# Package 'wppi'

April 24, 2025

**Type** Package

**Title** Weighting protein-protein interactions

**Version** 1.16.0

**Description** Protein-protein interaction data is essential for omics data analysis and modeling. Database knowledge is general, not specific for cell type, physiological condition or any other context determining which connections are functional and contribute to the signaling. Functional annotations such as Gene Ontology and Human Phenotype Ontology might help to evaluate the relevance of interactions. This package predicts functional relevance of protein-protein interactions based on functional annotations such as Human Protein Ontology and Gene Ontology, and prioritizes genes based on network topology, functional scores and a path search algorithm.

**License** MIT + file LICENSE

**URL** https://github.com/AnaGalhoz37/wppi

**BugReports** https://github.com/AnaGalhoz37/wppi/issues

**biocViews** GraphAndNetwork, Network, Pathways, Software, GeneSignaling, GeneTarget, SystemsBiology, Transcriptomics, Annotation

**Encoding** UTF-8

**VignetteBuilder** knitr

**Depends** R(>= 4.1)

**Imports** dplyr, igraph, logger, methods, magrittr, Matrix, OmnipathR(>= 2.99.8), progress, purrr, rlang, RCurl, stats, tibble, tidyr

**Suggests** knitr, testthat, rmarkdown

**RoxygenNote** 7.1.1

**NeedsCompilation** no

**git_url** https://git.bioconductor.org/packages/wppi

**git_branch** RELEASE_3_21

**git_last_commit** c49fef6

**git_last_commit_date** 2025-04-15

**Repository** Bioconductor 3.21

**Date/Publication** 2025-04-23

**Author** Ana Galhoz [cre, aut] (ORCID: <https://orcid.org/0000-0001-7402-5292>),
Denes Turei [aut] (ORCID: <https://orcid.org/0000-0002-7249-9379>),
Michael P. Menden [aut] (ORCID:
<https://orcid.org/0000-0003-0267-5792>),
Albert Krewinkel [ctb, cph] (pagebreak Lua filter)

**Maintainer** Ana Galhoz <ana.galhoz@helmholtz-muenchen.de>

# Contents

common_neighbors . . . . . 2
count_genes . . . . . 3
filter_annot_with_network . . . . . 4
functional_annot . . . . . 5
graph_from_op . . . . . 6
in_omnipath . . . . . 6
prioritization_genes . . . . . 7
process_annot . . . . . 9
random_walk . . . . . 10
score_candidate_genes_from_PPI . . . . . 11
subgraph_op . . . . . 13
weighted_adj . . . . . 14
wppi . . . . . 15
wppi_data . . . . . 16
wppi_go_data . . . . . 17
wppi_hpo_data . . . . . 18
wppi_omnipath_data . . . . . 19

**Index** **20**

---

`common_neighbors` *Shared neighbors of connected vertices*

---

## Description

For each interacting pair of proteins in the PPI network, store the nodes of the common neighbors. This function works for any igraph graph.

## Usage

```
common_neighbors(graph_op)
```

## Arguments

| | |
|---|---|
| `graph_op` | Igraph object based on OmniPath PPI interactions from `graph_from_op`. |

## Value

Data frame (tibble) with igraph vertex IDs of connected pairs of vertices (source and target), a list column with the IDs of their common neighbors, and a column with the number of neighbors.

## See Also

graph_from_op

## Examples

```
graph_op <- graph_from_op(wppi_omnipath_data())
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
graph_op_1 <- subgraph_op(graph_op, genes_interest, 1)
shared_neighbors <- common_neighbors(graph_op_1)
```

---

# count_genes *Number of total genes in an ontology database*

---

## Description

Number of total genes in an ontology database

## Usage

```
count_genes(data_annot)
```

## Arguments

| | |
|---|---|
| data_annot | Data frame (tibble) of GO or HPO datasets from wppi_go_data or wppi_hpo_data. |

## Value

Number of total unique genes in each ontology database.

## See Also

- wppi_go_data
- wppi_hpo_data

## Examples

```
go <- wppi_go_data()
count_genes(go)
# [1] 19712
```

`filter_annot_with_network` *Filter ontology datasets using PPI network object*

## Description

Filter ontology datasets using PPI network object

## Usage

```
filter_annot_with_network(data_annot, graph_op)
```

## Arguments

| | |
|---|---|
| `data_annot` | Data frame (tibble) of GO or HPO datasets from `wppi_go_data` or `wppi_hpo_data`. |
| `graph_op` | Igraph graph object obtained from built OmniPath PPI of genes of interest and x-degree neighbors. |

## Value

Data frame (tibble) of GO or HPO datasets filtered based on proteins available in the igraph object.

