Package ‘grafzahl’

March 26, 2024

Title Supervised Machine Learning for Textual Data Using Transformers and 'Quanteda'

Version 0.0.11

Description Duct tape the 'quanteda' ecosystem (Benoit et al., 2018) <doi:10.21105/joss.00774> to modern Transformer-based text classification models (Wolf et al., 2020) <doi:10.18653/v1/2020.emnlp-demos.6>, in order to facilitate supervised machine learning for textual data. This package mimics the behaviors of 'quanteda.textmodels' and provides a function to setup the 'Python' environment to use the pretrained models from 'Hugging Face' <`https://huggingface.co/`>. More information: <doi:10.5117/CCR2023.1.003.CHAN>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.1

URL `https://gesistsa.github.io/grafzahl/`,
`https://github.com/gesistsa/grafzahl`

BugReports `https://github.com/gesistsa/grafzahl/issues`

Suggests knitr, quanteda.textmodels, rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Imports jsonlite, lime, quanteda, reticulate, utils, stats

LazyData true

Depends R (>= 3.5)

VignetteBuilder knitr

Config/Needs/website gesistsa/tsatemplate

NeedsCompilation no

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

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

Repository CRAN

Date/Publication 2024-03-26 10:20:02 UTC

R topics documented:

`detect_conda` *Detecting Miniconda And Cuda*

Description

These functions detects miniconda and cuda.

Usage

```
detect_conda()

detect_cuda()
```

Details

`detect_conda` conducts a test to check whether 1) a miniconda installation and 2) the grafzahl miniconda environment exist.

`detect_cuda` checks whether cuda is available. If `setup_grafzahl` was executed with cuda being `FALSE`, this function will return `FALSE`. Even if `setup_grafzahl` was executed with cuda being `TRUE` but with any factor that can't enable cuda (e.g. no Nvidia GPU, the environment was incorrectly created), this function will also return `FALSE`.

Value

boolean, whether the system is available.

`ecosent` *A Corpus Of Dutch News Headlines*

Description

This is a dataset from the paper "The Validity of Sentiment Analysis: Comparing Manual Annotation, Crowd-Coding, Dictionary Approaches, and Machine Learning Algorithms." The data frame contains four columns: id (identifier), headline (the actual text data), value (sentiment: 0 Neutral, +1 Positive, -1 Negative), gold (whether or not this row is "gold standard", i.e. test set). The data is available from Wouter van Atteveldt's Github. `https://github.com/vanatteveldt/ecosent`

Usage

```
ecosent
```

Format

An object of class `data.frame` with 6322 rows and 4 columns.

References

Van Atteveldt, W., Van der Velden, M. A., & Boukes, M. (2021). The validity of sentiment analysis: Comparing manual annotation, crowd-coding, dictionary approaches, and machine learning algorithms. Communication Methods and Measures, 15(2), 121-140.

`get_amharic_data` *Download The Amharic News Text Classification Dataset*

Description

This function downloads the training and test sets of the Amharic News Text Classification Dataset from Hugging Face.

Usage

```
get_amharic_data()
```

Value

A named list of two corpora: training and test

References

Azime, Israel Abebe, and Nebil Mohammed (2021). "An Amharic News Text classification Dataset." arXiv preprint arXiv:2103.05639 grafzahl *Fine tune a pretrained Transformer model for texts*

Description

Fine tune (or train) a pretrained Transformer model for your given training labelled data `x` and `y`. The prediction task can be classification (if `regression` is `FALSE`, default) or regression (if `regression` is `TRUE`).

