Package 'pathfindR'

November 15, 2021

Type Package

Title Enrichment Analysis Utilizing Active Subnetworks

Version 1.6.3

Maintainer Ege Ulgen <egeulgen@gmail.com>

Description Enrichment analysis enables researchers to uncover mechanisms underlying a phenotype. However, conventional methods for enrichment analysis do not take into account protein-protein interaction information, resulting in incomplete conclusions. pathfindR is a tool for enrichment analysis utilizing active subnetworks. The main function identifies active subnetworks in a protein-protein interaction network using a user-provided list of genes and associated p values. It then performs enrichment analyses on the identified subnetworks, identifying enriched terms (i.e. pathways or, more broadly, gene sets) that possibly underlie the phenotype of interest. pathfindR also offers functionalities to cluster the enriched terms and identify representative terms in each cluster, to score the enriched terms per sample and to visualize analysis results. The enrichment, clustering and other methods implemented in pathfindR are described in detail in Ulgen E, Ozisik O, Sezerman OU. 2019. pathfindR: An R Package for Comprehensive Identification of Enriched Pathways in Omics Data Through Active Subnetworks. Front. Genet. <doi:10.3389/fgene.2019.00858>.

License MIT + file LICENSE

URL https://egeulgen.github.io/pathfindR/, https://github.com/egeulgen/pathfindR

BugReports https://github.com/egeulgen/pathfindR/issues

Encoding UTF-8

SystemRequirements Java (>= 8.0)

biocViews

Imports DBI, AnnotationDbi, doParallel, foreach, rmarkdown, org.Hs.eg.db, ggplot2, ggraph, ggupset, fpc, grDevices, igraph, R.utils, magick, msigdbr, KEGGREST, KEGGgraph, knitr

Depends R (>= 4.0), pathfindR.data

Suggests testthat (>= 2.3.2), covr

RoxygenNote 7.1.2

VignetteBuilder knitr

NeedsCompilation no

Author Ege Ulgen [cre, cph] (<https://orcid.org/0000-0003-2090-3621>),
Ozan Ozisik [aut] (<https://orcid.org/0000-0001-5980-8002>)

Repository CRAN

Date/Publication 2021-11-15 09:20:04 UTC

R topics documented:

`active_snw_search` *Perform Active Subnetwork Search*

Description

Perform Active Subnetwork Search

Usage

```
active_snw_search(
  input_for_search,
  pin_name_path = "Biogrid",
  snws_file = "active_snws",
  dir_for_parallel_run = NULL,
  score_quan_thr = 0.8,
  sig_gene_thr = 0.02,
  search_method = "GR",
  silent_option = TRUE,
  use_all_positives = FALSE,
  geneInitProbs = 0.1,
  saTemp0 = 1,
  saTemp1 = 0.01,
  saIter = 10000,
  gaPop = 400,
  gaIter = 10000,
  gaThread = 5,
  gaCrossover = 1,
  gaMut = 0,
  grMaxDepth = 1,
  grSearchDepth = 1,
  grOverlap = 0.5,
  grSubNum = 1000
)
```

Arguments

`input_for_search`
: input the input data that active subnetwork search uses. The input must be a data frame containing at least these 2 columns:

  GENE Gene Symbol

  P_VALUE p value obtained through a test, e.g. differential expression/methylation

`pin_name_path`
: Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")

`snws_file`
: name for active subnetwork search output data **without file extension** (default = "active_snws")

`dir_for_parallel_run`
: (previously created) directory for a parallel run iteration. Used in the wrapper function (see ?run_pathfindR) (Default = NULL)

`score_quan_thr`
: active subnetwork score quantile threshold. Must be between 0 and 1 or set to -1 for not filtering. (Default = 0.8)

`sig_gene_thr`
: threshold for the minimum proportion of significant genes in the subnetwork (Default = 0.02) If the number of genes to use as threshold is calculated to be < 2 (e.g. 50 signif. genes x 0.01 = 0.5), the threshold number is set to 2

`search_method`
: algorithm to use when performing active subnetwork search. Options are greedy search (GR), simulated annealing (SA) or genetic algorithm (GA) for the search (default = "GR").

`silent_option`
: boolean value indicating whether to print the messages to the console (FALSE) or not (TRUE, this will print to a temp. file) during active subnetwork search (default = TRUE). This option was added because during parallel runs, the console messages get disorderly printed.

`use_all_positives`
: if TRUE: in GA, adds an individual with all positive nodes. In SA, initializes candidate solution with all positive nodes. (default = FALSE)

`geneInitProbs`
: For SA and GA, probability of adding a gene in initial solution (default = 0.1)

`saTemp0`
: Initial temperature for SA (default = 1.0)

`saTemp1`
: Final temperature for SA (default = 0.01)

`saIter`
: Iteration number for SA (default = 10000)

`gaPop`
: Population size for GA (default = 400)

`gaIter`
: Iteration number for GA (default = 200)

`gaThread`
: Number of threads to be used in GA (default = 5)

`gaCrossover`
: Applies crossover with the given probability in GA (default = 1, i.e. always perform crossover)

`gaMut`
: For GA, applies mutation with given mutation rate (default = 0, i.e. mutation off)

`grMaxDepth`
: Sets max depth in greedy search, 0 for no limit (default = 1)

grSearchDepth	Search depth in greedy search (default = 1)
grOverlap	Overlap threshold for results of greedy search (default = 0.5)
grSubNum	Number of subnetworks to be presented in the results (default = 1000)

Value

A list of genes in every identified active subnetwork that has a score greater than the ‘score_quan_thr‘th quantile and that has at least ‘sig_gene_thr‘ affected genes.

Examples

```
processed_df <- RA_input[1:15, -2]
colnames(processed_df) <- c("GENE", "P_VALUE")
GR_snws <- active_snw_search(input_for_search = processed_df,
                             pin_name_path = "KEGG",
                             search_method = "GR",
                             score_quan_thr = 0.8)
# clean-up
unlink("active_snw_search", recursive = TRUE)
```

---

annotate_term_genes *Annotate the Affected Genes in the Provided Enriched Terms*

Description

Function to annotate the involved affected (input) genes in each term.

Usage

```
annotate_term_genes(
  result_df,
  input_processed,
  genes_by_term = pathfindR.data::kegg_genes
)
```

Arguments

result_df	data frame of enrichment results. The only must-have column is "ID".
input_processed	input data processed via input_processing
genes_by_term	List that contains genes for each gene set. Names of this list are gene set IDs (default = kegg_genes)

Value

The original data frame with two additional columns:

Up_regulated the up-regulated genes in the input involved in the given term's gene set, comma-separated

Down_regulated the down-regulated genes in the input involved in the given term's gene set, comma-separated

Examples

```
example_gene_data <- RA_input
colnames(example_gene_data) <- c("GENE", "CHANGE", "P_VALUE")

annotated_result <- annotate_term_genes(result_df = RA_output,
                                        input_processed = example_gene_data)
```

---

check_java_version *Check Java Version*

Description

Check Java Version

Usage

```
check_java_version(version = NULL)
```

Arguments

version	character vector containing the output of "java -version". If NULL, result of fetch_java_version is used (default = NULL)

Details

this function was adapted from the CRAN package sparklyr

Value

only parses and checks whether the java version is >= 1.8

`cluster_enriched_terms` *Cluster Enriched Terms*

Description

Cluster Enriched Terms

Usage

```
cluster_enriched_terms(
  enrichment_res,
  method = "hierarchical",
  plot_clusters_graph = TRUE,
  use_description = FALSE,
  use_active_snw_genes = FALSE,
  ...
)
```

Arguments

`enrichment_res`	data frame of pathfindR enrichment results. Must-have columns are "Term_Description" (if `use_description = TRUE`) or "ID" (if `use_description = FALSE`), "Down_regulated", and "Up_regulated". If `use_active_snw_genes = TRUE`, "non_Signif_Snw_Genes" must also be provided.
`method`	Either "hierarchical" or "fuzzy". Details of clustering are provided in the corresponding functions `hierarchical_term_clustering`, and `fuzzy_term_clustering`
`plot_clusters_graph`	boolean value indicate whether or not to plot the graph diagram of clustering results (default = TRUE)
`use_description`	Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = `FALSE`)
`use_active_snw_genes`	boolean to indicate whether or not to use non-input active subnetwork genes in the calculation of kappa statistics (default = FALSE, i.e. only use affected genes)
`...`	additional arguments for `hierarchical_term_clustering`, `fuzzy_term_clustering` and `cluster_graph_vis`. See documentation of these functions for more details.

