Package 'oolong'

April 15, 2024

Title Create Validation Tests for Automated Content Analysis

Version 0.6.1

Description Intended to create standard human-in-the-loop validity tests for typical automated content analysis such as topic modeling and dictionary-based methods. This package offers a standard workflow with functions to prepare, administer and evaluate a human-in-the-loop validity test. This package provides functions for validating topic models using word intrusion, topic intrusion (Chang et al. 2009, <https://papers.nips.cc/paper/3700-reading-tea-leaves-how-humans-interpret-topic-models>) and word set intrusion (Ying et al. 2021) <doi:10.1017/pan.2021.33> tests. This package also provides functions for generating gold-standard data which are useful for validating dictionary-based methods. The default settings of all generated tests match those suggested in Chang et al. (2009) and Song et al. (2020) <doi:10.1080/10584609.2020.1723752>.

License LGPL (>= 2.1)

Encoding UTF-8

URL https://gesistsa.github.io/oolong/, https://github.com/gesistsa/oolong

LazyData true

Depends R (>= 3.5.0)

Imports seededlda, purrr, tibble, shiny, digest, R6, quanteda (>= 3.0.0), irr, ggplot2, cowplot, cli, stats, utils

RoxygenNote 7.3.1

Suggests keyATM (>= 0.2.2), testthat (>= 3.0.2), text2vec (>= 0.6), BTM, dplyr, topicmodels, stm, covr, stringr, knitr, rmarkdown, fs, quanteda.textmodels, shinytest2

BugReports https://github.com/gesistsa/oolong/issues

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website gesistsa/tsatemplate

NeedsCompilation no

Author Chung-hong Chan [aut, cre] (<https://orcid.org/0000-0002-6232-7530>), Marius Sältzer [aut] (<https://orcid.org/0000-0002-8604-4666>)

Maintainer Chung-hong Chan <chainsawtiney@gmail.com>

Repository CRAN

Date/Publication 2024-04-15 12:40:04 UTC

R topics documented:

`abstracts` *Abstracts of communication journals dataset*

Description

This is a random sample of all abstracts of papers published in high-impact communication journals from 2000 to 2017. abstracts_dictionary is a list of terms that can be used for semisupervised techniques such as keyATM.

Usage

```
abstracts

abstracts_dfm

abstracts_dictionary
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2500 rows and 1 columns.

An object of class `dfm` with 2500 rows and 3998 columns.

An object of class `list` of length 10.

References

Chan, C-h, & Grill, C. (2020). [The Highs in Communication Research: Research Topics With High Supply, High Popularity, and High Prestige in High-Impact Journals.](https://doi.org/10.1177/0093650220944790) Communication Research.

`abstracts_seededlda` *Topic models trained with the abstracts dataset.*

Description

These are topic models trained with different topic model packages.

Usage

```
abstracts_seededlda

abstracts_btm
```

Format

An object of class `textmodel_lda` (inherits from `textmodel`, `list`) of length 10.

An object of class `BTM` of length 9.

`afinn` *AFINN dictionary*

Description

This is the AFINN sentiment dictionary in quanteda::dictionary format.

Usage

```
afinn
```

Format

An object of class `dictionary2` of length 11.

References

Nielsen, F. Å. (2011). A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. arXiv preprint arXiv:1103.2903.

check_oolong *Check whether the oolong needs to be updated*

Description

This function raises an error when the input oolong object needs to be updated. Oolong objects generated with an old version of oolong need to be updated to use the functionalities from the recent versions of oolong.

Usage

```
check_oolong(oolong, verbose = FALSE)
```

Arguments

oolong	an oolong object to be checked
verbose,	logical, display messages

Value

Nothing

Author(s)

Chung-hong Chan

clone_oolong *Clone an oolong object*

Description

Clone a new oolong object. The oolong must not be locked and ever coded.

Usage

```
clone_oolong(oolong, userid = NA)
```

Arguments

oolong	an oolong object.
userid	a character string to denote the name of the coder

Value

an oolong object

Author(s)

Chung-hong Chan

`create_oolong` *Generate an oolong test*

Description

`create_oolong` generates an oolong test object that can either be used for validating a topic model or for creating ground truth (gold standard) of a text corpus. `wi` (word intrusion test), `ti` (topic intrusion test), `witi` (word and topic intrusion tests), `wsi` (word set intrusion test) and `gs` are handy wrappers to `create_oolong`. It is recommended to use these wrappers instead of `create_oolong`.

Usage

