GeneWalk Documentation

Release 1.5.1

GeneWalk Developers

Feb 24, 2021

Contents

CHAPTER 1

GeneWalk modules reference

1.1 Process gene lists and map between ID types (genewalk.gene_lists)

genewalk.gene_lists.**map_ensembl_ids**(*ensembl_ids*, *gene_mapper*)
Return references based on a list of Ensembl IDs.

genewalk.gene_lists.**map_entrez_human**(*entrez_ids*, *gene_mapper*)
Return references based on human Entrez gene IDs.

genewalk.gene_lists.**map_entrez_mouse**(*entrez_ids*, *gene_mapper*)
Return references based on mouse Entrez gene IDs.

genewalk.gene_lists.**map_hgnc_ids**(*hgnc_ids*, *gene_mapper*)
Return references based on a list of HGNC IDs.

genewalk.gene_lists.**map_hgnc_symbols**(*hgnc_symbols*, *gene_mapper*)
Return references based on a list of HGNC symbols.

genewalk.gene_lists.**map_mgi_ids**(*mgi_ids*, *gene_mapper*)
Return references based on a list of MGI IDs.

genewalk.gene_lists.**map_rgd_ids**(*rgd_ids*, *gene_mapper*)
Return references based on a list of RGD IDs.

genewalk.gene_lists.**read_gene_list**(*fname*, *id_type*, *resource_manager*)
Return references for genes from a file with the given ID type.

Parameters

- **fname** (*str*) – The name of the file containing the list of genes. Each line of the file corresponds to a single gene.
- **id_type** (*str*) – The type of identifier contained in each line of the gene list file. Possible values are: hgnc_symbol, hgnc_id, ensembl_id, mgi_id.

- **resource_manager** (genewalk.resources.ResourceManager) – ResourceManager object, used to obtain entrez-mgi mappings if necessary.

Returns A dictionary of references with keys including HGNCSYMBOL, HGNC, UP, and if id_type is mgi_id, MGI, with values corresponding to the identifiers of the provided list of genes.

Return type dict

1.2 Networkx MultiGraph assembler (genewalk.nx_mg_assembler)

class genewalk.nx_mg_assembler.**IndraNxMgAssembler**(*genes*, *stmts*, *resource_manager=None*)

Bases: *genewalk.nx_mg_assembler.NxMgAssembler*

The IndraNxMgAssembler assembles INDRA Statements and GO ontology / annotations into a networkx (undirected) MultiGraph including edge attributes. This code is based on INDRA's SifAssembler http://indra.readthedocs.io/en/latest/_modules/indra/assemblers/sif_assembler.html

Parameters **stmts** (*list[indra.statements.Statement]*) – A list of INDRA Statements to be added to the assembler's list of Statements.

graph
A GeneWalk Network that is assembled by this assembler.

Type networkx.MultiGraph

add_agent_node(*agent*)
Add a node corresponding to an INDRA Agent.

add_fplx_edges()
Add edges between gene nodes and families/complexes they are part of.

add_indra_edges()
Add edges between gene nodes and GO nodes based on INDRA Statements.

node2stmts(*node_key*)
Return the INDRA Statements given the key of a graph node.

class genewalk.nx_mg_assembler.**NxMgAssembler**(*genes*, *resource_manager=None*)

Bases: object

Class which assembles a networkx MultiGraph based on a list of genes.

Parameters **genes** (*list of dict*) – A list of gene references based on which the graph is assembled.

graph
The assembled graph containing links for interactions between genes, GO annotations for genes, and the GO ontology.

Type networkx.MultiGraph

add_go_annotations()
Add edges between gene nodes and GO nodes based on GO annotations.

add_go_ontology()
Add edges between GO nodes based on the GO ontology.

node2edges(*node_key*)
Return the edges corresponding to a node.

save_graph(*fname*)
Save the file into a GraphML file.