Value

a data frame of clustering results. For "hierarchical", the cluster assignments (Cluster) and whether the term is representative of its cluster (Status) is added as columns. For "fuzzy", terms that are in multiple clusters are provided for each cluster. The cluster assignments (Cluster) and whether the term is representative of its cluster (Status) is added as columns.

See Also

See hierarchical_term_clustering for hierarchical clustering of enriched terms. See fuzzy_term_clustering for fuzzy clustering of enriched terms. See cluster_graph_vis for graph visualization of clustering.

Examples

```
example_clustered <- cluster_enriched_terms(RA_output[1:3, ],
  plot_clusters_graph = FALSE)
example_clustered <- cluster_enriched_terms(RA_output[1:3, ],
  method = "fuzzy", plot_clusters_graph = FALSE)
```

---

cluster_graph_vis *Graph Visualization of Clustered Enriched Terms*

Description

Graph Visualization of Clustered Enriched Terms

Usage

```
cluster_graph_vis(
  clu_obj,
  kappa_mat,
  enrichment_res,
  kappa_threshold = 0.35,
  use_description = FALSE
)
```

Arguments

clu_obj	clustering result (either a matrix obtained via hierarchical_term_clustering or fuzzy_term_clustering 'fuzzy_term_clustering' or a vector obtained via 'hierarchical_term_clustering')
kappa_mat	matrix of kappa statistics (output of create_kappa_matrix)
enrichment_res	data frame of pathfindR enrichment results. Must-have columns are "Term_Description" (if use_description = TRUE) or "ID" (if use_description = FALSE), "Down_regulated", and "Up_regulated". If use_active_snw_genes = TRUE, "non_Signif_Snw_Genes" must also be provided.
kappa_threshold	threshold for kappa statistics, defining strong relation (default = 0.35)
use_description	Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = FALSE)

Value

Plots a graph diagram of clustering results. Each node is an enriched term from 'enrichment_res'. Size of node corresponds to -log(lowest_p). Thickness of the edges between nodes correspond to the kappa statistic between the two terms. Color of each node corresponds to distinct clusters. For fuzzy clustering, if a term is in multiple clusters, multiple colors are utilized.

Examples

```
## Not run:
cluster_graph_vis(clu_obj, kappa_mat, enrichment_res)

## End(Not run)
```

`color_kegg_pathway` *Color hsa KEGG pathway*

Description

Color hsa KEGG pathway

Usage

```
color_kegg_pathway(
  pw_id,
  change_vec,
  normalize_vals = FALSE,
  node_cols = NULL,
  quiet = TRUE
)
```

Arguments

`pw_id`	hsa KEGG pathway id (e.g. hsa05012)
`change_vec`	vector of change values, names should be hsa KEGG gene ids
`normalize_vals`	should change values be normalized (default = `FALSE`)
`node_cols`	low, middle and high color values for coloring the pathway nodes (default = `NULL`). If `node_cols=NULL`, the low, middle and high color are set as "green", "gray" and "red". If all change values are 1e6 (in case no changes are supplied, this dummy value is assigned by `input_processing`), only one color ("#F38F18" if NULL) is used.
`quiet`	If `TRUE`, suppress status messages (if any), and the progress bar while downloading file(s)

Value

list containing:

1. file_path: path to colored hsa KEGG pathway diagram
2. all_key_cols: colors used for each change value bin
3. all_brks: breaks used for separating change values into bins

Examples

```
## Not run:
   pw_id <- "hsa00010"
   change_vec <- c(-2, 4, 6)
   names(change_vec) <- c("hsa:2821", "hsa:226", "hsa:229")
   result <- pathfindR:::color_kegg_pathway(pw_id, change_vec)

## End(Not run)
```

`combined_results_graph` *Combined Results Graph*

Description

Combined Results Graph

Usage

```
combined_results_graph(
  combined_df,
  selected_terms = "common",
  use_description = FALSE,
  layout = "stress",
  node_size = "num_genes"
)
```

Arguments

`combined_df`	Data frame of combined pathfindR enrichment results
`selected_terms`	the vector of selected terms for creating the graph (either IDs or term descriptions). If set to `"common"`, all of the common terms are used. (default = "common")
`use_description`	Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = `FALSE`)
`layout`	The type of layout to create (see `ggraph` for details. Default = "stress")
`node_size`	Argument to indicate whether to use number of significant genes ("num_genes") or the -log10(lowest p value) ("p_val") for adjusting the node sizes (default = "num_genes")

Value

a ggraph object containing the combined term-gene graph. Each node corresponds to an enriched term (orange if common, different shades of blue otherwise), an up-regulated gene (green), a down-regulated gene (red) or a conflicting (i.e. up in one analysis, down in the other or vice versa) gene (gray). An edge between a term and a gene indicates that the given term involves the gene. Size of a term node is proportional to either the number of genes (if `node_size = "num_genes"`) or the -log10(lowest p value) (if `node_size = "p_val"`).

Examples

```
combined_results <- combine_pathfindR_results(RA_output,
                                               RA_comparison_output,
                                               plot_common = FALSE)
g <- combined_results_graph(combined_results, selected_terms = sample(combined_results$ID, 3))
```

---

combine_pathfindR_results

Combine 2 pathfindR Results

Description

Combine 2 pathfindR Results

Usage

```
combine_pathfindR_results(result_A, result_B, plot_common = TRUE)
```

Arguments

result_A	data frame of first pathfindR enrichment results
result_B	data frame of second pathfindR enrichment results
plot_common	boolean to indicate whether or not to plot the term-gene graph of the common terms (default=TRUE)

Value

Data frame of combined pathfindR enrichment results. Columns are:

ID ID of the enriched term

Term_Description Description of the enriched term

Fold_Enrichment_A Fold enrichment value for the enriched term (Calculated using ONLY the input genes)

occurrence_A the number of iterations that the given term was found to enriched over all iterations

lowest_p_A the lowest adjusted-p value of the given term over all iterations

highest_p_A the highest adjusted-p value of the given term over all iterations

Up_regulated_A the up-regulated genes in the input involved in the given term's gene set, comma-separated

Down_regulated_A the down-regulated genes in the input involved in the given term's gene set, comma-separated

Fold_Enrichment_B Fold enrichment value for the enriched term (Calculated using ONLY the input genes)

occurrence_B the number of iterations that the given term was found to enriched over all iterations

lowest_p_B the lowest adjusted-p value of the given term over all iterations

highest_p_B the highest adjusted-p value of the given term over all iterations

Up_regulated_B the up-regulated genes in the input involved in the given term's gene set, comma-separated

Down_regulated_B the down-regulated genes in the input involved in the given term's gene set, comma-separated

combined_p the combined p value (via Fisher's method)

status whether the term is found in both analyses ("common"), found only in the first ("A only") or found only in the second ("B only)

By default, the function also displays the term-gene graph of the common terms

Examples

```
combined_results <- combine_pathfindR_results(RA_output, RA_comparison_output)
```

`create_kappa_matrix` *Create Kappa Statistics Matrix*

Description

Create Kappa Statistics Matrix

Usage

```
create_kappa_matrix(
  enrichment_res,
  use_description = FALSE,
  use_active_snw_genes = FALSE
)
```

Arguments

`enrichment_res`	data frame of pathfindR enrichment results. Must-have columns are "Term_Description" (if `use_description = TRUE`) or "ID" (if `use_description = FALSE`), "Down_regulated", and "Up_regulated". If `use_active_snw_genes = TRUE`, "non_Signif_Snw_Genes" must also be provided.

use_description
: Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = FALSE)

use_active_snw_genes
: boolean to indicate whether or not to use non-input active subnetwork genes in the calculation of kappa statistics (default = FALSE, i.e. only use affected genes)

Value

a matrix of kappa statistics between each term in the enrichment results.