```
create_oolong(
  input_model = NULL,
  input_corpus = NULL,
  n_top_terms = 5,
  bottom_terms_percentile = 0.6,
  exact_n = NULL,
  frac = 0.01,
  n_top_topics = 3,
  n_topiclabel_words = 8,
  use_frex_words = FALSE,
  frexweight = 0.5,
  input_dfm = NULL,
  construct = "positive",
  btm_dataframe = NULL,
  n_correct_ws = 3,
  wsi_n_top_terms = 20,
  userid = NA,
  type = "witi",
  lambda = 1,
  difficulty = NULL
)

wi(
  input_model = NULL,
  userid = NA,
  n_top_terms = 5,
  bottom_terms_percentile = 0.6,
  frexweight = 0.5,
  use_frex_words = FALSE,
  lambda = 1,
  difficulty = NULL
```

```
)

witi(
  input_model = NULL,
  input_corpus = NULL,
  userid = NA,
  n_top_terms = 5,
  bottom_terms_percentile = 0.6,
  exact_n = NULL,
  frac = 0.01,
  n_top_topics = 3,
  n_topiclabel_words = 8,
  frexweight = 0.5,
  use_frex_words = FALSE,
  input_dfm = NULL,
  btm_dataframe = NULL,
  lambda = 1,
  difficulty = NULL
)

ti(
  input_model = NULL,
  input_corpus = NULL,
  userid = NA,
  exact_n = NULL,
  frac = 0.01,
  n_top_topics = 3,
  n_topiclabel_words = 8,
  frexweight = 0.5,
  use_frex_words = FALSE,
  input_dfm = NULL,
  btm_dataframe = NULL,
  lambda = 1,
  difficulty = NULL
)

wsi(
  input_model = NULL,
  userid = NA,
  n_topiclabel_words = 4,
  n_correct_ws = 3,
  wsi_n_top_terms = 20,
  frexweight = 0.5,
  use_frex_words = FALSE,
  lambda = 1,
  difficulty = NULL
)
```

