Package: HoloFoodR (via r-universe)

July 28, 2024

Type Package

Version 0.99.8

Title R interface to EBI HoloFood resource

Description Utility package to facilitate integration and analysis of EBI HoloFood data in R. This package streamlines access to the resource, allowing for direct loading of data into formats optimized for downstream analytics.

biocViews Software, Infrastructure, DataImport

License Artistic-2.0 | file LICENSE

Encoding UTF-8

Depends R (>= 4.3.0), TreeSummarizedExperiment, MultiAssayExperiment

Imports dplyr, httr2, jsonlite, S4Vectors, stats, utils

Suggests testthat, knitr, rmarkdown, BiocStyle

URL https://github.com/EBI-Metagenomics/HoloFoodR

BugReports https://github.com/EBI-Metagenomics/HoloFoodR/issues

VignetteBuilder knitr

RoxygenNote 7.3.1

Repository https://bioc.r-universe.dev

RemoteUrl https://github.com/bioc/HoloFoodR

RemoteRef HEAD

RemoteSha 815071128cdd8e099fbee032dd56fc12a14bfd00

Contents

`doQuery` *Search HoloFood database for animals, genome catalogues, samples, or viral catalogues*

Description

Search HoloFood database for animals, genome catalogues, samples, or viral catalogues

Usage

```
doQuery(type, flatten = TRUE, ...)
```

Arguments

`type` `Character scalar` specifying the type of data to query. Must be one of the following options: `"animals"`, `"genome-catalogues"`, `"samples"` or `"viral-catalogues"`.

`flatten` `Logical scalar` specifying whether to flatten the resulting `data.frame`. This means that columns with multiple values are separated to multiple columns. (Default: `TRUE`)

`...` optional arguments:

- **max.hits** `NULL` or `integer scalar` specifying the maximum number of results to fetch. When NULL, all results are fetched. (Default: `NULL`)
- **spread.sample.types** `Logical scalar` specifying whether to create spread sample types column of animals data. In animals data, sample types column might have multiple values that might be hard to explore. This argument specifies whether to create presence/absence table from sample types. (Default: `TRUE`)
- **use.cache** `Logical scalar` specifying whether to use cache. (Default: `FALSE`)
- **cache.dir** `Character scalar` specifying cache directory. (Default: `tempdir()`)
- **clear.cache** `Logical scalar` specifying whether to remove and clear cache (Default: `FALSE`)

Details

`doQuery` is a flexible query function which can be utilized to search available animals, genome catalogues, samples, or viral catalogues. Search results can be filtered; for example, animals can be filtered based on available samples. See [Api browser](https://www.holofooddata.org/api/docs) for information on filters. You can find help on customizing queries from [here](https://emg-docs.readthedocs.io/en/latest/api.html#customising-queries).

Value

`data.frame`

Examples

```
# Find animals results. The maximum amount of results is 100. Use filter
# so that only chicken is searched.
res <- doQuery("animals", max.hits = 100, system = "chicken")
head(res)
```

`getData` *Get data from HoloFood database*

Description

Get data from HoloFood database

Usage

```
getData(
  type = NULL,
  accession.type = NULL,
  accession = NULL,
  flatten = FALSE,
  ...
)
```

Arguments

`type`	`NULL` or `character scalar` specifying the type of data to query. Must be one of the following options: `"analysis-summaries"`, `"animals"`, `"genome-catalogues"`, `"samples"`, `"sample_metadata_markers"` or `"viral-catalogues"`. When genome or viral catalogues is fetched by their accession ID, the type can also be `"genomes"` or `"fragments"`. (Default: `NULL`)
`accession.type`	`NULL` or `character scalar` specifying the type of accession IDs. Must be one of the following options: `"animals"`, `"genome-catalogues"`, `"samples"` or `"viral-catalogues"`. (Default: `NULL`)
`accession`	`NULL` or `character vector` specifying the accession IDs of type `accession.type`. (Default: `NULL`)
`flatten`	`Logical scalar` specifying whether to flatten the resulting `data.frame`. This means that columns with multiple values are separated to multiple columns. (Default: `FALSE`)
`...`	optional arguments: • **max.hits** `NULL` or `integer scalar` specifying the maximum number of results to fetch. When NULL, all results are fetched. (Default: `NULL`) • **use.cache** `Logical scalar` specifying whether to use cache (Default: `FALSE`) • **cache.dir** `Character scalar` specifying cache directory. (Default: `tempdir()`) • **clear.cache** `Logical scalar` specifying whether to remove and clear cache (Default: `FALSE`)

Details

With `getData`, you can fetch data from the database. Compared to `getResult`, this function is more flexible since it can fetch any kind of data from the database. However, this function returns the data without further wrangling as `list` or `data.frame` which are not optimized format for fetching data on samples.