## See Also

- `wppi_go_data`
- `wppi_hpo_data`
- `graph_from_op`

## Examples

```
# Get GO database
GO_data <- wppi_go_data()
# Create igraph object based on genes of interest and first neighbors
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
graph_op <- graph_from_op(wppi_omnipath_data())
graph_op_1 <- subgraph_op(graph_op, genes_interest, 1)
# Filter GO data
GO_data_filtered <- filter_annot_with_network(GO_data, graph_op_1)
```

`functional_annot` *Functional similarity score based on ontology*

## Description

Functional similarity between two genes in ontology databases (GO or HPO). Each pair of interacting proteins in the PPI graph network, is quantified the shared annotations between them using the Fisher's combined probability test (https://doi.org/10.1007/978-1-4612-4380-9_6). This is based on the number of genes annotated in each shared ontology term and the total amount of unique genes available in the ontology database.

## Usage

```
functional_annot(annot, gene_i, gene_j)
```

## Arguments

| | |
|---|---|
| `annot` | Processed annotation data as provided by `process_annot`. |
| `gene_i` | String with the gene symbol in the row of the adjacency matrix. |
| `gene_j` | String with the gene symbol in the column of the adjacency matrix. |

## Value

Numeric value with GO/HPO functional similarity between given pair of proteins.

## See Also

- `process_annot`
- `weighted_adj`

## Examples

```
hpo <- wppi_hpo_data()
hpo <- process_annot(hpo)
hpo_score <- functional_annot(hpo, 'AKT1', 'MTOR')
# [1] 106.9376
```

## graph_from_op *Igraph object from OmniPath network*

### Description

Creation of igraph object from PPI OmniPath database with information regarding proteins and gene symbols.

### Usage

```
graph_from_op(op_data)
```

### Arguments

| | |
|---|---|
| op_data | Data frame (tibble) of OmniPath PPI interactions from wppi_omnipath_data. |

### Value

Igraph PPI graph object with vertices defined by UniProt ID and Gene Symbol, and edges based on interactions, for all connections in OmniPath.

### See Also

wppi_omnipath_data

### Examples

```
graph_op <- graph_from_op(wppi_omnipath_data())
edges_op <- igraph::E(graph_op)
vertices_op <- igraph::V(graph_op)
```

## in_omnipath *Check which genes of interest are or not in OmniPath*

### Description

Check which genes of interest are or not in OmniPath

### Usage

```
in_omnipath(graph_op, gene_set, in_network = TRUE)
```

### Arguments

| | |
|---|---|
| graph_op | Igraph object based on OmniPath PPI interactions from graph_from_op. |
| gene_set | Character vector with known-disease specific genes from which is built the functional weighted PPI. |
| in_network | Logical: whether to return the genes in the network or the missing ones. |

### Value

Character vector with genes corresponding to the query.

### See Also

- wppi_omnipath_data
- graph_from_op

### Examples

```
# genes mapped and not mapped in OmniPath
graph_op <- graph_from_op(wppi_omnipath_data())
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
genes_mapped <- in_omnipath(graph_op, genes_interest, 1)
genes_notmapped <- in_omnipath(graph_op, genes_interest, 0)
```

---

## prioritization_genes *Candidate genes prioritization*

---

### Description

Ranks candidate genes based on correlation with the given seed genes of interest. For this, the source proteins/genes (i.e. starting nodes) are reduced to the candidate genes and the target proteins/genes (i.e. end nodes) to the given genes of interest. Each candidate gene score is defined by the sum of its correlations towards the known disease-related genes.

### Usage

```
prioritization_genes(
  graph_op,
  prob_matrix,
  genes_interest,
  percentage_genes_ranked = 100
)
```

## Arguments

| | |
|---|---|
| graph_op | Igraph object based on OmniPath PPI interactions from graph_from_op. |
| prob_matrix | Matrix object with correlations/probabilities of the all nodes in the network from random_walk. |
| genes_interest | Character vector with known-disease specific genes. |
| percentage_genes_ranked | Positive integer (range between 0 and 100) specifying the percentage ( network returned in the output. If not specified, the score of all the candidate genes is delivered. |

## Value

Data frame with the ranked candidate genes based on the functional score inferred from given ontology terms, PPI and Random Walk with Restart parameters.