Examples

```
sub_df <- RA_output[1:3, ]
create_kappa_matrix(sub_df)
```

---

download_kegg_png *Download Colored KEGG Diagram PNG*

Description

Download Colored KEGG Diagram PNG

Usage

```
download_kegg_png(pw_url, f_path, quiet = TRUE)
```

Arguments

pw_url	url to download
f_path	local path to save the file
quiet	If TRUE, suppress status messages (if any), and the progress bar while downloading file(s)

Value

download status

`enrichment` *Perform Enrichment Analysis for a Single Gene Set*

Description

Perform Enrichment Analysis for a Single Gene Set

Usage

```
enrichment(
  input_genes,
  genes_by_term = pathfindR.data::kegg_genes,
  term_descriptions = pathfindR.data::kegg_descriptions,
  adj_method = "bonferroni",
  enrichment_threshold = 0.05,
  sig_genes_vec,
  background_genes
)
```

Arguments

`input_genes`	The set of gene symbols to be used for enrichment analysis. In the scope of this package, these are genes that were identified for an active subnetwork
`genes_by_term`	List that contains genes for each gene set. Names of this list are gene set IDs (default = kegg_genes)
`term_descriptions`	Vector that contains term descriptions for the gene sets. Names of this vector are gene set IDs (default = kegg_descriptions)
`adj_method`	correction method to be used for adjusting p-values. (default = "bonferroni")
`enrichment_threshold`	adjusted-p value threshold used when filtering enrichment results (default = 0.05)
`sig_genes_vec`	vector of significant gene symbols. In the scope of this package, these are the input genes that were used for active subnetwork search
`background_genes`	vector of background genes. In the scope of this package, the background genes are taken as all genes in the PIN (see `enrichment_analyses`)

Value

A data frame that contains enrichment results

See Also

`p.adjust` for adjustment of p values. See `run_pathfindR` for the wrapper function of the pathfindR workflow. `hyperg_test` for the details on hypergeometric distribution-based hypothesis testing.

Examples

```
enrichment(input_genes = c("PER1", "PER2", "CRY1", "CREB1"),
           sig_genes_vec = "PER1",
           background_genes = unlist(pathfindR.data::kegg_genes))
```

`enrichment_analyses` *Perform Enrichment Analyses on the Input Subnetworks*

Description

Perform Enrichment Analyses on the Input Subnetworks

Usage

```
enrichment_analyses(
  snws,
  sig_genes_vec,
  pin_name_path = "Biogrid",
  genes_by_term = pathfindR.data::kegg_genes,
  term_descriptions = pathfindR.data::kegg_descriptions,
  adj_method = "bonferroni",
  enrichment_threshold = 0.05,
  list_active_snw_genes = FALSE
)
```

Arguments

`snws`	a list of subnetwork genes (i.e., vectors of genes for each subnetwork)
`sig_genes_vec`	vector of significant gene symbols. In the scope of this package, these are the input genes that were used for active subnetwork search
`pin_name_path`	Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")
`genes_by_term`	List that contains genes for each gene set. Names of this list are gene set IDs (default = kegg_genes)
`term_descriptions`	Vector that contains term descriptions for the gene sets. Names of this vector are gene set IDs (default = kegg_descriptions)
`adj_method`	correction method to be used for adjusting p-values. (default = "bonferroni")
`enrichment_threshold`	adjusted-p value threshold used when filtering enrichment results (default = 0.05)
`list_active_snw_genes`	boolean value indicating whether or not to report the non-significant active subnetwork genes for the active subnetwork which was enriched for the given term with the lowest p value (default = `FALSE`)

Value

a dataframe of combined enrichment results. Columns are:

ID ID of the enriched term

Term_Description Description of the enriched term

Fold_Enrichment Fold enrichment value for the enriched term

p_value p value of enrichment

adj_p adjusted p value of enrichment

support the support (proportion of active subnetworks leading to enrichment over all subnetworks) for the gene set

non_Signif_Snw_Genes (OPTIONAL) the non-significant active subnetwork genes, comma-separated

See Also

enrichment for the enrichment analysis for a single gene set

Examples

```
enr_res <- enrichment_analyses(snws = example_active_snws[1:2],
                               sig_genes_vec = RA_input$Gene.symbol[1:25],
                               pin_name_path = "KEGG")
```

---

enrichment_chart *Create Bubble Chart of Enrichment Results*

Description

This function is used to create a ggplot2 bubble chart displaying the enrichment results.

Usage

```
enrichment_chart(
  result_df,
  top_terms = 10,
  plot_by_cluster = FALSE,
  num_bubbles = 4,
  even_breaks = TRUE
)
```

Arguments

result_df	a data frame that must contain the following columns: **Term_Description** Description of the enriched term **Fold_Enrichment** Fold enrichment value for the enriched term **lowest_p** the lowest adjusted-p value of the given term over all iterations **Up_regulated** the up-regulated genes in the input involved in the given term's gene set, comma-separated **Down_regulated** the down-regulated genes in the input involved in the given term's gene set, comma-separated **Cluster(OPTIONAL)** the cluster to which the enriched term is assigned
top_terms	number of top terms (according to the "lowest_p" column) to plot (default = 10). If plot_by_cluster = TRUE, selects the top top_terms terms per each cluster. Set top_terms = NULL to plot for all terms.If the total number of terms is less than top_terms, all terms are plotted.
plot_by_cluster	boolean value indicating whether or not to group the enriched terms by cluster (works if result_df contains a "Cluster" column).
num_bubbles	number of sizes displayed in the legend # genes (Default = 4)
even_breaks	whether or not to set even breaks for the number of sizes displayed in the legend # genes. If TRUE (default), sets equal breaks and the number of displayed bubbles may be different than the number set by num_bubbles. If the exact number set by num_bubbles is required, set this argument to FALSE

Value

a ggplot2 object containing the bubble chart. The x-axis corresponds to fold enrichment values while the y-axis indicates the enriched terms. Size of the bubble indicates the number of significant genes in the given enriched term. Color indicates the -log10(lowest-p) value. The closer the color is to red, the more significant the enrichment is. Optionally, if "Cluster" is a column of result_df and plot_by_cluster == TRUE, the enriched terms are grouped by clusters.

Examples

```
g <- enrichment_chart(RA_output)
```

---

fetch_gene_set *Fetch Gene Set Objects*

Description

Function for obtaining the gene sets per term and the term descriptions to be used for enrichment analysis.

Usage

```
fetch_gene_set(
  gene_sets = "KEGG",
  min_gset_size = 10,
  max_gset_size = 300,
  custom_genes = NULL,
  custom_descriptions = NULL
)
```

Arguments

gene_sets	Name of the gene sets to be used for enrichment analysis. Available gene sets are "KEGG", "Reactome", "BioCarta", "GO-All", "GO-BP", "GO-CC", "GO-MF", "cell_markers", "mmu_KEGG" or "Custom". If "Custom", the arguments custom_genes and custom_descriptions must be specified. (Default = "KEGG")
min_gset_size	minimum number of genes a term must contain (default = 10)
max_gset_size	maximum number of genes a term must contain (default = 10)
custom_genes	a list containing the genes involved in each custom term. Each element is a vector of gene symbols located in the given custom term. Names should correspond to the IDs of the custom terms.
custom_descriptions	A vector containing the descriptions for each custom term. Names of the vector should correspond to the IDs of the custom terms.

Value

a list containing 2 elements

genes_by_term list of vectors of genes contained in each term

term_descriptions vector of descriptions per each term

Examples

```
KEGG_gset <- fetch_gene_set()
GO_MF_gset <- fetch_gene_set("GO-MF")
```

---

fetch_java_version *Obtain Java Version*

Description

Obtain Java Version

Usage

```
fetch_java_version()
```

Details

this function was adapted from the CRAN package `sparklyr`

Value

character vector containing the output of "java -version"

`filterActiveSnws` *Parse Active Subnetwork Search Output File and Filter the Subnetworks*

Description

Parse Active Subnetwork Search Output File and Filter the Subnetworks

Usage

```
filterActiveSnws(
  active_snw_path,
  sig_genes_vec,
  score_quan_thr = 0.8,
  sig_gene_thr = 0.02
)
```

Arguments

`active_snw_path`	path to the output of an Active Subnetwork Search
`sig_genes_vec`	vector of significant gene symbols. In the scope of this package, these are the input genes that were used for active subnetwork search
`score_quan_thr`	active subnetwork score quantile threshold. Must be between 0 and 1 or set to -1 for not filtering. (Default = 0.8)
`sig_gene_thr`	threshold for the minimum proportion of significant genes in the subnetwork (Default = 0.02) If the number of genes to use as threshold is calculated to be < 2 (e.g. 50 signif. genes x 0.01 = 0.5), the threshold number is set to 2

Value

A list containing `subnetworks`: a list of of genes in every active subnetwork that has a score greater than the `score_quan_thr`th quantile and that contains at least `sig_gene_thr` of significant genes and `scores` the score of each filtered active subnetwork

See Also

See `run_pathfindR` for the wrapper function of the pathfindR enrichment workflow

Examples

```
path2snw_list <- system.file("extdata/resultActiveSubnetworkSearch.txt",
                             package = "pathfindR")
filtered <- filterActiveSnws(active_snw_path = path2snw_list,
                            sig_genes_vec = RA_input$Gene.symbol)
```

`fuzzy_term_clustering` *Heuristic Fuzzy Multiple-linkage Partitioning of Enriched Terms*

Description

Heuristic Fuzzy Multiple-linkage Partitioning of Enriched Terms

Usage

```
fuzzy_term_clustering(
  kappa_mat,
  enrichment_res,
  kappa_threshold = 0.35,
  use_description = FALSE
)
```

Arguments

`kappa_mat`	matrix of kappa statistics (output of `create_kappa_matrix`)
`enrichment_res`	data frame of pathfindR enrichment results. Must-have columns are "Term_Description" (if `use_description = TRUE`) or "ID" (if `use_description = FALSE`), "Down_regulated", and "Up_regulated". If `use_active_snw_genes = TRUE`, "non_Signif_Snw_Genes" must also be provided.
`kappa_threshold`	threshold for kappa statistics, defining strong relation (default = 0.35)
`use_description`	Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = `FALSE`)

Details

The fuzzy clustering algorithm was implemented based on: Huang DW, Sherman BT, Tan Q, et al. The DAVID Gene Functional Classification Tool: a novel biological module-centric algorithm to functionally analyze large gene lists. Genome Biol. 2007;8(9):R183.