Usage

```
grafzahl(
  x,
  y = NULL,
  model_name = "xlm-roberta-base",
  regression = FALSE,
  output_dir,
  cuda = detect_cuda(),
  num_train_epochs = 4,
  train_size = 0.8,
  args = NULL,
  cleanup = TRUE,
  model_type = NULL,
  manual_seed = floor(runif(1, min = 1, max = 721831)),
  verbose = TRUE
)

## Default S3 method:
grafzahl(
  x,
  y = NULL,
  model_name = "xlm-roberta-base",
  regression = FALSE,
  output_dir,
  cuda = detect_cuda(),
  num_train_epochs = 4,
  train_size = 0.8,
  args = NULL,
  cleanup = TRUE,
  model_type = NULL,
  manual_seed = floor(runif(1, min = 1, max = 721831)),
  verbose = TRUE
)

## S3 method for class 'corpus'
grafzahl(
  x,
  y = NULL,
  model_name = "xlm-roberta-base",
  regression = FALSE,
  output_dir,
  cuda = detect_cuda(),
  num_train_epochs = 4,
  train_size = 0.8,
  args = NULL,
  cleanup = TRUE,
  model_type = NULL,
  manual_seed = floor(runif(1, min = 1, max = 721831)),
  verbose = TRUE
)

textmodel_transformer(...)

## S3 method for class 'character'
grafzahl(
  x,
  y = NULL,
  model_name = "xlmroberta",
  regression = FALSE,
  output_dir,
  cuda = detect_cuda(),
  num_train_epochs = 4,
  train_size = 0.8,
  args = NULL,
  cleanup = TRUE,
  model_type = NULL,
  manual_seed = floor(runif(1, min = 1, max = 721831)),
  verbose = TRUE
)
```

Arguments

`x`	the corpus or character vector of texts on which the model will be trained. Depending on `train_size`, some texts will be used for cross-validation.
`y`	training labels. It can either be a single string indicating which docvars of the corpus is the training labels; a vector of training labels in either character or factor; or `NULL` if the corpus contains exactly one column in docvars and that column is the training labels. If `x` is a character vector, `y` must be a vector of the same length.
`model_name`	string indicates either 1) the model name on Hugging Face website; 2) the local path of the model
`regression`	logical, if `TRUE`, the task is regression, classification otherwise.
`output_dir`	string, location of the output model. If missing, the model will be stored in a temporary directory. Important: Please note that if this directory exists, it will be overwritten.

cuda	logical, whether to use CUDA, default to detect_cuda().
num_train_epochs	numeric, if train_size is not exactly 1.0, the maximum number of epochs to try in the "early stop" regime will be this number times 5 (i.e. 4 * 5 = 20 by default). If train_size is exactly 1.0, the number of epochs is exactly that.
train_size	numeric, proportion of data in x and y to be used actually for training. The rest will be used for cross validation.
args	list, additionally parameters to be used in the underlying simple transformers
cleanup	logical, if TRUE, the runs directory generated will be removed when the training is done
model_type	a string indicating model_type of the input model. If NULL, it will be inferred from model_name. Supported model types are available in supported_model_types.
manual_seed	numeric, random seed
verbose	logical, if TRUE, debug messages will be displayed
...	paramters pass to grafzahl()

Value

a grafzahl S3 object with the following items

call	original function call
input_data	input_data for the underlying python function
output_dir	location of the output model
model_type	model type
model_name	model name
regression	whether or not it is a regression model
levels	factor levels of y
manual_seed	random seed
meta	metadata about the current session

See Also

predict.grafzahl()

Examples

```
if (detect_conda() && interactive()) {
library(quanteda)
set.seed(20190721)
## Using the default cross validation method
model1 <- grafzahl(unciviltweets, model_type = "bertweet", model_name = "vinai/bertweet-base")
predict(model1)

## Using LIME
input <- corpus(ecosent, text_field = "headline")
training_corpus <- corpus_subset(input, !gold)
model2 <- grafzahl(x = training_corpus,
                 y = "value",
                 model_name = "GroNLP/bert-base-dutch-cased")
test_corpus <- corpus_subset(input, gold)
predicted_sentiment <- predict(model2, test_corpus)
require(lime)
sentences <- c("Dijsselbloem pessimistisch over snelle stappen Grieken",
               "Aandelenbeurzen zetten koersopmars voort")
explainer <- lime(training_corpus, model2)
explanations <- explain(sentences, explainer, n_labels = 1,
                        n_features = 2)
plot_text_explanations(explanations)
}
```

`hydrate` *Create a grafzahl S3 object from the output_dir*

Description

Create a grafzahl S3 object from the output_dir

Usage

```
hydrate(output_dir, model_type = NULL, regression = FALSE)
```

Arguments

`output_dir`	string, location of the output model. If missing, the model will be stored in a temporary directory. Important: Please note that if this directory exists, it will be overwritten.
`model_type`	a string indicating model_type of the input model. If `NULL`, it will be inferred from `model_name`. Supported model types are available in supported_model_types.
`regression`	logical, if `TRUE`, the task is regression, classification otherwise.

