Package ‘rsemmed’

May 30, 2024

Version 1.14.0

Title An interface to the Semantic MEDLINE database

Description A programmatic interface to the Semantic MEDLINE database. It provides functions for searching the database for concepts and finding paths between concepts. Path searching can also be tailored to user specifications, such as placing restrictions on concept types and the type of link between concepts. It also provides functions for summarizing and visualizing those paths.

Depends R (>= 4.0), igraph

Suggests testthat, knitr, BiocStyle, rmarkdown

Imports methods, magrittr, stringr, dplyr

VignetteBuilder knitr

License Artistic-2.0

Encoding UTF-8

RoxygenNote 7.1.1

URL https://github.com/lmyint/rsemmed

BugReports https://github.com/lmyint/rsemmed/issues

biocViews Software, Annotation, Pathways, SystemsBiology

git_url https://git.bioconductor.org/packages/rsemmed

git_branch RELEASE_3_19

git_last_commit 56080ed

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-29

Author Leslie Myint [aut, cre] (<https://orcid.org/0000-0003-2478-0331>)

Maintainer Leslie Myint <leslie.myint@gmail.com>
find_nodes

Search for nodes by name or semantic type

Description

Search for nodes by name (exact match or using regular expressions) or which match supplied semantic types. Perform anti-matching by setting match = FALSE. Capitalization is ignored.

Usage

find_nodes(obj, pattern = NULL, names = NULL, semtypes = NULL, match = TRUE)

Arguments

obj Either the SemMed graph or a node set (igraph.vs)

pattern Regular expression used to find matches in node names

names Character vector of exact node names

semtypes Character vector of semantic types

match If TRUE, return nodes that DO match pattern (default). If FALSE, return nodes that DO NOT match.

Value

A vertex sequence of matching nodes
**find_paths**

**Examples**

```r
data(g_mini)
find_nodes(g_mini, pattern = "cortisol")
find_nodes(g_mini, pattern = "cortisol$")
find_nodes(g_mini, pattern = "stress")
find_nodes(g_mini, pattern = "stress") %>%
  find_nodes(pattern = "disorder", match = FALSE)

find_nodes(g_mini, names = "Serum cortisol")
find_nodes(g_mini, names = "Chronic Stress")

find_nodes(g_mini, semtypes = "dsyn")
find_nodes(g_mini, semtypes = c("dsyn", "fndg"))

## pattern and semtypes are combined via OR:
find_nodes(g_mini, pattern = "cortisol", semtypes = "horm")

## To make an AND query, chain find_nodes sequentially:
find_nodes(g_mini, pattern = "cortisol") %>%
  find_nodes(semtypes = "horm")
```

---

**find_paths**  
*Shortest paths between node sets*

**Description**

Find all shortest paths between sets of nodes

**Usage**

```r
find_paths(graph, from, to, weights = NULL)
```

**Arguments**

- `graph`: The SemMed graph
- `from`: A set of source nodes. `from` should be of class `igraph.vs` (a vertex sequence) or an integer vector.
- `to`: A set of destination nodes. `to` should be of class `igraph.vs` (a vertex sequence) or an integer vector.
- `weights`: A numeric vector of edge weights. If `NULL` (the default), all edges have the default weight of 1.
Details

find_paths relies on igraph::all_shortest_paths to find all shortest paths between the nodes in from and to. This function searches for undirected paths.

Because the Semantic MEDLINE graph is a multigraph, there may be multiple paths with the same sequence of nodes. This function collapses these into a single node sequence. The display functions (text_path and plot_path) take care of showing the multiple edges leading to repeated paths.

Value

A list of shortest paths. List items correspond to the node(s) given in from.

See Also

make_edge_weights to tailor the shortest path search

Examples

data(g_mini)

node_cortisol <- find_nodes(g_mini, names = "Serum cortisol")
node_stress <- find_nodes(g_mini, names = "Chronic Stress")
find_paths(g_mini, from = node_cortisol, to = node_stress)

details

get_edge_features

Get information about edges

Description

Search for nodes by name using regular expressions or which match given semantic types. Perform anti-matching by setting match = FALSE.

Usage

get_edge_features(
  graph,
  include_degree = FALSE,
  include_node_ids = FALSE,
  include_num_instances = FALSE
)

Arguments

graph The SemMed graph
include_degree If TRUE, include information on head/tail node degrees.
include_node_ids If TRUE, include the ID numbers of head/tail nodes.
**get_middle_nodes**

include_num_instances

If TRUE, include information on the number of times a predication was observed in the Semantic MEDLINE database.

**Value**

A tbl where each row corresponds to an edge in the Semantic MEDLINE graph. The ordering of the rows corresponds to $E(graph)$. Features (columns) always returned include the name and semantic type of the head (subject) and tail (object) nodes.

**See Also**

`make_edge_weights` for using this data to construct edge weights

**Examples**

data(g_mini)

e_feat <- get_edge_features(g_mini)

---

**Description**

For each pair of source and target nodes in `object`, obtain the names of middle nodes on paths.