Parameters **fname** (*str*) – The name of the file to save the graph into.

class genewalk.nx_mg_assembler.**PcNxMgAssembler**(*genes*, *resource_manager=None*)
Bases: *genewalk.nx_mg_assembler.NxMgAssembler*

The PcNxMgAssembler assembles a GeneWalk Network with gene reactions from Pathway Commons and GO ontology and annotations into a networkx (undirected) MultiGraph including edge attributes.

Parameters **genes** (*list fo dict*) – A list of gene references based on which the network is assembled.

graph
A GeneWalk Network that is assembled by this assembler.

Type networkx.MultiGraph

add_pc_edges()
Add edges between gene nodes based on PathwayCommons interactions.

class genewalk.nx_mg_assembler.**UserNxMgAssembler**(*genes*, *resource_manager*, *filepath*, *gwn_format='el'*)
Bases: *genewalk.nx_mg_assembler.NxMgAssembler*

Loads a user-provided GeneWalk Network from a given file.

Parameters

- **filepath** (*str*) – Path to the user-provided genewalk network file, assumed to contain gene symbols and GO IDs. See gwn_format for supported format details.
- **gwn_format** (*Optional[str]*) – 'el' (default, edge list: nodeA nodeB (if more columns present: interpreted as edge attributes) or 'sif' (simple interaction format: nodeA,<relationship type>,nodeB). Do not include column headers.

graph
A GeneWalk Network that is loaded by this assembler.

Type networkx.MultiGraph

add_network_edges()
Assemble the GeneWalk Network from the user-provided file path.

genewalk.nx_mg_assembler.**load_network**(*network_type*, *network_file*, *genes*, *resource_manager=None*)
Return a network assembler of the given type based on a set of genes.

Parameters

- **network_type** (*str*) – The type of the network to be constructed.
- **network_file** (*str*) – The path to a file containing information to construct the network.
- **genes** (*list*) – A list of gene references.
- **resource_manager** (Optional[*genewalk.resources.ResourceManager*]) – A resource manager object which, if specified, is used to get the resource files. Otherwise, the default resource manager is used.

Returns An instance of an NxMgAssembler containing the assembled networkx MultiGraph as its graph attribute.

Return type *genewalk.nx_mg_assembler.NxMgAssembler*

1.3 Run DeepWalk on a GeneWalk graph (genewalk.deepwalk)

This module implements a wrapper around DeepWalk as a class. The class contains a graph used as a basis for running the Deepwalk algorithm. It also implements a method to run a given number of walks and save the walks as an attribute of the instance.

class genewalk.deepwalk.**DeepWalk**(*graph*, *walk_length=10*, *niter=100*)

Bases: object

Perform DeepWalk (node2vec), i.e., unbiased random walk over nodes on an undirected networkx MultiGraph.

Parameters

- **graph** (*networkx.MultiGraph*) – A networkx multigraph to be used as the basis for DeepWalk.
- **walk_length** (*Optional[int]*) – The length of each random walk on the graph. Default: 10
- **niter** (*Optional[int]*) – The number of iterations for each node to run (this is multiplied by the number of neighbors of the node when determining the overall number of walks to start from a given node). Default: 100

walks

A list of walks.

Type list

get_walks(*workers=1*)

Generates walks (sentences) sampled by an (unbiased) random walk over the networkx MultiGraph.

Parameters **workers** (*Optional[int]*) – The number of workers to use when running random walks. If greater than 1, multiprocessing is used to speed up random walk generation. Default: 1

word2vec(*sg=1*, *size=8*, *window=1*, *min_count=1*, *negative=5*, *workers=1*, *sample=0*)

Set the model based on Word2Vec Source: https://radimrehurek.com/gensim/models/word2vec.html

Note that his function sets the model attribute if the DeepWalk object and doesn't return a value.

