Package 'kgraph'

March 10, 2025

Title Knowledge Graphs Constructions and Visualizations

Version 1.2.0

Description Knowledge graphs enable to efficiently visualize and gain insights into large-scale data analysis results, as p-values from multiple studies or embedding data matrices. The usual workflow is a user providing a data frame of association studies results and specifying target nodes, e.g. phenotypes, to visualize. The knowledge graph then shows all the features which are significantly associated with the phenotype, with the edges being proportional to the association scores. As the user adds several target nodes and grouping information about the nodes such as biological pathways, the construction of such graphs soon becomes complex. The 'kgraph' package aims to enable users to easily build such knowledge graphs, and provides two main features: first, to enable building a knowledge graph based on a data frame of concepts relationships, be it p-values or cosine similarities; second, to enable determining an appropriate cut-off on cosine similarities from a complete embedding matrix, to enable the building of a knowledge graph directly from an embedding matrix. The 'kgraph' package provides several display, layout and cut-off options, and has already proven useful to researchers to enable them to visualize large sets of p-value associations with various phenotypes, and to quickly be able to visualize embedding results. Two example datasets are provided to demonstrate these behaviors, and several live 'shiny' applications are hosted by the CELEHS laboratory and Parse Health, as the KESER Mental Health application <`https://keser-mental-health.parse-health.org/`> based on Hong C. (2021) <doi:10.1038/s41746-021-00519-z>.

Imports dplyr, htmltools, igraph, magrittr, opticskxi, plyr, pROC, RColorBrewer, reshape2, sgraph, shiny

Depends R (>= 3.5.0)

Suggests bslib, data.table, DT, grid, knitr, nlpembeds, rmarkdown, testthat

VignetteBuilder knitr

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

URL `https://gitlab.com/thomaschln/kgraph`

BugReports `https://gitlab.com/thomaschln/kgraph/-/issues`

NeedsCompilation no

Author Thomas Charlon [aut, cre] (<`https://orcid.org/0000-0001-7497-0470`>),
Hongyi Yuan [ctb] (<`https://orcid.org/0000-0003-2597-1973`>),
CELEHS [aut] (<`https://celehs.hms.harvard.edu`>),
PARSE Health [aut] (<`https://parse-health.org`>)

Maintainer Thomas Charlon <`charlon@protonmail.com`>

Repository CRAN

Date/Publication 2025-03-10 09:20:02 UTC

Contents

`build_kgraph` *Build a knowledge graph*

Description

Build a knowledge graph

Usage

```
build_kgraph(
  selected_concepts,
  df_weights,
  df_dict = NULL,
  rm_single_groups = TRUE,
  df_sup_nodes = NULL,
  display_val_digits = 3,
  display_val_str = "\nCosine similarity: ",
  str_other = "Other",
  highlight_mult = TRUE,
  multiline_labs = TRUE,
  autoscale = TRUE,
  spring_weights = TRUE,
  n_max_edges = 1000,
  ...
)
```

Arguments

`selected_concepts`	Concepts of interest
`df_weights`	Data frame with columns concept1, concept2, and weight; typically the df_projs slot of the object returned by function fit_embeds_to_pairs
`df_dict`	Dictionary data frame, with columns id (matched to concepts in df_weights), desc (for labels), color, and optionally group.
`rm_single_groups`	Should groups with only one element be removed
`df_sup_nodes`	Data frame of supplementary nodes (work in progress)
`display_val_digits`	Number of weight digits to be displayed in labels
`display_val_str`	String to prefix to the displayed value
`str_other`	String to use for missing groups
`highlight_mult`	Highlight nodes connected to multiple nodes of interest.
`multiline_labs`	Use multiline labels (shown when hovered on)
`autoscale`	Perform scaling with sgraph::scale_graph
`spring_weights`	Use spring weights (reverts edges weights)
`n_max_edges`	Threshold on number of edges
`...`	Passed to scale_kgraph

Value

Knowledge graph, list of slots df_nodes and df_links

`build_kgraph_from_fit` *Build a knowledge graph from a fit object*

Description

Computes similarities for nodes of interest on the fly to avoid having to deal with very large similarity matrices when number of features is large.