Value

a boolean matrix of cluster assignments. Each row corresponds to an enriched term, each column corresponds to a cluster.

Examples

```
## Not run:
fuzzy_term_clustering(kappa_mat, enrichment_res)
fuzzy_term_clustering(kappa_mat, enrichment_res, kappa_threshold = 0.45)

## End(Not run)
```

`get_biogrid_pin` *Retrieve the Requested Release of Organism-specific BioGRID PIN*

Description

Retrieve the Requested Release of Organism-specific BioGRID PIN

Usage

```
get_biogrid_pin(org = "Homo_sapiens", path2pin, release = "4.4.200")
```

Arguments

`org`	organism name. BioGRID naming requires underscores for spaces so "Homo sapiens" becomes "Homo_sapiens", "Mus musculus" becomes "Mus_musculus" etc. See https://wiki.thebiogrid.org/doku.php/statistics for a full list of available organisms (default = "Homo_sapiens")
`path2pin`	the path of the file to save the PIN data. By default, the PIN data is saved in a temporary file
`release`	the requested BioGRID release (default = "4.4.200")

Value

the path of the file in which the PIN data was saved. If `path2pin` was not supplied by the user, the PIN data is saved in a temporary file

---

`get_gene_sets_list` *Retrieve Organism-specific Gene Sets List*

Description

Retrieve Organism-specific Gene Sets List

Usage

```
get_gene_sets_list(
  source = "KEGG",
  org_code = "hsa",
  species = "Homo sapiens",
  collection,
  subcollection = NULL
)
```

Arguments

source	As of this version, either "KEGG", "Reactome" or "MSigDB" (default = "KEGG")
org_code	(Used for "KEGG" only) KEGG organism code for the selected organism. For a full list of all available organisms, see https://www.genome.jp/kegg/catalog/org_list.html
species	(Used for "MSigDB" only) species name, such as Homo sapiens, Mus musculus, etc. See msigdbr_show_species for all the species available in the msigdbr package (default = "Homo sapiens")
collection	(Used for "MSigDB" only) collection. i.e., H, C1, C2, C3, C4, C5, C6, C7.
subcollection	(Used for "MSigDB" only) sub-collection, such as CGP, MIR, BP, etc. (default = NULL, i.e. list all gene sets in collection)

Value

A list containing 2 elements:

- gene_setsA list containing the genes involved in each gene set
- descriptionsA named vector containing the descriptions for each gene set

. For "KEGG" and "MSigDB", it is possible to choose a specific organism. For a full list of all available KEGG organisms, see https://www.genome.jp/kegg/catalog/org_list.html. See msigdbr_show_species for all the species available in the msigdbr package used for obtaining "MSigDB" gene sets. For Reactome, there is only one collection of pathway gene sets.

---

get_kegg_gsets *Retrieve Organism-specific KEGG Pathway Gene Sets*

Description

Retrieve Organism-specific KEGG Pathway Gene Sets

Usage

```
get_kegg_gsets(org_code = "hsa")
```

Arguments

org_code	KEGG organism code for the selected organism. For a full list of all available organisms, see https://www.genome.jp/kegg/catalog/org_list.html

Value

list containing 2 elements:

- gene_setsA list containing the genes involved in each KEGG pathway
- descriptionsA named vector containing the descriptions for each KEGG pathway

---

get_mgsigdb_gsets *Retrieve Organism-specific MSigDB Gene Sets*

Description

Retrieve Organism-specific MSigDB Gene Sets

Usage

```
get_mgsigdb_gsets(species = "Homo sapiens", collection, subcollection = NULL)
```

Arguments

species	species name, such as Homo sapiens, Mus musculus, etc. See msigdbr_show_species for all the species available in the msigdbr package
collection	collection. i.e., H, C1, C2, C3, C4, C5, C6, C7.
subcollection	sub-collection, such as CGP, BP, etc. (default = NULL, i.e. list all gene sets in collection)

Details

this function utilizes the function msigdbr from the msigdbr package to retrieve the 'Molecular Signatures Database' (MSigDB) gene sets (Subramanian et al. 2005 <doi:10.1073/pnas.0506580102>, Liberzon et al. 2015 <doi:10.1016/j.cels.2015.12.004>). Available collections are: H: hallmark gene sets, C1: positional gene sets, C2: curated gene sets, C3: motif gene sets, C4: computational gene sets, C5: GO gene sets, C6: oncogenic signatures and C7: immunologic signatures

Value

Retrieves the MSigDB gene sets and returns a list containing 2 elements:

- gene_setsA list containing the genes involved in each of the selected MSigDB gene sets
- descriptionsA named vector containing the descriptions for each selected MSigDB gene set

get_pin_file *Retrieve Organism-specific PIN data*

Description

Retrieve Organism-specific PIN data

Usage

```
get_pin_file(source = "BioGRID", org = "Homo_sapiens", path2pin, ...)
```

Arguments

source	As of this version, this function is implemented to get data from "BioGRID" only. This argument (and this wrapper function) was implemented for future utility
org	organism name. BioGRID naming requires underscores for spaces so "Homo sapiens" becomes "Homo_sapiens", "Mus musculus" becomes "Mus_musculus" etc. See https://wiki.thebiogrid.org/doku.php/statistics for a full list of available organisms (default = "Homo_sapiens")
path2pin	the path of the file to save the PIN data. By default, the PIN data is saved in a temporary file
...	additional arguments for get_biogrid_pin

Value

the path of the file in which the PIN data was saved. If path2pin was not supplied by the user, the PIN data is saved in a temporary file

Examples

```
## Not run:
pin_path <- get_pin_file()

## End(Not run)
```

get_reactome_gsets *Retrieve Reactome Pathway Gene Sets*

Description

Retrieve Reactome Pathway Gene Sets

Usage

```
get_reactome_gsets()
```

Value

Gets the latest Reactome pathways gene sets in gmt format. Parses the gmt file and returns a list containing 2 elements:

- gene_setsA list containing the genes involved in each Reactome pathway
- descriptionsA named vector containing the descriptions for each Reactome pathway

---

gset_list_from_gmt *Retrieve Gene Sets from GMT-format File*

Description

Retrieve Gene Sets from GMT-format File

Usage

```
gset_list_from_gmt(path2gmt)
```

Arguments

path2gmt	path to the gmt file

Value

list containing 2 elements:

- gene_setsA list containing the genes involved in each gene set
- descriptionsA named vector containing the descriptions for each gene set

---

hierarchical_term_clustering *Hierarchical Clustering of Enriched Terms*

Description

Hierarchical Clustering of Enriched Terms

Usage

```
hierarchical_term_clustering(
  kappa_mat,
  enrichment_res,
  num_clusters = NULL,
  use_description = FALSE,
  clu_method = "average",
  plot_hmap = FALSE,
  plot_dend = TRUE
)
```

Arguments

kappa_mat	matrix of kappa statistics (output of create_kappa_matrix)
enrichment_res	data frame of pathfindR enrichment results. Must-have columns are "Term_Description" (if use_description = TRUE) or "ID" (if use_description = FALSE), "Down_regulated", and "Up_regulated". If use_active_snw_genes = TRUE, "non_Signif_Snw_Genes" must also be provided.
num_clusters	number of clusters to be formed (default = NULL). If NULL, the optimal number of clusters is determined as the number which yields the highest average silhouette width.
use_description	Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = FALSE)
clu_method	the agglomeration method to be used (default = "average", see hclust)
plot_hmap	boolean to indicate whether to plot the kappa statistics clustering heatmap or not (default = FALSE)
plot_dend	boolean to indicate whether to plot the clustering dendrogram partitioned into the optimal number of clusters (default = TRUE)

Details

The function initially performs hierarchical clustering of the enriched terms in enrichment_res using the kappa statistics (defining the distance as 1 -kappa_statistic). Next, the clustering dendrogram is cut into k = 2, 3, ..., n - 1 clusters (where n is the number of terms). The optimal number of clusters is determined as the k value which yields the highest average silhouette width. (if num_clusters not specified)

Value

a vector of clusters for each enriched term in the enrichment results.