## See Also

- graph_from_op
- weighted_adj
- random_walk
- score_candidate_genes_from_PPI

## Examples

```
db <- wppi_data()
GO_data <- db$go
HPO_data <- db$hpo
# Genes of interest
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
# Graph object with PPI
graph_op <- graph_from_op(db$omnipath)
graph_op_1 <- subgraph_op(graph_op, genes_interest, 1)
# Filter ontology data
GO_data_filtered <- filter_annot_with_network(GO_data, graph_op_1)
HPO_data_filtered <- filter_annot_with_network(HPO_data, graph_op_1)
# Weighted adjacency
w_adj <- weighted_adj(graph_op_1, GO_data_filtered, HPO_data_filtered)
# Random Walk with Restart
w_rw <- random_walk(w_adj)
# Ranked candidate genes
scores <- prioritization_genes(graph_op_1, w_rw, genes_interest)
```

process_annot *Processing of ontology annotations*

## Description

Ontology databases such as Gene Ontology (GO, http://geneontology.org/) and Human Phenotype Ontology (HPO, https://hpo.jax.org/app/) provide important genome and disease functional annotations of genes. These combined allow to build a connection between proteins/genes and phenotype/disease. This function aggregates information in the GO and HPO ontology datasets.

## Usage

```
process_annot(data_annot)
```

## Arguments

| | |
|---|---|
| data_annot | Data frame (tibble) of GO or HPO datasets from wppi_data, wppi_go_data or wppi_hpo_data. |

## Value

A list of four elements: 1) "term_size" a list which serves as a lookup table for size (number of genes) for each ontology term; 2) "gene_term" a list to look up terms by gene symbol; 3) "annot" the original data frame (data_annot); 4) "total_genes" the number of genes annotated in the ontology dataset.

## See Also

- wppi_data
- wppi_go_data
- wppi_hpo_data

## Examples

```
hpo_raw <- wppi_hpo_data()
hpo <- process_annot(hpo_raw)
```

random_walk *Random Walk with Restart (RWR)*

## Description

RWR on the normalized weighted adjacency matrix. The RWR algorithm estimates each protein/gene relevance based on the functional similarity of genes and disease/phenotype, and the topology of the network. This similarity score between nodes measures how closely two proteins/genes are related in a network. Thus, enabling to identify which candidate genes are more related to our given genes of interest.

## Usage

```
random_walk(weighted_adj_matrix, restart_prob = 0.4, threshold = 1e-05)
```

## Arguments

| | |
|---|---|
| weighted_adj_matrix | Matrix object corresponding to the weighted adjacency from weighted_adj. |
| restart_prob | Positive value between 0 and 1 defining the restart probability parameter used in the RWR algorithm. If not specified, 0.4 is the default value. |
| threshold | Positive value depicting the threshold parameter in the RWR algorithm. When the error between probabilities is smaller than the threshold defined, the algorithm stops. If not specified, 1e-5 is the default value. |

## Value

Matrix of correlation/probabilities for the functional similarities for all proteins/genes in the network.

## See Also

- weighted_adj
- prioritization_genes
- score_candidate_genes_from_PPI

## Examples

```
db <- wppi_data()
GO_data <- db$go
HPO_data <- db$hpo
# Genes of interest
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
# Graph object with PPI
graph_op <- graph_from_op(db$omnipath)
graph_op_1 <- subgraph_op(graph_op, genes_interest, 1)
```

```
# Filter ontology data
GO_data_filtered <- filter_annot_with_network(GO_data, graph_op_1)
HPO_data_filtered <- filter_annot_with_network(HPO_data, graph_op_1)
# Weighted adjacency
w_adj <- weighted_adj(graph_op_1, GO_data_filtered, HPO_data_filtered)
# Random Walk with Restart
w_rw <- random_walk(w_adj)
```

---

`score_candidate_genes_from_PPI` *The full WPPI workflow*

---

## Description

The wppi package implements a prioritization of genes according to their potential relevance in a disease or other experimental or physiological condition. For this it uses a PPI network and functional annotations. A protein-protein interactions (PPI) in the neighborhood of the genes of interest are weighted according to the number of common neighbors of interacting partners and the similarity of their functional annotations. The PPI networks are obtained using the OmniPath (https://omnipathdb.org/) resource and functionality is deduced using the Gene Ontology (GO, http://geneontology.org/) and Human Phenotype Ontology (HPO, https://hpo.jax.org/app/) ontology databases. To score the candidate genes, a Random Walk with Restart algorithm is applied on the weighted network.