Parameters

- **sg** (*Optional[int] {1, 0}*) – Defines the training algorithm. If 1, skip-gram is employed; otherwise, CBOW is used. For GeneWalk this is set to 1.
- **size** (*Optional[int]*) – Dimensionality of the node vectors. Default for GeneWalk is 8.
- **window** (*Optional[int]*) – a.k.a. context size. Maximum distance between the current and predicted word within a sentence. For GeneWalk this is set to 1 to assess directly connected nodes only.
- **min_count** (*Optional[int]*) – Ignores all words with total frequency lower than this. For GeneWalk this is set to 0.
- **negative** (*Optional[int]*) – If > 0, negative sampling will be used, the int for negative specifies how many "noise words" should be drawn (usually between 5-20). If set to 0, no negative sampling is used. Default for GeneWalk is 5.
- **workers** (*Optional[int]*) – Use these many worker threads to train the model (=faster training with multicore machines).

- **sample** (*Optional[float]*) – The threshold for configuring which higher-frequency words are randomly downsampled, useful range is (0, 1e-5). parameter t in eq 5 Mikolov et al. For GeneWalk this is set to 0.

genewalk.deepwalk.**run_single_walk**(*start_node*, *graph*, *length*)
Run a single random walk on a graph from a given start node.

Parameters

- **graph** (*networks.MultiGraph*) – The graph on which the random walk is to be run.
- **start_node** (*str*) – The identifier of the node from which the random walk starts.
- **length** (*int*) – The length of the random walk.

Returns A path of the given length, with each element corresponding to a node along the path.

Return type list of str

genewalk.deepwalk.**run_walks**(*graph*, ***kwargs*)
Run random walks and get node vectors on a given graph.

Parameters

- **graph** (*networkx.MultiGraph*) – The graph on which random walks are going to be run and node vectors calculated.
- ****kwargs** – Key word arguments passed as the arguments of the DeepWalk constructor, as well as the get_walks method and the word2vec method. See the DeepWalk class documentation for more information on these.

Returns A DeepWalk instance whose walks attribute contains the list of random walks produced on the graph.

Return type *genewalk.deepwalk.DeepWalk*

genewalk.deepwalk.**run_walks_for_node**(*node*, *graph*, *niter*, *walk_length*)
Run all random walks starting from a given node.

Parameters

- **node** (*str*) – The identifier of the node from which the walks start.
- **graph** (*networks.MultiGraph*) – The graph on which the random walks are to be run.
- **niter** (*int*) – The number of iterations to run for gene nodes.
- **walk_length** (*int*) – The length of the walk.

Returns A list of random walks starting from the given node.

Return type list of list of str

1.4 Calculate null distributions (genewalk.null_distributions)

This module implements functions related to the construction of a null distribution for GeneWalk networks.

genewalk.null_distributions.**get_null_distributions**(*rg*, *nv*)
Return a distribution with similarity values between (random) node vectors originating from the input randomized graph.

genewalk.null_distributions.**get_rand_graph**(*mg*)
Return a random graph with the same degree distribution as the input.

Parameters **mg** (*networkx.MultiGraph*) – An input graph based on which a random graph is generated.

Returns A random graph whose degree distribution matches that of the output.

Return type networkx.MultiGraph

1.5 Get relevant INDRA Statements (genewalk.get_indra_stmts)

This script creates a pickle file of INDRA Statements for a specific set of genes, based on a pre-assembled network of interactions produced by INDRA. It loads the INDRA interactions as a pandas DataFrame and, filters it to the genes and familiex/complexes of interest, as well as targeted biological processes. It downloads the relevant Statement objects for reference and dumps them into a pickle file.

genewalk.get_indra_stmts.**download_statements**(*df*, *ev_limit=5*)
Download the INDRA Statements corresponding to entries in a data frame.

genewalk.get_indra_stmts.**dump_pickle**(*stmts*, *fname*)
Dump a list of Statements into a picke file.