Examples

```
## Not run:
hierarchical_term_clustering(kappa_mat, enrichment_res)
hierarchical_term_clustering(kappa_mat, enrichment_res, method = "complete")

## End(Not run)
```

---

hyperg_test *Hypergeometric Distribution-based Hypothesis Testing*

Description

Hypergeometric Distribution-based Hypothesis Testing

Usage

```
hyperg_test(term_genes, chosen_genes, background_genes)
```

Arguments

term_genes	vector of genes in the selected term gene set
chosen_genes	vector containing the set of input genes
background_genes	vector of background genes (i.e. universal set of genes in the experiment)

Details

To determine whether the chosen_genes are enriched (compared to a background pool of genes) in the term_genes, the hypergeometric distribution is assumed and the appropriate p value (the value under the right tail) is calculated and returned.

Value

the p-value as determined using the hypergeometric distribution.

Examples

```
hyperg_test(letters[1:5], letters[2:5], letters)
hyperg_test(letters[1:5], letters[2:10], letters)
hyperg_test(letters[1:5], letters[2:13], letters)
```

---

input_processing *Process Input*

Description

Process Input

Usage

```
input_processing(
  input,
  p_val_threshold = 0.05,
  pin_name_path = "Biogrid",
  convert2alias = TRUE
)
```

Arguments

input	the input data that pathfindR uses. The input must be a data frame with three columns: 1. Gene Symbol (Gene Symbol) 2. Change value, e.g. log(fold change) (OPTIONAL) 3. p value, e.g. adjusted p value associated with differential expression
p_val_threshold	the p value threshold to use when filtering the input data frame. Must a numeric value between 0 and 1. (default = 0.05)
pin_name_path	Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")
convert2alias	boolean to indicate whether or not to convert gene symbols in the input that are not found in the PIN to an alias symbol found in the PIN (default = TRUE) IMPORTANT NOTE: the conversion uses human gene symbols/alias symbols.

Value

This function first filters the input so that all p values are less than or equal to the threshold. Next, gene symbols that are not found in the PIN are identified. If aliases of these gene symbols are found in the PIN, the symbols are converted to the corresponding aliases. The resulting data frame containing the original gene symbols, the updated symbols, change values and p values is then returned.

See Also

See run_pathfindR for the wrapper function of the pathfindR workflow

Examples

```
processed_df <- input_processing(input = RA_input[1:5, ],
                                 pin_name_path = "KEGG")
processed_df <- input_processing(input = RA_input[1:10, ],
                                 pin_name_path = "KEGG",
                                 convert2alias = FALSE)
```

---

input_testing *Input Testing*

Description

Input Testing

Usage

```
input_testing(input, p_val_threshold = 0.05)
```

Arguments

input	the input data that pathfindR uses. The input must be a data frame with three columns: 1. Gene Symbol (Gene Symbol) 2. Change value, e.g. log(fold change) (OPTIONAL) 3. p value, e.g. adjusted p value associated with differential expression
p_val_threshold	the p value threshold to use when filtering the input data frame. Must a numeric value between 0 and 1. (default = 0.05)

Value

Only checks if the input and the threshold follows the required specifications.

See Also

See run_pathfindR for the wrapper function of the pathfindR workflow

Examples

```
input_testing(RA_input, 0.05)
```

---

obtain_colored_url *Obtain URL for a KEGG pathway diagram with a given set of genes marked*

Description

Obtain URL for a KEGG pathway diagram with a given set of genes marked

Usage

```
obtain_colored_url(pw_id, KEGG_gene_ids, fg_cols, bg_cols)
```

Arguments

pw_id	KEGG pathway ID
KEGG_gene_ids	KEGG gene IDs for marking
fg_cols	colors for the text and border
bg_cols	background colors of the objects in a pathway diagram.

Value

download status

obtain_KEGGML_URL *Obtain KGML file for a KEGG pathway (hsa)*

Description

Obtain KGML file for a KEGG pathway (hsa)

Usage

```
obtain_KEGGML_URL(pw_id, pwKGML, quiet = TRUE)
```

Arguments

pw_id	KEGG pathway ID
pwKGML	destination file
quiet	If TRUE, suppress status messages (if any), and the progress bar while downloading file(s)

Value

KGML URL

pathfindR *pathfindR: A package for Enrichment Analysis Utilizing Active Subnetworks*

Description

pathfindR is a tool for active-subnetwork-oriented gene set enrichment analysis. The main aim of the package is to identify active subnetworks in a protein-protein interaction network using a user-provided list of genes and associated p values then performing enrichment analyses on the identified subnetworks, discovering enriched terms (i.e. pathways, gene ontology, TF target gene sets etc.) that possibly underlie the phenotype of interest.

Details

For analysis on non-Homo sapiens organisms, pathfindR offers utility functions for obtaining organism-specific PIN data and organism-specific gene sets data.

pathfindR also offers functionalities to cluster the enriched terms and identify representative terms in each cluster, to score the enriched terms per sample and to visualize analysis results.

See Also

See run_pathfindR for details on the pathfindR active-subnetwork-oriented enrichment analysis See cluster_enriched_terms for details on methods of enriched terms clustering to define clusters of biologically-related terms See score_terms for details on agglomerated score calculation for enriched terms to investigate how a gene set is altered in a given sample (or in cases vs. controls) See term_gene_heatmap for details on visualization of the heatmap of enriched terms by involved genes See term_gene_graph for details on visualizing terms and term-related genes as a graph to determine the degree of overlap between the enriched terms by identifying shared and/or distinct significant genes See UpSet_plot for details on creating an UpSet plot of the enriched terms. See get_pin_file for obtaining organism-specific PIN data and get_gene_sets_list for obtaining organism-specific gene sets data

---

plot_scores *Plot the Heatmap of Score Matrix of Enriched Terms per Sample*

Description

Plot the Heatmap of Score Matrix of Enriched Terms per Sample

Usage

```
plot_scores(
  score_matrix,
  cases = NULL,
  label_samples = TRUE,
  case_title = "Case",
  control_title = "Control",
  low = "green",
  mid = "black",
  high = "red"
)
```

Arguments

score_matrix	Matrix of agglomerated enriched term scores per sample. Columns are samples, rows are enriched terms
cases	(Optional) A vector of sample names that are cases in the case/control experiment. (default = NULL)
label_samples	Boolean value to indicate whether or not to label the samples in the heatmap plot (default = TRUE)
case_title	Naming of the 'Case' group (as in cases) (default = "Case")
control_title	Naming of the 'Control' group (default = "Control")
low	a string indicating the color of 'low' values in the coloring gradient (default = 'green')

mid	a string indicating the color of 'mid' values in the coloring gradient (default = 'black')
high	a string indicating the color of 'high' values in the coloring gradient (default = 'red')

Value

A 'ggplot2' object containing the heatmap plot. x-axis indicates the samples. y-axis indicates the enriched terms. "Score" indicates the score of the term in a given sample. If cases are provided, the plot is divided into 2 facets, named by case_title and control_title.

Examples

```
score_mat <- score_terms(RA_output, RA_exp_mat, plot_hmap = FALSE)
hmap <- plot_scores(score_mat)
```

---

process_pin *Process Data frame of Protein-protein Interactions*

Description

Process Data frame of Protein-protein Interactions

Usage

```
process_pin(pin_df)
```

Arguments

pin_df	data frame of protein-protein interactions with 2 columns: "Interactor_A" and "Interactor_B"

Value

processed PIN data frame (removes self-interactions and duplicated interactions)

`return_pin_path` *Return The Path to Given Protein-Protein Interaction Network (PIN)*

Description

This function returns the absolute path/to/PIN.sif. While the default PINs are "Biogrid", "STRING", "GeneMania", "IntAct", "KEGG" and "mmu_STRING". The user can also use any other PIN by specifying the "path/to/PIN.sif". All PINs to be used in this package must formatted as SIF files: i.e. have 3 columns with no header, no row names and be tab-separated. Columns 1 and 3 must be interactors' gene symbols, column 2 must be a column with all rows consisting of "pp".

Usage

```
return_pin_path(pin_name_path = "Biogrid")
```

Arguments

`pin_name_path`	Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")

Value

The absolute path to chosen PIN.