Search results can be filtered; for example, animals can be filtered based on available samples. See [Api browser](https://www.holofooddata.org/api/docs) for information on filters. You can find help on customizing queries from [here](https://emg-docs.readthedocs.io/en/latest/api.html#customising-queries).

Value

`list` or `data.frame`

See Also

`getResult`

Examples

```
# Find genome catalogues
catalogues <- getData(type = "genome-catalogues")
head(catalogues)

# Find genomes based on certain genome catalogue iD
res <- getData(
    type = "genomes", accession.type = "genome-catalogues",
    accession = catalogues[1, "id"], max.hits = 100)
head(res)
```

`getMetaboLights` *Get metabolomic data from MetaboLights database*

Description

Get metabolomic data from MetaboLights database

Usage

```
getMetaboLights(url, ...)
```

Arguments

`url`	`character vector` specifying the URL address of study in MetaboLights database.
`...`	optional arguments: • **cache.dir** `Character scalar` specifying directory where downloaded file is stored. (Default: `tempdir()`)

Details

The HoloFood database primarily comprises targeted metabolomic data, omitting non-targeted metabolomic information. Nonetheless, it features URLs linking to studies within the MetaboLights database. This functionality enables users to access non-targeted metabolomic data. The function returns a structured list encompassing data frames for study metadata, assay metadata, and assay.

Value

`list`

See Also

`getResult` `getData`

Examples

```
# This example is not run, because the server fails to respond sometimes.

url <- "https://www.ebi.ac.uk/metabolights/ws/studies/MTBLS4381"

if( FALSE ){
    res <- getMetaboLights(url)
    names(res)
    head(res[["feat_meta"]])
}
```

---

getResult — *Get data on samples from HoloFood database*

Description

Get data on samples from HoloFood database

Usage

```
getResult(accession, get.metabolomic = FALSE, ...)
```

Arguments

`accession`	`Character vector` specifying the accession IDs of type samples.
`get.metabolomic`	`Logical scalar` specifying whether to retrieve metabolomic data from MetaboLights database. (Default: `FALSE`)
`...`	optional arguments:

- **use.cache** `Logical scalar` specifying whether to use cache. Note that when `get.metabolomic = TRUE` is specified, the file from the MetaboLights is stored in the local system to the location specified by `cache.dir` despite of the value of `use.cache`. (Default: `FALSE`)
- **cache.dir** `Character scalar` specifying cache directory. (Default: `tempdir()`)
- **clear.cache** `Logical scalar` specifying whether to use.cache (Default: `FALSE`)
- **assay.type** `Character scalar` specifying the name of assay in resulting `TreeSummarizedExperiment` object. (Default: `"counts"`)

Details

With `getResult`, you can fetch data on samples from the HoloFood database. Compared to `getData`, this function is more convenient for fetching the samples data because it converts the data to `MultiAssayExperiment` where different omics are stored as `TreeSummarizedExperiment` objects which are optimized for downstream analytics. Columns of returned `MultiAssayExperiment` are individual animals. These columns are linked with individual samples that are stored in `TreeSummarizedExperiment` objects.

The HoloFood database lacks non-targeted metabolomic data but fetched from MetaboLights resource. The function `getResult` facilitates the automatic retrieval of metabolomic data and its integration with other datasets from HoloFood.

Furthermore, while the HoloFoodR database does not include metagenomic assembly data, users can access such data from the MGnify database. The MGnifyR package provides a convenient interface for accessing this database. By employing `MGnifyR::getResult()`, users can obtain data formatted as a `MultiAssayExperiment` object, containing multiple `TreeSummarizedExperiment` objects. Consequently, data from both HoloFood and MGnify databases are inherently compatible for subsequent downstream analysis.

Value

`MultiAssayExperiment`

See Also

`getData` `TreeSummarizedExperiment` `MultiAssayExperiment` `MGnifyR:getResult`

Examples

```
# Find samples on certain animal
samples <- doQuery("samples", animal_accession = "SAMEA112904746")

# Get the data
mae <- getResult(samples[["accession"]])
mae
```

HoloFoodR HoloFoodR *package*

Description

HoloFoodR implements an interface to the EBI HoloFood database. See the vignette for a general introduction to this package, [about HoloFood](https://www.holofood.eu/) for general HoloFood information, and [API documentation](https://docs.holofooddata.org/api.html) for details on the JSONAPI implementation.

Author(s)

Maintainer: Tuomas Borman <tuomas.v.borman@utu.fi> (ORCID)

Authors:

- Leo Lahti <leo.lahti@iki.fi> (ORCID)

See Also

TreeSummarizedExperiment MultiAssayExperiment

Index