Usage

```
build_kgraph_from_fit(selected_concepts, m_embeds, fit_kg, ...)
```

Arguments

`selected_concepts`	Concepts of interest
`m_embeds`	Embeddings matrix
`fit_kg`	Fit object
`...`	Passed to build_kgraph

Value

Knowledge graph, list of slots df_nodes and df_links

`cov_simi` *Covariance similarity*

Description

Covariance similarity

Usage

```
cov_simi(m_data)
```

Arguments

`m_data`	Data matrix

Value

Similarity matrixd

df_cuis_pairs *A dataset containing CUIs pairs*

Description

The dataframe provides clinician-curated pairs of related of medical concepts, useful to evaluate the performance of a machine learning model. It's an extract of the PrimeKG database (see vignette for URL).

Usage

```
data("df_cuis_pairs")
```

Format

A dataframe with 2358 rows and 4 columns.

Details

Each row defines a relationship between two CUIs, along with their textual descriptions.

Examples

```
data('m_embeds')
data('df_cuis_pairs')

fit_kg = fit_embeds_kg(m_embeds, 'cosine',
                       df_pairs = df_cuis_pairs[c(1, 3)])
pROC::plot.roc(fit_kg$roc, print.auc = TRUE)
```

df_embeds_dict *A dictionary for the m_embeds object*

Description

Dataframe with columns id (for the CUI), desc (textual description), group and color (higher level groups)

Usage

```
data("df_embeds_dict")
```

Format

A dataframe with 1118 rows and 4 columns.

Details

Each row corresponds to one rowname of m_embeds.

Examples

```
data('m_embeds')
data('df_embeds_dict')

fit_kg = fit_embeds_kg(m_embeds, 'cosine')
target_nodes_idxs = grep('suicide', df_embeds_dict$desc) %>% head(2)
target_nodes = df_embeds_dict$id[target_nodes_idxs]

kg_obj = build_kgraph_from_fit(target_nodes, m_embeds, fit_kg,
                               df_dict = df_embeds_dict)
```

df_phecode_pairs   *A dataset containing Phecode pairs*

Description

The dataframe provides clinician-curated pairs of related of medical concepts, useful to evaluate the performance of a machine learning model. It's an extract of the PrimeKG database (see vignette for URL).

Usage

```
data("df_phecode_pairs")
```

Format

A dataframe with 3288 rows and 4 columns.

Details

Each row defines a relationship between two Phecodes, along with their textual descriptions.

df_pval *A dataset containing GWAS p-values*

Description

This dataframe provides association scores between SNPs and mantal health-related phenotypes.

Usage

```
data("df_pval")
```

Format

A dataframe with 364 rows and 3 columns

Details

Each row defines an association between a SNP and a phenotype. Downloaded from GWAS Catalog at https://www.ebi.ac.uk/gwas/efotraits/EFO_0007623.

Examples

```
data('df_pval')

kg_obj = build_kgraph('EFO_0007623', df_pval)
```

df_pval_dict *A dictionary for the df_pval object*

Description

Dataframe with columns id (for the phenotype or SNP identifier), desc (textual description), group, and color

Usage

```
data("df_pval_dict")
```

Format

A dataframe with 333 rows and 4 columns.

Details

Row IDs correspond to the identifiers found in columns concept1 and concept2 of the df_pval object.