See Also

See `run_pathfindR` for the wrapper function of the pathfindR workflow

Examples

```
## Not run:
pin_path <- return_pin_path("GeneMania")

## End(Not run)
```

`run_pathfindR` *Wrapper Function for pathfindR - Active-Subnetwork-Oriented Enrichment Analysis*

Description

`run_pathfindR` is the wrapper function for the pathfindR workflow

Usage

```
run_pathfindR(
  input,
  gene_sets = "KEGG",
  min_gset_size = 10,
  max_gset_size = 300,
  custom_genes = NULL,
  custom_descriptions = NULL,
  pin_name_path = "Biogrid",
  p_val_threshold = 0.05,
  visualize_enriched_terms = TRUE,
  max_to_plot = 10,
  convert2alias = TRUE,
  enrichment_threshold = 0.05,
  adj_method = "bonferroni",
  search_method = "GR",
  use_all_positives = FALSE,
  saTemp0 = 1,
  saTemp1 = 0.01,
  saIter = 10000,
  gaPop = 400,
  gaIter = 200,
  gaThread = 5,
  gaCrossover = 1,
  gaMut = 0,
  grMaxDepth = 1,
  grSearchDepth = 1,
  grOverlap = 0.5,
  grSubNum = 1000,
  iterations = 10,
  n_processes = NULL,
  score_quan_thr = 0.8,
  sig_gene_thr = 0.02,
  plot_enrichment_chart = TRUE,
  output_dir = "pathfindR_Results",
  list_active_snw_genes = FALSE,
  silent_option = TRUE
)
```

Arguments

input	the input data that pathfindR uses. The input must be a data frame with three columns: 1. Gene Symbol (Gene Symbol) 2. Change value, e.g. log(fold change) (OPTIONAL) 3. p value, e.g. adjusted p value associated with differential expression
gene_sets	Name of the gene sets to be used for enrichment analysis. Available gene sets are "KEGG", "Reactome", "BioCarta", "GO-All", "GO-BP", "GO-CC",

"GO-MF", "cell_markers", "mmu_KEGG" or "Custom". If "Custom", the arguments `custom_genes` and `custom_descriptions` must be specified. (Default = "KEGG")

`min_gset_size` minimum number of genes a term must contain (default = 10)

`max_gset_size` maximum number of genes a term must contain (default = 10)

`custom_genes` a list containing the genes involved in each custom term. Each element is a vector of gene symbols located in the given custom term. Names should correspond to the IDs of the custom terms.

`custom_descriptions`
A vector containing the descriptions for each custom term. Names of the vector should correspond to the IDs of the custom terms.

`pin_name_path` Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")

`p_val_threshold`
the p value threshold to use when filtering the input data frame. Must a numeric value between 0 and 1. (default = 0.05)

`visualize_enriched_terms`
Boolean value to indicate whether or not to create diagrams for enriched terms (default = TRUE)

`max_to_plot` (necessary only if `gene_sets = "KEGG"` and `visualize_enriched_terms = TRUE`) The number of top hsa kegg pathways to visualize. If `NULL`, visualizes all (default = 10)

`convert2alias` boolean to indicate whether or not to convert gene symbols in the input that are not found in the PIN to an alias symbol found in the PIN (default = TRUE) IMPORTANT NOTE: the conversion uses human gene symbols/alias symbols.

`enrichment_threshold`
adjusted-p value threshold used when filtering enrichment results (default = 0.05)

`adj_method` correction method to be used for adjusting p-values. (default = "bonferroni")

`search_method` algorithm to use when performing active subnetwork search. Options are greedy search (GR), simulated annealing (SA) or genetic algorithm (GA) for the search (default = "GR").

`use_all_positives`
if TRUE: in GA, adds an individual with all positive nodes. In SA, initializes candidate solution with all positive nodes. (default = FALSE)

`saTemp0` Initial temperature for SA (default = 1.0)

`saTemp1` Final temperature for SA (default = 0.01)

`saIter` Iteration number for SA (default = 10000)

`gaPop` Population size for GA (default = 400)

`gaIter` Iteration number for GA (default = 200)

`gaThread` Number of threads to be used in GA (default = 5)

`gaCrossover` Applies crossover with the given probability in GA (default = 1, i.e. always perform crossover)

`gaMut` For GA, applies mutation with given mutation rate (default = 0, i.e. mutation off)

`grMaxDepth` Sets max depth in greedy search, 0 for no limit (default = 1)

`grSearchDepth` Search depth in greedy search (default = 1)

`grOverlap` Overlap threshold for results of greedy search (default = 0.5)

`grSubNum` Number of subnetworks to be presented in the results (default = 1000)

`iterations` number of iterations for active subnetwork search and enrichment analyses (Default = 10)

`n_processes` optional argument for specifying the number of processes used by foreach. If not specified, the function determines this automatically (Default == NULL. Gets set to 1 for Genetic Algorithm)

`score_quan_thr` active subnetwork score quantile threshold. Must be between 0 and 1 or set to -1 for not filtering. (Default = 0.8)

`sig_gene_thr` threshold for the minimum proportion of significant genes in the subnetwork (Default = 0.02) If the number of genes to use as threshold is calculated to be < 2 (e.g. 50 signif. genes x 0.01 = 0.5), the threshold number is set to 2

`plot_enrichment_chart` boolean value. If TRUE, a bubble chart displaying the enrichment results is plotted. (default = TRUE)

`output_dir` the directory to be created where the output and intermediate files are saved (default = "pathfindR_Results")

`list_active_snw_genes` boolean value indicating whether or not to report the non-significant active subnetwork genes for the active subnetwork which was enriched for the given term with the lowest p value (default = `FALSE`)

`silent_option` boolean value indicating whether to print the messages to the console (FALSE) or not (TRUE, this will print to a temp. file) during active subnetwork search (default = TRUE). This option was added because during parallel runs, the console messages get disorderly printed.

Details

This function takes in a data frame consisting of Gene Symbol, log-fold-change and adjusted-p values. After input testing, any gene symbols that are not in the PIN are converted to alias symbols if the alias is in the PIN. Next, active subnetwork search is performed. Enrichment analysis is performed using the genes in each of the active subnetworks. Terms with adjusted-p values lower than `enrichment_threshold` are discarded. The lowest adjusted-p value (over all subnetworks) for each term is kept. This process of active subnetwork search and enrichment is repeated for a selected number of `iterations`, which is done in parallel. Over all iterations, the lowest and the highest adjusted-p values, as well as number of occurrences are reported for each enriched term.

Value

Data frame of pathfindR enrichment results. Columns are:

ID ID of the enriched term

Term_Description Description of the enriched term

Fold_Enrichment Fold enrichment value for the enriched term (Calculated using ONLY the input genes)

occurrence the number of iterations that the given term was found to enriched over all iterations

support the median support (proportion of active subnetworks leading to enrichment within an iteration) over all iterations

lowest_p the lowest adjusted-p value of the given term over all iterations

highest_p the highest adjusted-p value of the given term over all iterations

non_Signif_Snw_Genes (OPTIONAL) the non-significant active subnetwork genes, comma-separated

Up_regulated the up-regulated genes (as determined by 'change value' > 0, if the 'change column' was provided) in the input involved in the given term's gene set, comma-separated. If change column not provided, all affected are listed here.

Down_regulated the down-regulated genes (as determined by 'change value' < 0, if the 'change column' was provided) in the input involved in the given term's gene set, comma-separated

The function also creates an HTML report with the pathfindR enrichment results linked to the visualizations of the enriched terms in addition to the table of converted gene symbols. This report can be found in "`output_dir`/results.html" under the current working directory.

By default, a bubble chart of top 10 enrichment results are plotted. The x-axis corresponds to fold enrichment values while the y-axis indicates the enriched terms. Sizes of the bubbles indicate the number of significant genes in the given terms. Color indicates the -log10(lowest-p) value; the more red it is, the more significant the enriched term is. See `enrichment_chart`.

Warning

Especially depending on the protein interaction network, the algorithm and the number of iterations you choose, "active subnetwork search + enrichment" component of `run_pathfindR` may take a long time to finish.

See Also

`input_testing` for input testing, `input_processing` for input processing, `active_snw_search` for active subnetwork search and subnetwork filtering, `enrichment_analyses` for enrichment analysis (using the active subnetworks), `summarize_enrichment_results` for summarizing the active-subnetwork-oriented enrichment results, `annotate_term_genes` for annotation of affected genes in the given gene sets, `visualize_terms` for visualization of enriched terms, `enrichment_chart` for a visual summary of the pathfindR enrichment results, `foreach` for details on parallel execution of looping constructs, `cluster_enriched_terms` for clustering the resulting enriched terms and partitioning into clusters.