**Usage**

get_middle_nodes(graph, object, collapse = TRUE)

**Arguments**

- `graph`: The SemMed graph
- `object`: A vertex sequence (**igraph.vs**), a list of vertex sequences, or a list of vertex sequence lists
- `collapse`: If TRUE, middle node names for different source-target pairs are combined into one character vector.

**Value**

A tbl where each row corresponds to a source-target pair in `object`. The last column is a list-column containing character vectors of names of middle nodes.
Examples

data(g_mini)

node_cortisol <- find_nodes(g_mini, "Serum cortisol")
node_stress <- find_nodes(g_mini, "Chronic Stress")
paths <- find_paths(g_mini, from = node_cortisol, to = node_stress)
middle <- get_middle_nodes(g_mini, paths)

grow_nodes

Obtain immediate neighbors

Description

Grow a set of nodes into its first order neighborhood.

Usage

grow_nodes(graph, nodes)

Arguments

graph The SemMed graph
nodes A vertex sequence (igraph.vs) of nodes to be grown

Details

grow_nodes obtains the set of immediate neighbors of the supplied nodes using igraph::ego. Unlike ego, grow_nodes flattens the result from a list to an ordinary vertex sequence and removes the original search nodes.

Value

A vertex sequence of nodes in the neighborhood (not including the original nodes)

See Also

find_nodes for filtering out irrelevant nodes from this set.

Examples

data(g_mini)

node_cortisol <- find_nodes(g_mini, name = "hypercortisolemia")
nbrs <- grow_nodes(g_mini, node_cortisol)
**Description**

A dataset containing a very small subset of the full Semantic MEDLINE graph.

**Usage**

```r
data(g_mini)
```

**Format**

An igraph with 7 nodes and 15 edges

---

**Description**

A dataset containing a small subset of the full Semantic MEDLINE graph.

**Usage**

```r
data(g_small)
```

**Format**

An igraph with 1038 nodes and 318,105 edges

---

**Description**

Create edge weights to modify the shortest path search (`find_paths`). Discourage and/or encourage certain types of paths by supplying `out` and `in` arguments, respectively. Node semantic types, node names, and edge predicates are the features that can influence the edge weights. Capitalization is ignored.

**Usage**

```r
make_edge_weights
```

**Format**

Create edge weights
make_edge_weights

Usage

make_edge_weights(
  graph,           # The SemMed graph
  e_feat,          # A data.frame of edge features from get_edge_features.
  node_semtypes_out = NULL, # A character vector of semantic types to exclude from shortest paths.
  node_names_out = NULL, # A character vector of exact node names to exclude.
  edge_preds_out = NULL, # A character vector of edge predicates to exclude.
  node_semtypes_in = NULL, # A character vector of semantic types to include/encourage in shortest paths.
  node_names_in = NULL, # A character vector of exact node names to include.
  edge_preds_in = NULL) # A character vector of edge predicates to include.

Arguments

graph | The SemMed graph

(e_feat) | A data.frame of edge features from get_edge_features.

node_semtypes_out | A character vector of semantic types to exclude from shortest paths.

node_names_out | A character vector of exact node names to exclude.

eedge_preds_out | A character vector of edge predicates to exclude.

node_semtypes_in | A character vector of semantic types to include/encourage in shortest paths.

node_names_in | A character vector of exact node names to include.

eedge_preds_in | A character vector of edge predicates to include.

Value

A numeric vector of weights

See Also

find_paths, get_middle_nodes for a way to obtain node names to remove

Examples

data(g_mini)

node_cortisol <- find_nodes(g_mini, names = "Serum cortisol")
node_stress <- find_nodes(g_mini, names = "Chronic Stress")
paths <- find_paths(g_mini, from = node_cortisol, to = node_stress)

e_feat <- get_edge_features(g_mini)

w1 <- make_edge_weights(g_mini, e_feat, edge_preds_in = "COEXISTS_WITH")
paths1 <- find_paths(g_mini, from = node_cortisol, to = node_stress, weights = w1)

w2 <- make_edge_weights(g_mini, e_feat, edge_preds_in = "ISA",
### plot_path

```r
node_names_out = "Stress"
paths2 <- find_paths(g_mini,
  from = node_cortisol, to = node_stress, weights = w2)
```

---

### Description

Plot the graph form of a path

### Usage

```r
plot_path(graph, path)
```

### Arguments

- `graph`: The SemMed graph
- `path`: A vertex sequence (igraph.vs) (the path to display)

### Details

All connections among nodes along the supplied path are plotted with nodes labeled with their name and edges labeled with their predicate.