genewalk.get_indra_stmts.**filter_to_genes**(*df*, *genes*, *fplx_terms*)
Filter a data frame of INDRA Statements given gene and FamPlex IDs.

genewalk.get_indra_stmts.**get_famplex_links**(*df*, *fname*)
Given a list of INDRA Statements, construct FamPlex links.

genewalk.get_indra_stmts.**get_famplex_terms**(*genes*)
Get a list of associated FamPlex IDs from a list of gene IDs.

genewalk.get_indra_stmts.**load_genes**(*fname*)
Return a list of genes IDs from a file with lines like HGNC:123.

genewalk.get_indra_stmts.**load_indra_df**(*fname*)
Return an INDRA Statement data frame from a pickle file.

genewalk.get_indra_stmts.**load_mouse_genes**(*fname*)
Return a list of human genes based on a table of mouse genes.

1.6 Perform statistics and output results (genewalk.perform_statistics)

class genewalk.perform_statistics.**GeneWalk**(*graph*, *genes*, *nvs*, *null_dist*, *gene_id_type='hgnc_symbol'*)

Bases: object

GeneWalk object that generates the final output list of significant GO terms for each gene in the input list with genes of interest from an experiment, for example differentially expressed genes or CRISPR screen hits. If an input gene is not in the output file, this could have the following reasons: 1) No corresponding HGNC gene symbol, HGNC:ID and/or UniProt:ID could be identified (for all –id_type values except custom). All are required to map human genes and assemble their GO annotations. See the genewalk log file for all genes filtered out this way.

2) (if alpha_FDR set to < 1) no GO terms were significant at the chosen significance level alpha_FDR.

3) (in case of mouse or rat genes) no mapped human ortholog was identified.

See the genewalk log file to see all genes filtered out because of 1) or 3).

If a gene is listed in the output file with a value >= 0 in the column ncon_gene but without any listed GO annotations: no GO annotations (with the right GO evidence codes) could be retrieved. If a gene is listed but has no ncon_gene value (NaN) in the output file: the gene was correctly mapped but could not be included in the GeneWalk network. This scenario is uncommon and warrants further inspection.

Parameters

- **graph** (*networkx.MultiGraph*) – GeneWalk network for which the statistics are calculated.
- **genes** (*list of dict*) – List of gene references for relevant genes.
- **nvs** (*list of dict*) – Node vectors for nodes in the graph.
- **null_dist** (*np.array*) – Similarity random (null) distribution.
- **gene_id_type** (*Optional[str]*) – The type of gene IDs that were the basis of doing the analysis. In case of mgi_id, rgd_id or ensembl_id, we prepend a column to the table for MGI, RGD, or ENSEMBL IDs, respectively. If custom, the input genes were not mapped to human genes, so the hgnc_symbol and hgnc_id columns are not present in the output. Default: hgnc_symbol

generate_output(*alpha_fdr=1*)

Main function of GeneWalk object that generates the final GeneWalk output table (in csv format).

Parameters **alpha_fdr** (*Optional[float]*) – Significance level for FDR [0,1] (default=1, i.e. all GO terms and their statistics are output). If set to a lower value, only connected GO terms with mean padj < alpha_FDR are output.

get_gene_attribs(*gene*)

Return an attribute dict for a given gene.

get_go_attribs(*gene_attribs*, *nv*, *alpha_fdr*)

Return GO entries and their attributes for a given gene.

global_fdr(*df*, *alpha_fdr*)

Determine the global_padj values through FDR multiple testing correction over all gene - GO annotation pairs present in the output file.

psim(*sim*)

Determine the p-value of the experimental similarity by determining its percentile, i.e. the normalized rank, in the null distribution with random similarity values.

1.7 Plotting results (genewalk.plot)

1.8 Manage resource files (genewalk.resources)

class genewalk.resources.**ResourceManager**(*base_folder=None*)

Bases: object

Class to manage the download, caching and access of resource files.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

Index

M

N

P

R

S

U

W