Examples

```
## Not run:
run_pathfindR(RA_input)

## End(Not run)
```

`score_terms` *Calculate Agglomerated Scores of Enriched Terms for Each Subject*

Description

Calculate Agglomerated Scores of Enriched Terms for Each Subject

Usage

```
score_terms(
  enrichment_table,
  exp_mat,
  cases = NULL,
  use_description = FALSE,
  plot_hmap = TRUE,
  ...
)
```

Arguments

`enrichment_table`
: a data frame that must contain the 3 columns below:

  Term_Description Description of the enriched term (necessary if `use_description = TRUE`)

  ID ID of the enriched term (necessary if `use_description = FALSE`)

  Up_regulated the up-regulated genes in the input involved in the given term's gene set, comma-separated

  Down_regulated the down-regulated genes in the input involved in the given term's gene set, comma-separated

`exp_mat`
: the experiment (e.g., gene expression/methylation) matrix. Columns are samples and rows are genes. Column names must contain sample names and row names must contain the gene symbols.

`cases`
: (Optional) A vector of sample names that are cases in the case/control experiment. (default = NULL)

`use_description`
: Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = `FALSE`)

`plot_hmap`
: Boolean value to indicate whether or not to draw the heatmap plot of the scores. (default = TRUE)

`...`
: Additional arguments for `plot_scores` for aesthetics of the heatmap plot

Value

Matrix of agglomerated scores of each enriched term per sample. Columns are samples, rows are enriched terms. Optionally, displays a heatmap of this matrix.

Conceptual Background

For an experiment matrix (containing expression, methylation, etc. values), the rows of which are genes and the columns of which are samples, we denote:

- E as a matrix of size $m \times n$
- G as the set of all genes in the experiment $G = E_{i.}, \ \ i \in [1, m]$
- S as the set of all samples in the experiment $S = E_{j.}, \ \in [1, n]$

We next define the gene score matrix GS (the standardized experiment matrix, also of size $m \times n$) as:

$GS_{gs} = \frac{E_{gs} - \bar{e_g}}{s_g}$

where $g \in G$, $s \in S$, $\bar{e_g}$ is the mean of all values for gene g and $\bar{s_g}$ is the standard deviation of all values for gene g.

We next denote T to be a set of terms (where each $t \in T$ is a set of term-related genes, i.e., $t = \{g_x, ..., g_y\} \subset G$) and finally define the agglomerated term scores matrix TS (where rows correspond to genes and columns corresponds to samples s.t. the matrix has size $|T| \times n$) as:

$TS_{ts} = \frac{1}{|t|} \sum_{g \in t} GS_{gs}$, where $t \in T$ and $s \in S$.

Examples

```
score_matrix <- score_terms(RA_output, RA_exp_mat, plot_hmap = FALSE)
```

`summarize_enrichment_results` *Summarize Enrichment Results*

Description

Summarize Enrichment Results

Usage

```
summarize_enrichment_results(enrichment_res, list_active_snw_genes = FALSE)
```

Arguments

`enrichment_res` a dataframe of combined enrichment results. Columns are:

ID ID of the enriched term

Term_Description Description of the enriched term

Fold_Enrichment Fold enrichment value for the enriched term

p_value p value of enrichment

adj_p adjusted p value of enrichment

non_Signif_Snw_Genes (OPTIONAL) the non-significant active subnetwork genes, comma-separated

`list_active_snw_genes`

boolean value indicating whether or not to report the non-significant active subnetwork genes for the active subnetwork which was enriched for the given term with the lowest p value (default = `FALSE`)

Value

a dataframe of summarized enrichment results (over multiple iterations). Columns are:

ID ID of the enriched term

Term_Description Description of the enriched term

Fold_Enrichment Fold enrichment value for the enriched term

occurrence the number of iterations that the given term was found to enriched over all iterations

support the median support (proportion of active subnetworks leading to enrichment within an iteration) over all iterations

lowest_p the lowest adjusted-p value of the given term over all iterations

highest_p the highest adjusted-p value of the given term over all iterations

non_Signif_Snw_Genes (OPTIONAL) the non-significant active subnetwork genes, comma-separated

Examples

```
## Not run:
summarize_enrichment_results(enrichment_res)

## End(Not run)
```

`term_gene_graph` *Create Term-Gene Graph*

Description

Create Term-Gene Graph

Usage

```
term_gene_graph(
  result_df,
  num_terms = 10,
  layout = "stress",
  use_description = FALSE,
  node_size = "num_genes"
)
```

Arguments

result_df	A dataframe of pathfindR results that must contain the following columns: **Term_Description** Description of the enriched term (necessary if use_description = TRUE) **ID** ID of the enriched term (necessary if use_description = FALSE) **lowest_p** the lowest adjusted-p value of the given term over all iterations **Up_regulated** the up-regulated genes in the input involved in the given term's gene set, comma-separated **Down_regulated** the down-regulated genes in the input involved in the given term's gene set, comma-separated
num_terms	Number of top enriched terms to use while creating the graph. Set to NULL to use all enriched terms (default = 10, i.e. top 10 terms)
layout	The type of layout to create (see ggraph for details. Default = "stress")
use_description	Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = FALSE)
node_size	Argument to indicate whether to use number of significant genes ("num_genes") or the -log10(lowest p value) ("p_val") for adjusting the node sizes (default = "num_genes")

Details

This function (adapted from the Gene-Concept network visualization by the R package enrichplot) can be utilized to visualize which input genes are involved in the enriched terms as a graph. The term-gene graph shows the links between genes and biological terms and allows for the investigation of multiple terms to which significant genes are related. The graph also enables determination of the overlap between the enriched terms by identifying shared and distinct significant term-related genes.

Value

a ggraph object containing the term-gene graph. Each node corresponds to an enriched term (beige), an up-regulated gene (green) or a down-regulated gene (red). An edge between a term and a gene indicates that the given term involves the gene. Size of a term node is proportional to either the number of genes (if node_size = "num_genes") or the -log10(lowest p value) (if node_size = "p_val").

Examples

```
p <- term_gene_graph(RA_output)
p <- term_gene_graph(RA_output, num_terms = 5)
p <- term_gene_graph(RA_output, node_size = "p_val")
```

`term_gene_heatmap` *Create Terms by Genes Heatmap*

Description

Create Terms by Genes Heatmap

Usage

```
term_gene_heatmap(
  result_df,
  genes_df,
  num_terms = 10,
  use_description = FALSE,
  low = "green",
  mid = "black",
  high = "red",
  ...
)
```

Arguments

`result_df`	A dataframe of pathfindR results that must contain the following columns: **Term_Description** Description of the enriched term (necessary if `use_description = TRUE`) **ID** ID of the enriched term (necessary if `use_description = FALSE`) **lowest_p** the highest adjusted-p value of the given term over all iterations **Up_regulated** the up-regulated genes in the input involved in the given term's gene set, comma-separated **Down_regulated** the down-regulated genes in the input involved in the given term's gene set, comma-separated
`genes_df`	the input data that was used with `run_pathfindR`. It must be a data frame with 3 columns: 1. Gene Symbol (Gene Symbol) 2. Change value, e.g. log(fold change) (optional) 3. p value, e.g. adjusted p value associated with differential expression The change values in this data frame are used to color the affected genes
`num_terms`	Number of top enriched terms to use while creating the plot. Set to `NULL` to use all enriched terms (default = 10)

use_description
: Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = FALSE)

low
: a string indicating the color of 'low' values in the coloring gradient (default = 'green')

mid
: a string indicating the color of 'mid' values in the coloring gradient (default = 'black')

high
: a string indicating the color of 'high' values in the coloring gradient (default = 'red')

...
: additional arguments for input_processing (used if genes_df is provided)

Value

a ggplot2 object of a heatmap where rows are enriched terms and columns are involved input genes. If genes_df is provided, colors of the tiles indicate the change values.

