

Package: LLMing (via r-universe)

May 25, 2026

Title Large Language Model (LLM) Tools for Psychological Text Analysis

Version 1.2.1

Maintainer Lindley Slipetz <ddj6tu@virginia.edu>

Description A collection of large language model (LLM) text analysis methods designed with psychological data in mind. Currently, LLMing (aka ``lemming") includes a text anomaly detection method based on the angle-based subspace approach described by Zhang, Lin, and Karim (2015) and a text generation method.
<doi:10.1016/j.res.2015.05.025>.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports Rdpack, quanteda, stopwords, stringi, reticulate, text, dbscan, pracma, stats, jsonlite, quanteda, stopwords, stringi, utils, Matrix, text2vec

SystemRequirements Python (>= 3.10) with packages: torch, transformers, pandas, numpy

RdMacros Rdpack

URL <https://github.com/sliplr19/LLMing>

BugReports <https://github.com/sliplr19/LLMing/issues>

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/pak/sysreqs cmake libfontconfig1-dev libfreetype6-dev make libicu-dev libjpeg-dev libpng-dev libuv1-dev libxml2-dev libssl-dev python3 libx11-dev

Repository <https://sliplr19.r-universe.dev>

Date/Publication 2026-03-26 17:32:48 UTC

RemoteUrl <https://github.com/sliplr19/llming>

RemoteRef HEAD

RemoteSha f6c30365a1bd6fcad6e102c92448b9f07e1e3dad

Contents

embed	2
G_thres	4
normahalo	4
pCOS	5
pCOS_row	5
rep_set	6
sim_SNN	6
text_datagen	7
textanomaly	9
vector_SNN	9
z_score	10

Index	11
--------------	-----------

embed	<i>Embed texts with a selected embedding model</i>
-------	--

Description

Cleans a text column and converts it to a numeric embedding matrix or dataframe, with one row per input text and one column per embedding dimension. Supports pure R and Python-backed embedding methods.

Usage

```
embed(
  dat,
  method = c("E5", "Qwen3", "NV-Embed", "BERT", "GloVe"),
  text_col = "text",
  py_e5_qwen = NULL,
  py_nv = NULL,
  hf_cache_root = NULL,
  max_length = 256,
  prefer_gpu = FALSE,
  auto_install_sentence_transformers = FALSE
)
```

Arguments

dat	A dataframe containing one text per row.
method	Character scalar specifying the embedding method. One of "E5", "Qwen3", "NV-Embed", "BERT", or "GloVe".
text_col	Character scalar giving the name of the text column in dat. Defaults to "text".
py_e5_qwen	Optional path to the Python executable to use for the "E5" and "Qwen3" methods.

<code>py_nv</code>	Optional path to the Python executable to use for the "NV-Embed" method.
<code>hf_cache_root</code>	Optional path to a cache directory for Hugging Face models and related files. Required for Python-backed methods and used as a temporary cache for BERT when not otherwise supplied.
<code>max_length</code>	Integer maximum sequence length passed to the tokenizer for the "NV-Embed" method. Defaults to 256.
<code>prefer_gpu</code>	Logical; if TRUE, Python-backed methods will try to use a CUDA-enabled GPU when available. Defaults to FALSE.
<code>auto_install_sentence_transformers</code>	Logical; if TRUE, attempts to install or update the sentence-transformers Python package for the "E5" and "Qwen3" methods. Defaults to FALSE.

Details

The input text is lightly preprocessed before embedding. This preprocessing:

- removes punctuation, symbols, numbers, and URLs,
- converts text to lowercase,
- removes English stopwords,
- transliterates text to ASCII.

Method-specific behavior:

- "GloVe" uses a pure R workflow via `text2vec` and returns document-level mean embeddings.
- "BERT" uses `text::textEmbed()` with the "bert-base-uncased" model on CPU.
- "E5" and "Qwen3" use Python and `sentence-transformers`.
- "NV-Embed" uses Python and `transformers`.

Value

A numeric matrix or dataframe with one row per input text and one column per embedding dimension.

Examples

```
df <- data.frame(  
  text = c(  
    "I slept well and feel great today!",  
    "I saw friends and it went well.",  
    "I think I failed that exam. I'm such a disappointment."  
  )  
)  
  
emb_dat <- embed(  
  dat = df,  
  method = "GloVe",  
  text_col = "text"  
)
```

G_thres *Thresholding of pCOS dataframe*

Description

Converts each column of a pCOS score matrix into binary indicators

Usage

```
G_thres(pCOS_mat, theta)
```

Arguments

pCOS_mat	Dataframe of pCOS values
theta	Numeric threshold

Value

A matrix of 0s and 1s of which cells meet the threshold

Examples

```
z_dat <- data.frame("A" = rnorm(500,0,1), "B" = rnorm(500,0,1), "C" = rnorm(500,0,1))
snn <- sim_SNN(z_dat, 10, 5)
vec_snn <- vector_SNN(z_dat, snn)
pCOSdat <- pCOS(z_dat, vec_snn)
G <- G_thres(pCOSdat, theta = 0.1)
```

normahalo *Local outlier score*

Description

Computes a normalized Mahalanobis distance score. Only features with nonzero scores in S receive nonzero Mahalanobis scores.

Usage

```
normahalo(z, rs, S)
```

Arguments

z	Dataframe of z scores
rs	List of reference sets
S	Dataframe of numeric values

Value

A dataframe of local outlier scores

pCOS	<i>pCOS scores for every row of dataframe</i>
------	---

Description

Applies pCOS_row() to corresponding rows of two data frames, returning one pCOS value per row.