```
gs(
  input_corpus = NULL,
  userid = NA,
  construct = "positive",
  exact_n = NULL,
  frac = 0.01
)
```

Arguments

input_model	(wi, ti, witi, wsi) a STM, WarpLDA, topicmodels, KeyATM, seededlda, textmodel_nb, or BTM object; if it is NULL, create_oolong assumes that you want to create gold standard.
input_corpus	(wi, ti, witi, wsi, gs) if input_model is not null, it should be the corpus (character vector or quanteda::corpus object) to generate the model object. If input_model and input_corpus are not NULL, topic intrusion test cases are generated. If input_model is a BTM object, this argument is ignored. If input_model is null, it generates gold standard test cases.
n_top_terms	(wi, witi) integer, number of top topic words to be included in the candidates of word intrusion test.
bottom_terms_percentile	(wi, witi) double, a term is considered to be an word intruder when its theta less than the percentile of this theta, must be within the range of 0 to 1
exact_n	(ti, witi, gs) integer, number of topic intrusion test cases to generate, ignore if frac is not NULL
frac	(ti, witi, gs) double, fraction of test cases to be generated from the corpus
n_top_topics	(wi, witi) integer, number of most relevant topics to be shown alongside the intruder topic
n_topiclabel_words	(witi, ti, wsi) integer, number of topic words to be shown as the topic ("ti" and "witi") / word set ("wsi") label
use_frex_words	(wi, witi, ti, wsi) logical, for a STM object, use FREX words if TRUE, use PROB words if FALSE
frexweight	(wi, witi, ti, wsi) double, adjust the 'frexweight' for STM (see [stm::labelTopics()]), no effect for STM if use_frex_words is FALSE
input_dfm	(wi, witi, ti, wsi) a dfm object used for training the input_model, if input_model is a WarpLDA object
construct	(gs) string, an adjective to describe the construct you want your coders to code the the gold standard test cases
btm_dataframe	(witi, ti) dataframe used for training the input_model, if input_model is a BTM object
n_correct_ws	(wsi) number of word sets to be shown alongside the intruder word set
wsi_n_top_terms	(wsi) number of top topic words from each topic to be randomized selected as the word set label

`userid`	a character string to denote the name of the coder. Default to NA (no userid); not recommended
`type`	(create_oolong) a character string to denote what you want to create. "wi": word intrusion test; "ti": topic intrusion test; "witi": both word intrusion test and topic intrusion test; "gs": gold standard generation
`lambda`	(wi, witi, ti, wsi) double, adjust the 'lambda' for WarpLDA (see [text2vec::LatentDirichletAllocation()])
`difficulty`	(wi, witi, ti, wsi) double, deprecated, for backward compatibility

Value

an oolong test object.

Usage

Use `wi`, `ti`, `witi`, `wsi` or `gs` to generate an oolong test of your choice. It is recommended to supply also `userid` (current coder). The names of the tests (word intrusion test and topic intrusion test) follow Chang et al (2009). In Ying et al. (2021), topic intrusion test is named "T8WSI" (Top 8 Word Set Intrusion). Word set intrusion test in this package is actually the "R4WSI" (Random 4 Word Set Intrusion) in Ying et al. The default settings of `wi`, `witi`, and `ti` follow Chang et al (2009), e.g. `n_top_terms` = 5; instead of `n_top_terms` = 4 as in Ying et al. The default setting of `wsi` follows Ying et al., e.g. `n_topiclabel_words` = 4. As suggested by Song et al. (2020), 1

About create_oolong

Because `create_oolong` is not intuitive to use, it is no longer recommended to use `create_oolong` to generate oolong test. `create_oolong` is retained only for backward compatibility purposes. This function generates an oolong test object based on `input_model` and `input_corpus`. If `input_model` is not NULL, it generates oolong test for a topic model (tm). If `input_model` is NULL but input_corpus is not NULL, it generates oolong test for generating gold standard (gs).

Methods

An oolong object, depends on its purpose, has the following methods:

`$do_word_intrusion_test()` (tm) launch the shiny-based word intrusion test. The coder should find out the intruder word that is not related to other words.

`$do_topic_intrusion_test()` (tm) launch the shiny-based topic intrusion test. The coder should find out the intruder topic that is least likely to be the topic of the document.

`$do_word_set_intrusion_test()` (tm) launch the shiny-based word set intrusion test. The coder should find out the intruder word set that is not related to other word sets.

`$do_gold_standard_test()` (gs) launch the shiny-based test for generating gold standard. The coder should determine the level of the predetermined constructs with a 5-point Likert scale.

`$lock(force = FALSE)` (gs/tm) lock the object so that it cannot be changed anymore. It enables `summarize_oolong` and the following method.

`$turn_gold()` (gs) convert the oolong object into a quanteda compatible corpus.

For more details, please see the overview vignette: `vignette("overview", package = "oolong")`

Author(s)

Chung-hong Chan, Marius Sältzer

References

Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J. L., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. In Advances in neural information processing systems (pp. 288-296).

Song et al. (2020) In validations we trust? The impact of imperfect human annotations as a gold standard on the quality of validation of automated content analysis. Political Communication.

Ying, L., Montgomery, J. M., & Stewart, B. M. (2021). Topics, Concepts, and Measurement: A Crowdsourced Procedure for Validating Topics as Measures. Political Analysis

Examples

```
## Creation of oolong test with only word intrusion test
data(abstracts_seededlda)
data(abstracts)
oolong_test <- wi(input_model = abstracts_seededlda, userid = "Hadley")
## Creation of oolong test with both word intrusion test and topic intrusion test
oolong_test <- witi(input_model = abstracts_seededlda,
input_corpus = abstracts$text, userid = "Julia")
## Creation of oolong test with topic intrusion test
oolong_test <- ti(input_model = abstracts_seededlda,
input_corpus = abstracts$text, userid = "Jenny")
## Creation of oolong test with word set intrusion test
oolong_test <- wsi(input_model = abstracts_seededlda, userid = "Garrett")
## Creation of gold standard
oolong_test <- gs(input_corpus = trump2k, userid = "Yihui")
## Using create_oolong(); not recommended
oolong_test <- create_oolong(input_model = abstracts_seededlda,
input_corpus = abstracts$text, userid = "JJ")
oolong_test <- create_oolong(input_model = abstracts_seededlda,
input_corpus = abstracts$text, userid = "Mara", type = "ti")
oolong_test <- create_oolong(input_corpus = abstracts$text, userid = "Winston", type = "gs")
```

---

deploy_oolong *Deploy an oolong test*

Description

In most of the time, you should not use this function. You should write the deployable version of your app into a directory using `export_oolong` instead. Please refer to `vignette("deploy", package = "oolong")` for more details.

Usage