## Usage

```
score_candidate_genes_from_PPI(
  genes_interest,
  HPO_interest = NULL,
  percentage_output_genes = 100,
  graph_order = 1,
  GO_annot = TRUE,
  GO_slim = NULL,
  GO_aspects = c("C", "F", "P"),
  GO_organism = "human",
  HPO_annot = TRUE,
  restart_prob_rw = 0.4,
  threshold_rw = 1e-05,
  databases = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `genes_interest` | Character vector of gene symbols with genes known to be related to the investigated disease or condition. |

`HPO_interest` Character vector with Human Phenotype Ontology (HPO) annotations of interest from which to construct the functionality (for a list of available annotations see the 'Name' column in the data frame provided by `wppi_hpo_data`). If not specified, all the annotations available in the HPO database will be used.

`percentage_output_genes` Positive integer (range between 0 and 100) specifying the percentage (%) of the total candidate genes in the network returned in the output. If not specified, the score of all the candidate genes is delivered.

`graph_order` Integer larger than zero: the neighborhood range counted as steps from the genes of interest. These genes, also called candidate genes, together with the given genes of interest define the Protein-Protein Interaction (PPI) network used in the analysis. If not specified, the first order neighbors are used.

`GO_annot` Logical: use the Gene Ontology (GO) annotation database to weight the PPI network. The default is to use it.

`GO_slim` Character: use a GO subset (slim). If `NULL`, the full GO is used. The most often used slim is called "generic". For a list of available slims see `OmnipathR::go_annot_slim`.

`GO_aspects` Character vector with the single letter codes of the gene ontology aspects to use. By default all three aspects are used. The aspects are "C": cellular component, "F": molecular function and "P" biological process.

`GO_organism` Character: name of the organism for GO annotations.

`HPO_annot` Logical: use the Human Phenotype Ontology (HPO) annotation database to weight the PPI network. The default is to use it.

`restart_prob_rw` Numeric: between 0 and 1, defines the restart probability parameter used in the Random Walk with Restart algorithm. The default value is 0.4.

`threshold_rw` Numeric: the threshold parameter in the Random Walk with Restart algorithm. When the error between probabilities is smaller than the threshold, the algorithm stops. The default is 1e-5.

`databases` Database knowledge as produced by `wppi_data`.

`...` Passed to `OmnipathR::import_post_translational_interactions`. With these options you can customize the network retrieved from OmniPath.

## Details

If you use a GO subset (slim), building it at the first time might take around 20 minutes. The result is saved into the cache so next time loading the data from there is really quick. Gene Ontology annotations are available for a few other organisms apart from human. The currently supported organisms are "chicken", "cow", "dog", "human", "pig" and "uniprot_all". If you disable `HPO_annot` you can use `wppi` to score PPI networks other than human.

## Value

Data frame with the ranked candidate genes based on the functional score inferred from given ontology terms, PPI and Random Walk with Restart parameters.

## See Also

- wppi_data
- weighted_adj
- random_walk
- prioritization_genes

## Examples

```
# example gene set
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
# example HPO annotations set
hpo <- wppi_hpo_data()
HPO_interest <- unique(
    dplyr::filter(hpo, grepl("Diabetes", .data$Name))$Name
)
# Score 1st-order candidate genes
new_genes_diabetes <-
    score_candidate_genes_from_PPI(
        genes_interest = genes_interest,
        HPO_interest = HPO_interest,
        percentage_output_genes = 10,
        graph_order = 1)
new_genes_diabetes
# # A tibble: 30 x 3
#    score gene_symbol uniprot
#    <dbl> <chr>       <chr>
#  1 0.247 KNL1        Q8NG31
#  2 0.247 HTRA2       O43464
#  3 0.247 KAT6A       Q92794
#  4 0.247 BABAM1      Q9NWV8
#  5 0.247 SKI         P12755
# # . with 25 more rows
```

---

`subgraph_op` *Extract PPI subgraph by genes of interest*

---

## Description

From the igraph object of a PPI network obtained from OmniPath extracts a subnetwork around the provided genes of interest. The size of the graph is determined by the `sub_level` parameter, i.e. the maximum number of steps (order) from the genes of interest.

## Usage