Examples

```
data('df_pval')
data('df_pval_dict')

kg_obj = build_kgraph(c('EFO_0007623', 'EFO_0007624'), df_pval, df_pval_dict)
```

---

fit_embeds_kg *Fit embeddings to a kgraph object*

Description

Build a fit_kgraph object to act as an intermediate between the embeddings and the knowledge graph. If possible (i.e. if number of features is not too large) compute all pair-wise similarities, otherwise determine the similarity threshold using a number of random pairs. If a data frame of known pairs is available, call fit_embeds_to_pairs which will produce an AUC and use the threshold_projs parameter as the specificity threshold (e.g. the default specificity of 0.9 corresponds to 10 percent false positives). Otherwise take the quantile of similarity values corresponding to threshold_projs.

Usage

```
fit_embeds_kg(
  m_embeds,
  similarity = c("cosine", "inprod", "cov_simi", "norm_inprod"),
  threshold_projs = 0.9,
  df_pairs = NULL,
  df_pairs_cols = 1:2,
  max_concepts = 1000,
  ...
)
```

Arguments

`m_embeds`	Embedding matrix, rownames must be able to be matched to concepts in df_pairs
`similarity`	Similarity measure to be computed. One of 'inprod' (inner product), 'cosine', 'cov_simi' (covariance similarity), 'norm_inprod' (normalized inner product).
`threshold_projs`	Specificity threshold to use for projections. (default 0.9 is equivalent to 10 percent false positives, and 0.95 to 5 percent false positives)
`df_pairs`	Known relationships data frame
`df_pairs_cols`	Columns of df_pairs for identifiers, that map to m_embeds rownames
`max_concepts`	Maximum number of concepts to compute all pair-wise similarities
`...`	Passed to gen_df_notpairs

Value

Knowledge graph, list of slots df_nodes and df_links

fit_embeds_to_pairs *Fit embeds to pairs*

Description

Fit an embeddings matrix to a dataframe of known pairs of related concepts. Depending on matrix dimension, either compute all pair-wise similarities, or only those existing in the known pairs.

Usage

```
fit_embeds_to_pairs(
  m_embeds,
  df_pairs,
  df_pairs_cols = 1:2,
  similarity = c("inprod", "cosine", "cov_simi", "norm_inprod"),
  threshold_projs = 0.9,
  max_concepts = 1000
)
```

Arguments

`m_embeds`	Embedding matrix, rownames must be able to be matched to concepts in df_pairs
`df_pairs`	Known relationships data frame
`df_pairs_cols`	Columns of df_pairs for identifiers, that map to m_embeds rownames
`similarity`	Similarity measure to be computed. One of 'inprod' (inner product), 'cosine', 'cov_simi' (covariance similarity), 'norm_inprod' (normalized inner product).
`threshold_projs`	Specificity threshold to use for projections. (default 0.9 is equivalent to 10 percent false positives, and 0.95 to 5 percent false positives)
`max_concepts`	Maximum number of concepts to compute all pair-wise similarities

Value

List object with slots roc (pROC::roc return), sims and truth (to recompute partial AUCs using pROC), threshold_5fp (5 percent false positive threshold), n_concepts (length of concepts in embeddings), and df_projs (data frame listing pair-wise concepts similarities above threshold_projs).

`gen_df_notpairs` *Generate null pairs*

Description

Generate null pairs

Usage

```
gen_df_notpairs(
  ids,
  df_pairs = NULL,
  n_notpairs = if (is.null(df_pairs)) 1000 else nrow(df_pairs)
)
```

Arguments

`ids`	Identifiers to sample from
`df_pairs`	Known pairs data frame, to make sure no null pairs are in
`n_notpairs`	Direct parameter to set number of null pairs returned, bypasses parameter type.