```
deploy_oolong(oolong)
```

Arguments

oolong	an oolong object to be deployed. Please note that the "witi" type, i.e. oolong object with both word and topic intrusion tests, cannot be deployed. Also the object must not be locked and ever coded.

Value

Nothing, it launches a deployable version of the coding interface

Author(s)

Chung-hong Chan

Examples

```
# Please try this example in interactive R sessions only.
if (interactive()) {
   data(abstracts_stm)
   x <- wi(abstracts_stm)
   deploy_oolong(x)
}
```

---

export_oolong *Export a deployable Shiny app from an oolong object into a directory*

Description

This function exports your oolong test into a launched Shiny app that is ideal for online deployment. Deploying the Shiny app online allows coders to conduct the test online with their browser, rather than having to install R on their own computer. In contrast to the testing interfaces launched with methods such as `$do_word_intrusion_test()`, the deployable version provides data download after the coder finished coding. Downloaded data can then revert back to a locked oolong object using `revert_oolong`. Further version might provide solutions to permanent storage. The deployable Shiny app will be in a directory. The Shiny app is both launchable with shiny::runApp() and deployable with rsconnect::deployApp(). Please refer to `vignette("deploy", package = "oolong")` for more details.

Usage

```
export_oolong(
  oolong,
  dir = base::tempdir(),
  verbose = TRUE,
  use_full_path = TRUE
)
```

Arguments

oolong	an oolong object to be exported. Please note that the "witi" type, i.e. oolong object with both word and topic intrusion tests, cannot be exported. Also the object must not be locked and ever coded.
dir	character string, the directory to be exported. Default to a temporary directory
verbose	logical, whether to display information after exporting
use_full_path	logical, whether to expand dir into full path

Value

directory exported, invisible

Author(s)

Chung-hong Chan

Examples

```
# Please try this example in interactive R sessions only.
if (interactive()) {
   data(abstracts_stm)
   x <- wi(abstracts_stm)
   export_oolong(x)
}
```

---

newsgroup_nb *Naive Bayes model trained on 20 newsgroups data*

Description

This is a Naive Bayes model (of the class 'textmodel_nb') trained on 20 newsgroups data.

Usage

```
newsgroup_nb
```

Format

An object of class `textmodel_nb` (inherits from `textmodel`, `list`) of length 7.

References

Lang, K. (1995). Newsweeder: Learning to filter netnews. In Machine Learning Proceedings 1995 (pp. 331-339). Morgan Kaufmann.

print.oolong_gold_standard *Print oolong gold standard object*

Description

This function prints a summary of the oolong gold standard object. An oolong gold standard object is a result of $turn_gold() method. It is a quanteda::corpus compatible object.

Usage

```
## S3 method for class 'oolong_gold_standard'
print(x, ...)
```

Arguments

x	an oolong gold standard object
...	other parameters

Value

None, a summary of the quanteda::corpus and what you should do are displayed

Author(s)

Chung-hong Chan

print.oolong_summary *Print and plot oolong summary*

Description

These functions print or plot a useful summary of the results from summarize_oolong. For details, please see the overview vignette: vignette("overview", package = "oolong")

Usage

```
## S3 method for class 'oolong_summary'
print(x, ...)