Usage

```
pCOS(z_dat, vec_SNN)
```

Arguments

z_dat	Numeric dataframe, typically z-scores
vec_SNN	Numeric dataframe, typically the output of vector_SNN

Value

A dataframe with same dimensions as z_dat

pCOS_row	<i>Pairwise cosine-style row score</i>
----------	--

Description

Given two numeric vectors, computes an average cosine-based similarity.

Usage

```
pCOS_row(z, v_SNN)
```

Arguments

z	Numeric vector
v_SNN	Numeric vector, same size as z

Value

A numeric vector

rep_set	<i>The vectors of the shared nearest neighbors</i>
---------	--

Description

Creates a list of the vectors of the top shared nearest neighbors for each row of the z dataframe

Usage

```
rep_set(z, snn)
```

Arguments

z	Dataframe of values of reference set
snn	Dataframe of shared nearest neighbors indices

Value

A list of dataframes where each row of the dataframe is the vector representation of a given shared nearest neighbor

sim_SNN	<i>Compute shared nearest neighbors</i>
---------	---

Description

Builds a shared nearest neighbors matrix and, for each row (observation), returns the indices of the top neighbors with the largest SNN overlap counts

Usage

```
sim_SNN(z_dat, k, tops)
```

Arguments

z_dat	A dataframe with numeric columns
k	An integer representing number of nearest neighbors
tops	An integer representing how many of shared nearest neighbors to return

Value

A dataframe of top rows with shared nearest neighbors

text_datagen	<i>Generate text data via Python LLM</i>
--------------	--

Description

All prompt components and example texts are provided by the user as function arguments. This function generates text data based on severity score from a given questionnaire.

Usage

```
text_datagen(
    prompts,
    examples,
    scenario = NULL,
    overall_rules = NULL,
    percentile_scaffold = NULL,
    item_rules = NULL,
    items = NULL,
    structure_rules = NULL,
    percentile_specification = NULL,
    band_specification = NULL,
    example_instruction = NULL,
    what_to_write = NULL,
    task_desc = NULL,
    target_min = 90L,
    target_max = 100L,
    temperature = 0.4,
    top_p = 0.9,
    repetition_penalty = 1.1,
    model_name = "meta-llama/Meta-Llama-3-8B-Instruct",
    batch_size = 2L,
    python = Sys.getenv("RETICULATE_PYTHON", "python"),
    env = NULL,
    output_file = NULL
)
```

Arguments

prompts	A data.frame with one row per diary to generate. Must contain at least a column indicating severity level.
examples	A data.frame of example diary texts with columns text and label.
scenario	Character string used in the SCENARIO section.
overall_rules	Character string describing global writing rules.
percentile_scaffold	Character string describing how percentiles map onto severity.
item_rules	Character string describing how to internally choose symptom patterns.

items	Character string of the battery under study.
structure_rules	Character string describing structural rules.
percentile_specification	Character string describing what the severity percentile means.
band_specification	Character string describing severity bands.
example_instruction	Character string introducing the example texts.
what_to_write	Character string describing what the model should write about.
task_desc	Character string for the system-level role description.
target_min	Integer minimum number of tokens to generate.
target_max	Integer maximum number of tokens to generate.
temperature	Numeric temperature for sampling.
top_p	Numeric top-p nucleus sampling value.
repetition_penalty	Numeric repetition penalty.
model_name	Model identifier string to pass to transformers.
batch_size	Integer passed through to the Python script.
python	Path to the Python executable.
env	Optional named character vector or list of environment variables.
output_file	Optional path to save the output CSV.

Value

A data.frame with columns id, severity, and response.

Examples

```
prompts <- data.frame(
  id = 1:2,
  severity = c(10, 80),
  num_examples = c(1, 1)
)

examples <- data.frame(
  text = c("Example A", "Example B"),
  label = c("group1", "group2"),
  stringsAsFactors = FALSE
)
```

textanomaly	<i>Text anomaly score</i>
-------------	---------------------------

Description

Text anomaly detection method adapted from (Zhang et al. 2015).

Usage

```
textanomaly(dat, k, tops, theta, text_method)
```

Arguments

dat	A dataframe with text data, one text per row
k	An integer representing number of nearest neighbors
tops	An integer representing how many of shared nearest neighbors to return
theta	Numeric threshold
text_method	Character scalar specifying the embedding method. One of "E5", "Qwen3", "NV-Embed", "BERT", or "GloVe"

Value

Dataframe of local outlier score

References

Zhang L, Lin J, Karim R (2015). "An angle-based subspace anomaly detection approach to high-dimensional data: With an application to industrial fault detection." *Reliability Engineering & System Safety*, **142**, 482–497. ISSN 0951-8320. doi:[10.1016/j.res.2015.05.025](https://doi.org/10.1016/j.res.2015.05.025).

vector_SNN	<i>Aggregate dataframe into mean feature vectors Aggregate dataframe into mean feature vectors</i>
------------	--

Description

For each row of the SNN index matrix, this function takes the rows of reference dataframe, z, and computes their column means, yielding one mean vector per observation.

Usage

```
vector_SNN(z, snn)
```

Arguments

z	Numeric dataframe
snn	Dataframe of shared nearest neighbors indices

Value

Dataframe of same dimensions as z

z_score	<i>Z-score on columns</i>
---------	---------------------------

Description

Z-score on columns

Usage

```
z_score(dat)
```

Arguments

dat	A dataframe with numeric cells
-----	--------------------------------

Value

A dataframe with numeric cells with the same dimensions as dat

Index

embed, [2](#)

G_thres, [4](#)

normahalo, [4](#)

pCOS, [5](#)

pCOS_row, [5](#)

rep_set, [6](#)

sim_SNN, [6](#)

text_datagen, [7](#)

textanomaly, [9](#)

vector_SNN, [9](#)

z_score, [10](#)