Examples

```
term_gene_heatmap(RA_output, num_terms = 3)
```

UpSet_plot *Create UpSet Plot of Enriched Terms*

Description

Create UpSet Plot of Enriched Terms

Usage

```
UpSet_plot(
  result_df,
  genes_df,
  num_terms = 10,
  method = "heatmap",
  use_description = FALSE,
  low = "green",
  mid = "black",
  high = "red",
  ...
)
```

Arguments

result_df	A dataframe of pathfindR results that must contain the following columns: **Term_Description** Description of the enriched term (necessary if use_description = TRUE) **ID** ID of the enriched term (necessary if use_description = FALSE) **lowest_p** the highest adjusted-p value of the given term over all iterations **Up_regulated** the up-regulated genes in the input involved in the given term's gene set, comma-separated **Down_regulated** the down-regulated genes in the input involved in the given term's gene set, comma-separated
genes_df	the input data that was used with run_pathfindR. It must be a data frame with 3 columns: 1. Gene Symbol (Gene Symbol) 2. Change value, e.g. log(fold change) (optional) 3. p value, e.g. adjusted p value associated with differential expression The change values in this data frame are used to color the affected genes
num_terms	Number of top enriched terms to use while creating the plot. Set to NULL to use all enriched terms (default = 10)
method	the option for producing the plot. Options include "heatmap", "boxplot" and "barplot". (default = "heatmap")
use_description	Boolean argument to indicate whether term descriptions (in the "Term_Description" column) should be used. (default = FALSE)
low	a string indicating the color of 'low' values in the coloring gradient (default = 'green')
mid	a string indicating the color of 'mid' values in the coloring gradient (default = 'black')
high	a string indicating the color of 'high' values in the coloring gradient (default = 'red')
...	additional arguments for input_processing (used if genes_df is provided)

Value

UpSet plots are plots of the intersections of sets as a matrix. This function creates a ggplot object of an UpSet plot where the x-axis is the UpSet plot of intersections of enriched terms. By default (i.e. method = "heatmap") the main plot is a heatmap of genes at the corresponding intersections, colored by up/down regulation (if genes_df is provided, colored by change values). If method = "barplot", the main plot is bar plots of the number of genes at the corresponding intersections. Finally, if method = "boxplot" and if genes_df is provided, then the main plot displays the boxplots of change values of the genes at the corresponding intersections.

Examples

```
UpSet_plot(RA_comparison_output)
```

`visualize_active_subnetworks` *Visualize Active Subnetworks*

Description

Visualize Active Subnetworks

Usage

```
visualize_active_subnetworks(
  active_snw_path,
  genes_df,
  pin_name_path = "Biogrid",
  num_snws,
  layout = "stress",
  score_quan_thr = 0.8,
  sig_gene_thr = 0.02,
  ...
)
```

Arguments

`active_snw_path`	path to the output of an Active Subnetwork Search
`genes_df`	the input data that was used with `run_pathfindR`. It must be a data frame with 3 columns: 1. Gene Symbol (Gene Symbol) 2. Change value, e.g. log(fold change) (optional) 3. p value, e.g. adjusted p value associated with differential expression The change values in this data frame are used to color the affected genes
`pin_name_path`	Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")
`num_snws`	number of top subnetworks to be visualized (leave blank if you want to visualize all subnetworks)
`layout`	The type of layout to create (see `ggraph` for details. Default = "stress")
`score_quan_thr`	active subnetwork score quantile threshold. Must be between 0 and 1 or set to -1 for not filtering. (Default = 0.8)
`sig_gene_thr`	threshold for the minimum proportion of significant genes in the subnetwork (Default = 0.02) If the number of genes to use as threshold is calculated to be < 2 (e.g. 50 signif. genes x 0.01 = 0.5), the threshold number is set to 2
`...`	additional arguments for `input_processing`

Value

a list of ggplot objects of graph visualizations of identified active subnetworks. Green nodes are down-regulated genes, reds are up-regulated genes and yellows are non-input genes

Examples

```
path2snw_list <- system.file("extdata/resultActiveSubnetworkSearch.txt",
                             package = "pathfindR")
# visualize top 2 active subnetworks
g_list <- visualize_active_subnetworks(active_snw_path = path2snw_list,
                                       genes_df = RA_input[1:10, ],
                                       pin_name_path = "KEGG",
                                       num_snws = 2)
```

---

visualize_hsa_KEGG *Visualize Human KEGG Pathways*

Description

Visualize Human KEGG Pathways

Usage

```
visualize_hsa_KEGG(
  hsa_kegg_ids,
  input_processed,
  max_to_plot = NULL,
  normalize_vals = FALSE,
  node_cols = NULL,
  quiet = TRUE,
  key_gravity = "northeast",
  logo_gravity = "southeast"
)
```

Arguments

hsa_kegg_ids	hsa KEGG ids of pathways to be colored and visualized
input_processed	input data processed via input_processing
max_to_plot	The number of hsa kegg pathways (from beginning until the max_to_plotth id) to visualize. If NULL, visualizes all (default = NULL)
normalize_vals	should change values be normalized (default = FALSE)
node_cols	low, middle and high color values for coloring the pathway nodes (default = NULL). If node_cols=NULL, the low, middle and high color are set as "green", "gray" and "red". If all change values are 1e6 (in case no changes are supplied, this dummy value is assigned by input_processing), only one color ("#F38F18" if NULL) is used.

quiet	If TRUE, suppress status messages (if any), and the progress bar while downloading file(s)
key_gravity	gravity value (character) for the color key legend placement (see gravity_types)
logo_gravity	gravity value (character) for the logo placement (see gravity_types)

Value

Creates colored visualizations of the enriched human KEGG pathways and saves them in the folder "term_visualizations" under the current working directory.

See Also

See visualize_terms for the wrapper function for creating enriched term diagrams. See run_pathfindR for the wrapper function of the pathfindR enrichment workflow.

Examples

```
## Not run:
visualize_hsa_KEGG(hsa_kegg_ids, input_processed)

## End(Not run)
```

---

visualize_terms *Create Diagrams for Enriched Terms*

Description

Create Diagrams for Enriched Terms

Usage

```
visualize_terms(
  result_df,
  input_processed = NULL,
  hsa_KEGG = TRUE,
  pin_name_path = "Biogrid",
  ...
)
```

Arguments

result_df	Data frame of enrichment results. Must-have columns for KEGG human pathway diagrams (hsa_kegg = TRUE) are: "ID" and "Term_Description". Must-have columns for the rest are: "Term_Description", "Up_regulated" and "Down_regulated"
input_processed	input data processed via input_processing, not necessary when hsa_KEGG = FALSE

hsa_KEGG	boolean to indicate whether human KEGG gene sets were used for enrichment analysis or not (default = TRUE)
pin_name_path	Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")
...	additional arguments for visualize_hsa_KEGG (used when hsa_kegg = TRUE)

Details

For hsa_KEGG = TRUE, KEGG human pathway diagrams are created, affected nodes colored by up/down regulation status. For other gene sets, interactions of affected genes are determined (via a shortest-path algorithm) and are visualized (colored by change status) using igraph.

Value

Depending on the argument hsa_KEGG, creates visualization of interactions of genes involved in the list of enriched terms in result_df and saves them in the folder "term_visualizations" under the current working directory.

See Also

See visualize_hsa_KEGG for the visualization function of human KEGG diagrams. See visualize_term_interactions for the visualization function that generates diagrams showing the interactions of input genes in the PIN. See run_pathfindR for the wrapper function of the pathfindR workflow.

Examples

```
## Not run:
visualize_terms(result_df, input_processed)
visualize_terms(result_df, hsa_KEGG = FALSE, pin_name_path = "IntAct")

## End(Not run)
```

---

visualize_term_interactions

Visualize Interactions of Genes Involved in the Given Enriched Terms

---

Description

Visualize Interactions of Genes Involved in the Given Enriched Terms

Usage

```
visualize_term_interactions(result_df, pin_name_path)
```

Arguments

result_df	Data frame of enrichment results. Must-have columns are: "Term_Description", "Up_regulated" and "Down_regulated"
pin_name_path	Name of the chosen PIN or path/to/PIN.sif. If PIN name, must be one of c("Biogrid", "STRING", "GeneMania", "IntAct", "KEGG", "mmu_STRING"). If path/to/PIN.sif, the file must comply with the PIN specifications. (Default = "Biogrid")

Details

The following steps are performed for the visualization of interactions of genes involved for each enriched term:

1. shortest paths between all affected genes are determined (via igraph)
2. the nodes of all shortest paths are merged
3. the PIN is subsetted using the merged nodes (genes)
4. using the PIN subset, the graph showing the interactions is generated
5. the final graph is visualized using igraph, colored by changed status (if provided), and is saved as a PNG file.

Value

Creates PNG files visualizing the interactions of genes involved in the given enriched terms (annotated in the result_df) in the PIN used for enrichment analysis (specified by pin_name_path). The PNG files are saved in the folder "term_visualizations" under the current working directory.

See Also

See visualize_terms for the wrapper function for creating enriched term diagrams. See run_pathfindR for the wrapper function of the pathfindR enrichment workflow.

Examples

```
## Not run:
visualize_term_interactions(result_df, pin_name_path = "IntAct")

## End(Not run)
```

Index