## S3 method for class 'oolong_summary'
plot(x, ...)
```

Arguments

x	an oolong_summary
...	other parameters

Value

None

Summary

Print function displays the following information:

Mean model precision (wi, wsi) Higher value indicates better topic interpretability

Quantiles of model precision (wi) Higher value indicates better topic interpretability

P-value of the model precision (wi) Model precision's p-value calculated by one-sample binomial test and Fisher's Omnibus method.

Krippendorff's alpha (wi, wsi, gs) Krippendorff's Alpha, if more than one oolong object is analyzed.

K Precision (wi, wsi) Model precision for each topic.

Mean TLO (ti) Mean topic log odds, higher value indicates better interpretability

Median TLO (ti) Median topic log odds, higher value indicates better interpretability

Quantiles of TLO (ti) Quantiles of topic log odds

P-Value of the median TLO (ti) Median topic log odds's p-value calculated by permutation test.

Correlation (average answer) (gs) Pearson's correlation between average answer and target value

Corrlation (content length) (gs) Pearson's correlation between content length and target value

Diagnostic plot

Plot function displays a diagnostic plot with the following subplots (gs only).

Top left Correlation between answer from coders and target value to check for correlation between two values. Both axes are minmax transformed.

Top right Bland-altman plot of answer from coders and target value to check for agreement between two values.

Bottom left Correlation between target value and content length to check for the influence of content length.

Bottom right Cook's distance to check for influential observations.

Author(s)

Chung-hong Chan

revert_oolong *Obtain a locked oolong from a downloaded data file*

Description

To generate a locked oolong object with the original oolong object and the RDS file. The RDS file should have been downloaded from a deployed Shiny app.

Usage

```
revert_oolong(oolong, rds_file)
```

Arguments

oolong	an oolong object used for deployment
rds_file	path to the downloaded RDS file

Value

a locked oolong object based on the data in the downloaded RDS file

Author(s)

Chung-hong Chan

summarize_oolong *Summarize oolong objects*

Description

This function summarizes one or more oolong objects. All oolong objects must be locked.

Usage

```
summarize_oolong(..., target_value = NULL, n_iter = 1500)

summarise_oolong(..., target_value = NULL, n_iter = 1500)
```

Arguments

...	(tm/gs) one or more oolong objects to be summarized
target_value	(gs) a vector of numeric values, the value you want to validate against the human-coded gold standard. One example of this target value is sentiment score extracted automatically from text
n_iter	(ti) number of iterations to calculate the median test

Value

An oolong summary. Depends on purpose, an oolong summary object has the following values:

`$type` (gs/tm) type of analysis, either 'gs' or 'tm'

`$kripp_alpha`; `$kripp_alpha_wsi` (wi, wsi) Krippendorff's Alpha, if more than one oolong object is analyzed.

`$rater_precision`; `$rater_precision_wsi` (wi, wsi) Model precision

`$res$rater_precision_p_value` (wi) Model precision's p-value calculated by one-sample binomial test and Fisher's Omnibus method.

`$k_precision`; `$k_precision_wsi` (wi, wsi) precision for each topic

`$tlo` (ti) vector of topic log odds

`$tlo_pvalue` (ti) Median topic log odds's p-value calculated by permutation test.

`$cor` (gs) Pearson's correlation between average answer and target value

`$cor_length` (gs) Pearson's correlation between content length and target value

`$diag_plot` (gs) diagnostic plot.

A useful summary of an object can be obtained either by `print.oolong_summary` or `plot.oolong_summary`. For details, please see the overview vignette: `vignette("overview", package = "oolong")`

Author(s)

Chung-hong Chan

References

Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J. L., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. In Advances in neural information processing systems (pp. 288-296).

Song et al. (2020) In validations we trust? The impact of imperfect human annotations as a gold standard on the quality of validation of automated content analysis. Political Communication.

Ying, L., Montgomery, J. M., & Stewart, B. M. (2021). Topics, Concepts, and Measurement: A Crowdsourced Procedure for Validating Topics as Measures. Political Analysis.

Examples

```
# Please try this example in interactive R sessions only.
if (interactive()) {
   data(abstracts_stm)
   oolong_test1 <- create_oolong(abstracts_stm)
   oolong_test2 <- clone_oolong(oolong_test1)
   oolong_test1$do_word_intrusion_test()
   oolong_test2$do_word_intrusion_test()
   oolong_test1$lock()
   oolong_test2$lock()
   summarize_oolong(oolong_test1, oolong_test2)
}
```

trump2k *Trump's tweets dataset*

Description

This is a random sample of 2000 tweets from @realdonaldtrump account before his assumption of duty as the president of the United States.

Usage

```
trump2k
```

Format

An object of class `character` of length 2000.

update_oolong *Update an oolong object to the latest version*

Description

This function update an old oolong object to the latest version.

Usage

```
update_oolong(oolong, verbose = TRUE)
```

Arguments

`oolong`	an oolong object to be updated
`verbose,`	logical, display messages

Value

an updated oolong object

Author(s)

Chung-hong Chan

Index