Package 'dexdash'

July 30, 2025

Type Package

Title Shiny app to explore results from differential expression experiments

Version 0.2.16

Description This R package, designed for interactive exploration and analysis of differential expression or abundance data, integrates Shiny-based visualization tools with rapid functional enrichment analysis. Users can dynamically select genes from volcano or MA plots and instantly access enriched GO, KEGG, and Reactome pathway analyses, facilitating a deeper understanding of gene functions and biological pathways. The package supports customized plotting, feature information retrieval, and handles large datasets efficiently, making it an ideal tool for genomic researchers and bioinformaticians seeking to uncover the biological significance behind differential expression patterns.

License MIT + file LICENSE

Encoding UTF-8

URL https://github.com/bartongroup/dexdash

BugReports https://github.com/bartongroup/dexdash/issues

biocViews FunctionalPrediction, DifferentialExpression, GeneSetEnrichment, GO, KEGG, Reactome, Visualization

Imports methods, stats, rlang, assertthat, tibble, tidyselect, dplyr, stringr, readr, purrr, ggplot2, ggbeeswarm, DT, jsonlite, fenr, biomaRt, shiny, bslib, bsicons, shinyWidgets, htmltools

Suggests dexdata, markdown, tidyr, testthat, quarto, forcats

Depends R (>= 4.1)

RoxygenNote 7.3.2

VignetteBuilder quarto

LazyData true

LazyDataCompression xz

Author Marek Gierlinski [aut, cre] (ORCID: <https://orcid.org/0000-0001-9149-3514>)

Maintainer Marek Gierlinski <M.Gierlinski@dundee.ac.uk>

RemoteType github

RemoteHost api.github.com

RemoteUsername bartongroup

RemoteRepo dexdash

RemoteRef main

RemoteSha c9c13efeac27df32ebe0e8ed4794f8b20335c713

GithubHost api.github.com

GithubRepo dexdash

GithubUsername bartongroup

GithubRef main

GithubSHA1 c9c13efeac27df32ebe0e8ed4794f8b20335c713

Remotes bartongroup/dexdash@v0.2.16

Contents

`dexdash_list` *Create a list of *dexdash* data sets.*

Description

Merge multiple `dexdash_set` objects into a `dexdah_list`. This function is used to merge multiple data sets into one object, that can be fed into the Shiny app. This allows for browsing results from, e.g., different experiments in one app.

Usage

```
dexdash_list(...)
```

Arguments

...	One or more objects of class ‘dexdash_set‘.

Value

A named list of ‘dexdash_set‘ objects with class attribute ‘dexdash_list‘.

dexdash_set *Create *dexdash* data set*

Description

This function creates a data set for *dexdash* Shiny app, based on user-provided data.

Usage

```
dexdash_set(de, data, metadata, name)
```

Arguments

de	Differential expression data as a data frame, expected to contain the following columns: ‘id‘ - feature id, ‘log_fc‘ - log-fold change, ‘expr‘ - expression or abundance, e.g. gene read count, ‘p_value‘ - uncorrected p-value from the DE test, ‘contrast‘ - name of the contrast used for the test.
data	A data frame containing the primary dataset for exploration, expected to contain the following columns: ‘id‘ - feature id, the same as in ‘de‘ data frame, ‘sample‘ - sample name, the same as in the ‘metadata‘ data frame, ‘value‘ - the expression or abundance.
metadata	Metadata associated with the ‘data‘ parameter, providing additional context or grouping for the samples or features included in ‘data‘. Expected to contain the following columns: ‘sample‘ - must match samples in the ‘data‘ object, ‘group‘ - a grouping variable, e.g., condition or treatment, ‘replicate‘ - replicate name.
name	A string to identify the name of the set.

Value

An object of class `dexdash_set` required by the Shiny app launcher `run_app`.

Examples

```
## Not run:
library(dexdata)
data(yeast_de, yeast_data, yeast_metadata)
dexset <- dexdash_set(yeast_de, yeast_data, yeast_metadata, "Yeast")

## End(Not run)
```

download_feature_information *Download feature (gene or protein) information for a given species*

Description

Connects to the Ensembl BioMart for the specified species and downloads feature information including feature IDs, gene symbols, and descriptions. It cleans up the gene descriptions and handles missing gene names and duplicate entries.

Usage

```
download_feature_information(
  species,
  species_file = NULL,
  id = "ensembl_gene_id"
)
```

Arguments

species	A character string specifying the species for which to download data. Available species can be listed using function 'list_species()'.
species_file	(Optional) A character string providing the path to a JSON file containing species information. If 'NULL', the default species.json file from the package will be used. See Details for more information.
id	A string with the BioMart attribute used as identifier. Default value is "ensembl_gene_id", referring to Ensembl gene ID. For UniProt identifiers, use "uniprotswissprot"; for NCBI identifiers use "entrezgene_id".

Details

BioMart requires a biomart name, a host name and a dataset, to connect to the right database. For example, for human, these arguments are "ensembl", "https://www.ensembl.org", and "hsapiens_gene_ensembl", respectively. This package contains a small JSON file (can be found at `system.file("extdata", "species.json", package = "dexdash")`), with Ensembl information for a few species. If your species is not included, you need to create a JSON file in the same format, as the included file.