```
subgraph_op(graph_op, gene_set, sub_level = 1L)
```

### Arguments

| | |
|---|---|
| graph_op | Igraph object based on OmniPath PPI interactions from graph_from_op. |
| gene_set | Character vector of gene symbols. These are the genes of interest, for example known disease specific genes. |
| sub_level | Integer larger than 0 defining the order of neighborhood (number of steps) from the genes of interest. If not specified, the first-order neighbors are used. |

### Value

Igraph graph object with PPI network of given genes of interest and their x-order degree neighbors.

### Examples

```
# Subgraphs of first and second order
graph_op <- graph_from_op(wppi_omnipath_data())
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
graph_op_1 <- subgraph_op(graph_op, genes_interest, 1)
graph_op_2 <- subgraph_op(graph_op, genes_interest, 2)
```

---

## weighted_adj *Weighted adjacency matrix*

---

### Description

Converts adjacency to weighted adjacency using network topology information (shared neighbors between connected nodes via common_neighbors) integrated with genome and phenotype factors from GO and HPO annotation terms (functionality computed by functional_annot). At the end, the weighted adjacency matrix is normalized by column.

### Usage

```
weighted_adj(graph_op, GO_data, HPO_data)
```

### Arguments

| | |
|---|---|
| graph_op | Igraph object based on OmniPath PPI interactions from graph_from_op. |
| GO_data | Data frame with GO annotations as provided by wppi_go_data. |
| HPO_data | Data frame with HPO annotations as provided by wppi_hpo_data. |

### Value

Weighted adjacency matrix based on network topology and functional similarity between interacting proteins/genes based on ontology databases.

## See Also

- random_walk
- prioritization_genes
- common_neighbors
- graph_from_op
- subgraph_op
- score_candidate_genes_from_PPI
- wppi_go_data
- wppi_hpo_data

## Examples

```
db <- wppi_data()
GO_data <- db$go
HPO_data <- db$hpo
# Genes of interest
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
# Graph object with PPI
graph_op <- graph_from_op(db$omnipath)
graph_op_1 <- subgraph_op(graph_op, genes_interest, 1)
# Filter ontology data
GO_data_filtered <- filter_annot_with_network(GO_data, graph_op_1)
HPO_data_filtered <- filter_annot_with_network(HPO_data, graph_op_1)
# Weighted adjacency
w_adj <- weighted_adj(graph_op_1, GO_data_filtered, HPO_data_filtered)
```

---

wppi *The wppi package*

---

## Description

The wppi package calculates context specific scores for genes in the network neighborhood of genes of interest. The context specificity is ensured by the selection of the genes of interest and potentially by using a more relevant subset of the ontology annotations, e.g. selecting only the diabetes related categories. The PPI network and the functional annotations are obtained automatically from public databases, though it's possible to use custom databases. The network is limited to a neighborhood of the genes of interest. The ontology annotations are also filtered to the genes in this subnetwork. Then the adjacency matrix is weighted according to the number of common neighbors and the similarity in functional annotations of each pair of interacting proteins. On this weighted adjacency matrix a random walk with restart is performed. The final score for the genes in the neighborhood is the sum of their scores (probabilities to be visited) in the random walk. The method can be fine tuned by setting the neighborhood range, the restart probability of the random walk and the threshold for the random walk.

## Author(s)

Ana Galhoz <ana.galhoz@helmholtz-muenchen.de> and Denes Turei <turei.denes@gmail.com>

## Examples

```
# Example with a single call:
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
scores <- score_candidate_genes_from_PPI(genes_interest)
# The workflow step by step:
db <- wppi_data()
genes_interest <-
    c("ERCC8", "AKT3", "NOL3", "GFI1B", "CDC25A", "TPX2", "SHE")
graph_op <- graph_from_op(db$omnipath)
graph_op_1 <- subgraph_op(graph_op, genes_interest, 1)
w_adj <- weighted_adj(graph_op_1, db$go, db$hpo)
w_rw <- random_walk(w_adj)
scores <- prioritization_genes(graph_op_1, w_rw, genes_interest)
```

---

# wppi_data *Database knowledge for wppi*

---

## Description

Retrieves the database knowledge necessary for WPPI directly from the databases. The databases used here are the Human Phenotype Ontology (HPO, https://hpo.jax.org/app/), Gene Ontology (GO, http://geneontology.org/) and OmniPath (https://omnipathdb.org/). The downloads carried out by the OmnipathR package and data required by wppi are extracted from each table.

## Usage