### Value

A plot is created on the current graphics device

### See Also

- `text_path` for textual display of paths

### Examples

```r
data(g_mini)

node_cortisol <- find_nodes(g_mini, names = "Serum cortisol")
node_stress <- find_nodes(g_mini, names = "Chronic Stress")
paths <- find_paths(g_mini, from = node_cortisol, to = node_stress)
plot_path(g_mini, paths[[1]][[1]])
```
### summarize_predicates

Summarize the predicates present in a collection of paths

#### Usage

```r
code
summarize_predicates(graph, object, print = TRUE)
```

#### Arguments

- `graph`: The SemMed graph
- `object`: A vertex sequence (`igraph.vs`), a list of vertex sequences, or a list of vertex sequence lists
- `print`: If `TRUE`, information on predicates will be printed to the screen.

#### Details

Because predicates are edge features, it is assumed that by using `summarize_predicates` the nodes contained in `object` are ordered (paths). This is why `summarize_semtypes` has the `is_path` argument, but `summarize_predicates` does not. `summarize_predicates` tabulates edge predicates across paths corresponding to each from-to pair in `object`.

#### Value

A `tbl` where each row corresponds to a from-to pair in `object`. The last column is a list-column containing table’s of predicate counts.

#### See Also

- `summarize_semtypes` for tabulating semantic types of nodes in paths or other node collections

#### Examples

```r
data(g_mini)
node_cortisol <- find_nodes(g_mini, "Serum cortisol")
node_stress <- find_nodes(g_mini, "Chronic Stress")
paths <- find_paths(g_mini, from = node_cortisol, to = node_stress)
summarize_predicates(g_mini, paths)
```
**summarize_semtypes**

**Summarize semantic types**

**Description**
Summarize the semantic types present in a collection of nodes

**Usage**

```
summarize_semtypes(graph, object, print = TRUE, is_path = TRUE)
```

**Arguments**

- `graph`: The SemMed graph
- `object`: A vertex sequence (`igraph.vs`), a list of vertex sequences, or a list of vertex sequence lists
- `print`: If `TRUE`, information on semantic types will be printed to the screen.
- `is_path`: If `TRUE`, object contains paths (ordered sequences of nodes).

**Details**

`summarize_semtypes` summarizes the semantic types present in supplied node collections and has different behavior depending on whether the node collection is ordered (paths) or unordered. Using `is_path = TRUE` indicates that the nodes are ordered. Using `is_path = FALSE` indicates that the nodes are an unordered collection, often from `find_nodes` or `grow_nodes`.

Using `is_path = TRUE`: When the node collection is ordered, the object is assumed to be the result of `find_paths` or a subset of such an object. Because `find_paths` returns a list of paths lists, `summarize_semtypes` takes a single path, a list of paths, or a list of path lists as input. In the case of a collection of ordered nodes, `summarize_semtypes` counts the semantic types present in `object`. If a node is associated with multiple semantic types, each type is counted once. The first and last nodes of each path are removed they correspond to the nodes in `from` and to `from find_paths`, and it is assumed that the middle nodes on the paths are more of interest. The tabulations are printed to screen (if `print = TRUE`) and returned as table's. These table's are bundled into a list-column of a tbl in the (invisibly returned) output. Each row of the tbl corresponds to a `from-to` pair present in `object`.

Using `is_path = FALSE`: This option is for summarizing results from `find_nodes` and `grow_nodes`, which return unordered node sets. (Note: paths and unordered node sets are both represented as igraph vertex sequences (class `igraph.vs`).) The printed output shows information for each semantic type present in `object`. It shows all nodes of that semantic type as well as their degree and degree percentile within the entire graph. The (invisibly returned) output combines all of the printed information in a tbl.
text_path

Description

Show a text display of a path and obtain output that can be used to explore predications along the path. (A predication is a SUBJECT–LINKING VERB–OBJECT triple.)

Usage

text_path(graph, path, print = TRUE)

Arguments

graph The SemMed graph
path A vertex sequence (igraph.vs) (the path to display)
print Print the path to screen?
text_path

Details
text_path invisibly returns a list of tbl’s containing information on the predications on the path. Each list element is a tbl that corresponds to a (sequential) pair of nodes along the path. The tbl contains information on the subject and object node’s name and semantic type as well as all predicates linking the subject and object.

Value
Invisibly returns a list of predications for each pair of nodes along the path.

See Also
plot_path for plotting paths

Examples
data(g_mini)

node_cortisol <- find_nodes(g_mini, names = "Serum cortisol")
node_stress <- find_nodes(g_mini, names = "Chronic Stress")
paths <- find_paths(g_mini, from = node_cortisol, to = node_stress)
text_path(g_mini, paths[[1]][[1]])
result <- text_path(g_mini, paths[[1]][[1]], print = FALSE)
Index

* datasets
  g_mini, 7
  g_small, 7

find_nodes, 2, 6, 12
find_paths, 3, 8, 12

g_mini, 7
  g_small, 7
get_edge_features, 4
get_middle_nodes, 5, 8
grow_nodes, 6, 12

make_edge_weights, 4, 5, 7

plot_path, 9, 13

summarize_predicates, 10, 12
summarize_semtypes, 10, 11

text_path, 9, 12