Value

A tibble containing gene IDs, gene names (cleaned), and gene descriptions (cleaned).

Examples

```
## Not run:
gene_info <- download_feature_information("mouse")

## End(Not run)
```

---

download_functional_terms

Download Functional Annotation Terms

Description

Retrieves Gene Ontology (GO), Reactome, and KEGG terms for a given species.

Usage

```
download_functional_terms(species, species_file = NULL)
```

Arguments

species	A character string specifying the species for which to download data. Available species can be listed using function ‘list_species()‘.
species_file	(Optional) A character string providing the path to a JSON file containing species information. If ‘NULL‘, the default species.json file from the package will be used. See Details for more information.

Details

The GO, KEGG and Reactome databases use different species designation names. For example, designation for yeast is "sgd", "Saccharomyces cerevisiae" and "sce", for GO, Reactome and KEGG, respectively. In oder to interrogate these databases, the correct designations must be passed on. This package contains a small JSON file (can be found at `system.file("extdata", "species.json", package = "dexdash")`), with designation information for a few species. If your species is not included, you need to create a JSON file in the same format, as the included file. The species designations can be found using ‘fenr::fetch_go_species()‘, ‘fenr::fetch_reactome_species()‘ and ‘fenr::fetch_kegg_species()‘. These three functions return data frames, where column ‘designation‘ contains the species designation required.

Value

A list of three elements named "go", "reactome" and "kegg", each containing two data frames with term descriptions and feature mapping.

Examples

```
## Not run:
fterms <- download_functional_terms(species = "yeast", feature_name = "gene_id")

## End(Not run)
```

---

list_species *List Available Species*

Description

Retrieves a list of species names used in the package. These names are specified available in the JSON configuration file.

Usage

```
list_species(species_file = NULL)
```

Arguments

species_file	Optional path to the species JSON file. If 'NULL', the default file, available at `system.file("extdata", "species.json",package = "dexdash")` will be used.

Value

A character vector of species names.

prepare_functional_terms *Prepare functional terms for fast enrichment*

Description

Prepare functional terms for fast enrichment

Usage

```
prepare_functional_terms(
  terms,
  feature_name = c("gene_symbol", "gene_id"),
  all_features = NULL
)
```

Arguments

terms	A list of three elements, named "go", "reactome" and "kegg". Each element is a list with two data frames, "terms" and "mapping". See `vignette("fenr")` for more details. In a normal workflow, this object is created with `download_functional_terms`.
feature_name	The name of the column in the `mapping` tibble to be used as the feature identifier. It can be "gene_symbol" or "gene_id". If your data contain gene symbols (e.g. "BRCA1" or "FOXP1"), use `feature_name = "gene_symbol"`. If your data contain other identifiers (e.g. "ENSG00000012048" or "ENSG00000114861"), use `feature_name = "gene_id"`.
all_features	(Optional) A vector of all possible features (such as gene symbols) to prepare the data for enrichment analysis.

Value

An object to be used by the Shiny app.

Examples

```
## Not run:
terms <- download_functional_terms(species = "yeast")
fterms <- prepare_functional_terms(terms)

## End(Not run)
```

run_app *Launches an interactive differential expression (DE) data explorer*

Description

This function creates and launches a Shiny application designed for exploring differential expression (DE) data. It integrates various data inputs and initializes interactive visualization modules for an enhanced data exploration experience. The application offers a sidebar layout with themed UI components and a range of interactive modules including global input, volume-magnitude plot, feature plot, feature information, and enrichment analysis.

Usage

```
run_app(
  dexset,
  features,
  fterms,
  title = "DE explorer",
  x_variable = "sample",
  value_variable = "value",
  colour_variable = NULL
)
```

Arguments

dexset	Either a dexdash_set object containing data from one set, or a dexdash_list object containing data from multiple sets. The former one is created using dexdash_set() function, the latter one is created with dexdash_list() function.
features	A data frame that maps feature identifiers to names and descriptions. Expected to contain the following columns: ‘id‘ - feature id, must match the identifier in the ‘data‘ object, ‘name‘ - human-friendly name of the feature, e.g. gene symbol, ‘description‘ - a brief description of the feature. This data frame can be obtained using function download_gene_informarion().
fterms	An object containing functional term information, created using function download_functional_ter
title	A string with a short title, which is presented at the top of the side bar.
x_variable	A string with the name of the startup x-axis variable for the feature plot. It should correspond to a column name in dexset$metadata. The default is "sample".
value_variable	A string with the name of the startup value variable for the feature plot, used on the y-axis of a single feature plot and for fill colour in the mutliple features heatmap. It should correspond to a column name in dexset$data. The default is "value".
colour_variable	A string with the name of the startup colour variable for the feature plot. It should correspond to a column name in dexset$metadata. If left NULL (default), the second column from metadata will be used.

Value

The function does not return a value but launches a Shiny application in the user's default web browser, allowing for interactive exploration of the differential expression data.

Examples

```
if(interactive()) {
  library(dexdata)
  data(de, data, metadata, features)
  dexset <- dexdash_set(de, data, metadata, "Yeast")
  fterms <- download_functional_terms("yeast", feature_name = "gene_id")
  run_app(dexset, features, fterms)
}
```