Value

Data frame with columns concept1, concept2, weight

`get_cutoff_threshold` *Get cut-off threshold*

Description

Get cut-off threshold

Usage

```
get_cutoff_threshold(roc_obj, specificity_lvl = 0.95)
```

Arguments

`roc_obj`	Object returned by pROC::roc
`specificity_lvl`	Specificity threshold (default 0.95 is equivalent to 5 percent false positives, and 0.9 to 10 percent false positives)

Value

Similarity value threshold

`get_sgraph` *Wrapper to build a sgraph object fromk a kgraph object*

Description

Wrapper to build a sgraph object fromk a kgraph object

Usage

```
get_sgraph(
  l_graph,
  colors_mapping = NULL,
  label_attrs = "label",
  igraph = NULL,
  ...
)
```

Arguments

`l_graph`	List of df_nodes and df_links dataframes
`colors_mapping`	Output of get_colors_map
`label_attrs`	Column name of df_nodes that will be displayed
`igraph`	Intermediary igraph object, if already computed
`...`	Passed to sgraph::sgraph_clusters

Value

Sgraph htmlwidget object

`m_embeds` *A dataset containing medical word embeddings*

Description

The embedding matrix has been fitted using Glove word embeddings on 1,700 open-access publications related to mental health.

Usage

```
data("m_embeds")
```

Format

A matrix with 1122 rows and 100 columns.

Details

Each row is the embedding vector of a CUI in 100 Glove dimensions.

Examples

```
data('m_embeds')

fit_kg = fit_embeds_kg(m_embeds, 'cosine')
```

`project_pairs` *Predict known pairs*

Description

Predict known pairs

Usage

```
project_pairs(m_simi, threshold)
```

Arguments

`m_simi`	Similarity matrix
`threshold`	Similarity value threshold

Value

Data frame with columns concept1, concept2, weight

`reshape_multiple_traits` *Reshape multiple traits in example data*

Description

Reshape multiple traits in example data

Usage

```
reshape_multiple_traits(df_pval)
```

Arguments

`df_pval`	Data frame of p-values

Value

Reshaped data frame

`reshape_multiple_traits_dict` *Reshape multiple traits in example data dictionary*

Description

Reshape multiple traits in example data dictionary

Usage

```
reshape_multiple_traits_dict(df_dict)
```

Arguments

`df_dict`	Data frame of p-values dictionary

Value

Reshaped data frame

`sparse_encode` *sparse_encode*

Description

Sparse encoding method by closest neighbors. Three methods are available: - hard encoding: each patient's closest neighbors are set to 1, others are set to 0 - soft encoding: each patient's closest neighbors distances are transformed by the exponential norm, others are set to 0 - epsilon encoding: each patient's neighbors closer than the mean of the distance matrix are transformed by the exponential norm and others are set to 0.

Usage

```
sparse_encode(
  m_data,
  dist_method = "norm_inprod",
  encoding = c("epsilon", "hard", "soft"),
  sigma,
  n_neighbors = floor(nrow(m_data)/10),
  scale_obs = TRUE
)
```

Arguments

`m_data`	Numeric matrix
`dist_method`	Distance method passed to qb_dist
`encoding`	Encoding method: one of hard, soft, or epsilon
`sigma`	Parameter for the exponential norm transform. Default is mean of std. dev. of distance matrix columns
`n_neighbors`	Number of neighbors (ignored in epsilon encoding)
`scale_obs`	Scale by observations

Value

Projected matrix

`%<>%` *Assignment pipe*

Description

Pipe an object forward into a function or call expression and update the ‘lhs‘ object with the resulting value. Magrittr imported function, see details and examples in the magrittr package.

Arguments

`lhs`	An object which serves both as the initial value and as target.
`rhs`	a function call using the magrittr semantics.

Value

None, used to update the value of lhs.

`%$%` *Exposition pipe*

Description

Expose the names in ‘lhs‘ to the ‘rhs‘ expression. Magrittr imported function, see details and examples in the magrittr package.

Arguments

`lhs`	A list, environment, or a data.frame.
`rhs`	An expression where the names in lhs is available.

Value

Result of rhs applied to one or several names of lhs.

`%>%` *Pipe*

Description

Pipe an object forward into a function or call expression. Magrittr imported function, see details and examples in the magrittr package.

Arguments

`lhs`	A value or the magrittr placeholder.
`rhs`	A function call using the magrittr semantics.

Value

Result of rhs applied to lhs, see details in magrittr package.

Index