <- preprocess("example", rb)
length(pp); lapply(pp, nrow)
```

---

wa_cache_clear *Clear writeAlizer's user cache*

Description

Deletes all files under wa_cache_dir(). If ask = TRUE *and* in an interactive session, a short preview (item count, total size, and up to 10 sample paths) is printed before asking for confirmation.

Usage

```
wa_cache_clear(ask = interactive(), preview = TRUE)
```

Arguments

ask	Logical; if TRUE and interactive, ask for confirmation.
preview	Logical; if TRUE and ask is TRUE, show a brief listing/size summary before asking.

Value

Invisibly returns TRUE if the cache was cleared (or already absent), FALSE if the user declined or deletion failed.

See Also

wa_cache_dir

Examples

```
# Safe demo: redirect cache to tempdir(), create a file, then clear it
```

---

wa_cache_dir *Path to writeAlizer's user cache*

Description

Returns the directory used to store cached model artifacts. By default this is a platform-appropriate user cache path from `tools::R_user_dir("writeAlizer","cache")`. If the option `writeAlizer.cache_dir` is set to a non-empty string, that location is used instead. This makes it easy to redirect the cache during tests or examples (e.g., to `tempdir()`).

Usage

```
wa_cache_dir()
```

Value

Character scalar path.

See Also

wa_cache_clear

Examples

```
# Inspect the cache directory (no side effects)
wa_cache_dir()
```

wa_download *Download and cache an artifact (graceful offline behavior)*

Description

Public helper to fetch an artifact into the user cache. This function delegates to the internal downloader used by the package at runtime, so it benefits from the same behavior:

Usage

```
wa_download(file, url, sha256 = NULL, quiet = TRUE)

download(file, url) # deprecated
```

Arguments

file	Character scalar; filename to use in the cache (e.g., '"rb_mod1a.rda"').
url	Character scalar; source URL. May be a 'file://' URL for local testing.
sha256	Optional 64-hex SHA-256 checksum for verification. If provided, the cached file must match it (or a re-download is attempted).
quiet	Logical; if 'TRUE', suppresses download progress messages.

Details

- Respects `options(writeAlizer.mock_dir)` to load local mock copies (useful for tests/examples and offline runs). - Fails *gracefully* with a clear, informative message when Internet resources are unavailable or have changed (per CRAN policy). - Verifies an optional SHA-256 checksum and re-downloads or errors if it does not match.

Value

A character scalar: the absolute path to the cached file.

Examples

```
# Offline-friendly example using a local source (no network) — CRAN-safe
if (requireNamespace("withr", quietly = TRUE)) {
  withr::local_options(writeAlizer.mock_dir = NULL, writeAlizer.offline = FALSE)
}

src <- tempfile(fileext = ".bin")
writeBin(as.raw(1:10), src)
url <- paste0("file:///", normalizePath(src, winslash = "/"))

# Deterministic and quiet: checksum + cache reuse
sha <- digest::digest(src, algo = "sha256", file = TRUE)
dest <- wa_download("example.bin", url = url, sha256 = sha, quiet = TRUE)
file.exists(dest)
```

```
# Using a mock directory to avoid network access:
# options(writeAlizer.mock_dir = "/path/to/local/artifacts")
# dest <- wa_download("rb_mod1a.rda", url = "https://example.com/rb_mod1a.rda")
```

`wa_seed_example_models` *Seed example model files in a temporary directory*

Description

This helper writes a minimal model file to a subdirectory of ‘dir‘ (default: ‘tempdir()‘), and sets the option ‘writeAlizer.mock_dir‘ to that location so examples can run without downloads or network access.

Usage

```
wa_seed_example_models(model = c("example"), dir = tempdir())
```

Arguments

`model`	Character scalar. Only ‘"example"‘ is currently supported.
`dir`	Directory in which to create the example model (default: ‘tempdir()‘).

Details

Creates an ultra-tiny model artifact used in examples and points the package loader to it via a temporary option.

- Writes only under ‘tempdir()‘ and returns the created path. - Sets ‘options(writeAlizer.mock_dir = <path>)‘; callers should restore prior options when appropriate (see Examples).

Value

(Invisibly) the path to the created example model directory.

Examples

```
old <- getOption("writeAlizer.mock_dir")
on.exit(options(writeAlizer.mock_dir = old), add = TRUE)

ex <- wa_seed_example_models(dir = tempdir())
# Use the package normally here; the loader will find `ex`
# ...
unlink(ex, recursive = TRUE, force = TRUE)
```

Index