Value

a `grafzahl` S3 object with the following items

`call`	original function call
`input_data`	input_data for the underlying python function
`output_dir`	location of the output model
`model_type`	model type
`model_name`	model name
`regression`	whether or not it is a regression model
`levels`	factor levels of y
`manual_seed`	random seed
`meta`	metadata about the current session

predict.grafzahl *Prediction from a fine-tuned grafzahl object*

Description

Make prediction from a fine-tuned grafzahl object.

Usage

```
## S3 method for class 'grafzahl'
predict(object, newdata, cuda = detect_cuda(), return_raw = FALSE, ...)
```

Arguments

object	an S3 object trained with `grafzahl()`
newdata	a corpus or a character vector of texts on which prediction should be made.
cuda	logical, whether to use CUDA, default to `detect_cuda()`.
return_raw	logical, if `TRUE`, return a matrix of logits; a vector of class prediction otherwise
...	not used

Value

a vector of class prediction or a matrix of logits

setup_grafzahl *Setup grafzahl*

Description

Install a self-contained miniconda environment with all Python components (PyTorch, Transformers, Simpletransformers, etc) which grafzahl required. The default location is "~/.local/share/r-miniconda/envs/grafzahl_condaenv" (suffix "_cuda" is added if `cuda` is `TRUE`). On Linux or Mac and if miniconda is not found, this function will also install miniconda. The path can be changed by the environment variable `GRAFZAHL_MINICONDA_PATH`

Usage

```
setup_grafzahl(cuda = FALSE, force = FALSE, cuda_version = "11.3")
```

Arguments

cuda	logical, if `TRUE`, indicate whether a CUDA-enabled environment is wanted.
force	logical, if `TRUE`, delete previous environment (if exists) and create a new environment
cuda_version	character, indicate CUDA version, ignore if `cuda` is `FALSE`

Value

TRUE (invisibly) if installation is successful.

Examples

```
# setup an environment with cuda enabled.
if (detect_conda() && interactive()) {
    setup_grafzahl(cuda = TRUE)
}
```

`supported_model_types` *Supported model types*

Description

A vector of all supported model types.

Usage

```
supported_model_types
```

Format

An object of class `character` of length 23.

`unciviltweets` *A Corpus Of Tweets With Incivility Labels*

Description

This is a dataset from the paper "The Dynamics of Political Incivility on Twitter". The tweets were by Members of Congress elected to the 115th Congress (2017–2018). It is important to note that not all the incivility labels were coded by human. Majority of the labels were coded by the Google Perspective API. All mentions were removed. The dataset is available from Pablo Barbera's Github. `https://github.com/pablobarbera/incivility-sage-open`

Usage

```
unciviltweets
```

Format

An object of class `corpus` (inherits from `character`) of length 19982.

References

Theocharis, Y., Barberá, P., Fazekas, Z., & Popa, S. A. (2020). The dynamics of political incivility on Twitter. Sage Open, 10(2), 2158244020919447.

use_nonconda *Set up grafzahl to be used on Google Colab or similar environments*

Description

Set up grafzahl to be used on Google Colab or similar environments. This function is also useful if you do not want to use conda on a local machine, e.g. you have configurateed the required Python package.

Usage

```
use_nonconda(install = TRUE, check = TRUE, verbose = TRUE)
```

Arguments

`install`	logical, whether to install the required Python packages
`check`	logical, whether to perform a check after the setup. The check displays 1) whether CUDA can be detected, 2) whether the non-conda mode has been activated, i.e. whether the option 'grafzahl.nonconda' is `TRUE`.
`verbose,`	logical, whether to display messages

Value

TRUE (invisibly) if installation is successful.

Examples

```
# A typical use case for Google Colab
if (interactive()) {
    use_nonconda()
}
```

Index