```
wppi_data(
  GO_slim = NULL,
  GO_aspects = c("C", "F", "P"),
  GO_organism = "human",
  ...
)
```

## Arguments

| | |
|---|---|
| GO_slim | Character: use a GO subset (slim). If NULL, the full GO is used. The most often used slim is called "generic". For a list of available slims see OmnipathR::go_annot_slim. |
| GO_aspects | Character vector with the single letter codes of the gene ontology aspects to use. By default all three aspects are used. The aspects are "C": cellular component, "F": molecular function and "P" biological process. |

| | |
|---|---|
| GO_organism | Character: name of the organism for GO annotations. |
| ... | Passed to OmnipathR::import_post_translational_interactions. With these options you can customize the network retrieved from OmniPath. |

## Details

If you use a GO subset (slim), building it at the first time might take around 20 minutes. The result is saved into the cache so next time loading the data from there is really quick. Gene Ontology annotations are available for a few other organisms apart from human. The currently supported organisms are "chicken", "cow", "dog", "human", "pig" and "uniprot_all". If you disable HPO_annot you can use wppi to score PPI networks other than human.

## Value

A list of data frames (tibbles) with database knowledge from HPO, GO and OmniPath.

## See Also

- wppi_go_data
- wppi_hpo_data
- wppi_omnipath_data

## Examples

```
# Download all data
data_wppi <- wppi_data()
# OmniPath
omnipath_data <- data_wppi$omnipath
# HPO
HPO_data <- data_wppi$hpo
# GO
GO_data <- data_wppi$go
```

---

# wppi_go_data *Retrieves data from Gene Ontology (GO)*

---

## Description

Gene Ontology (http://geneontology.org/), GO) annotates genes by their function, localization and biological processes.

## Usage

```
wppi_go_data(slim = NULL, aspects = c("C", "F", "P"), organism = "human")
```

### Arguments

| | |
|---|---|
| slim | Character: use a GO subset (slim). If NULL, the full GO is used. The most often used slim is called "generic". For a list of available slims see OmnipathR::go_annot_slim. |
| aspects | Character vector with the single letter codes of the gene ontology aspects to use. By default all three aspects are used. The aspects are "C": cellular component, "F": molecular function and "P" biological process. |
| organism | Character: name of the organism for GO annotations. |

### Details

If you use a GO subset (slim), building it at the first time might take around 20 minutes. The result is saved into the cache so next time loading the data from there is really quick. Gene Ontology annotations are available for a few other organisms apart from human. The currently supported organisms are "chicken", "cow", "dog", "human", "pig" and "uniprot_all". If you disable HPO_annot you can use wppi to score PPI networks other than human.

### Value

A data frame (tibble) with GO annotation data.

### See Also

wppi_data

### Examples

```
go <- wppi_go_data()
```

---

## wppi_hpo_data *Retrieves data from Human Phenotype Ontology (HPO)*

---

### Description

Human Phenotype Ontology (https://hpo.jax.org/app/), HPO) annotates proteins with phenotypes and diseases.

### Usage

```
wppi_hpo_data()
```

### Value

A data frame (tibble) with HPO data.

### See Also

wppi_data

## Examples

```
hpo <- wppi_hpo_data()
```

---

`wppi_omnipath_data` *Protein-protein interaction data from OmniPath*

---

## Description

OmniPath (https://omnipathdb.org/) integrates protein-protein interactions (PPI) from more than 30 resources. The network created is highly customizable by passing parameters to `OmnipathR::import_post_transla`

## Usage

```
wppi_omnipath_data(...)
```

## Arguments

| | |
|---|---|
| `...` | Passed to `OmnipathR::import_post_translational_interactions`. |

## Value

A data frame (tibble) with protein-protein interaction data from OmniPath.

## See Also

`wppi_data`

## Examples

```
omnipath <- wppi_omnipath_data()
```

# Index

common_neighbors, 2, *14, 15*
count_genes, 3

filter_annot_with_network, 4
functional_annot, 5, *14*

graph_from_op, *2–4*, 6, *7, 8*, *14, 15*

in_omnipath, 6

prioritization_genes, 7, *10*, *13*, *15*
process_annot, *5*, 9

random_walk, *8*, 10, *13*, *15*

score_candidate_genes_from_PPI, *8*, *10*, 11, *15*
subgraph_op, 13, *15*

weighted_adj, *5*, *8*, *10*, *13*, 14
wppi, 15
wppi_data, *9*, *12, 13*, 16, *18, 19*
wppi_go_data, *3, 4*, *9*, *14, 15*, *17*, 17
wppi_hpo_data, *3, 4*, *9*, *12*, *14, 15*, *17*, 18
wppi_omnipath_data, *6, 7*, *17*, 19