Package ‘ontoFAST’

April 29, 2021

Type Package

Title Interactive Annotation of Characters with Biological Ontologies

Version 1.0.0

Author Sergei Tarasov

Maintainer Sergei Tarasov <sergei.tarasov@helsinki.fi>

Description Tools for annotating characters (character matrices) with anatomical and phenotype ontologies. Includes functions for visualising character annotations and creating simple queries using ontological relationships.

URL https://github.com/sergeitarasov/ontoFAST

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Imports shinydashboard, pbapply, stringr, utils, stats, dplyr, plyr, sunburstR, usethis

Depends R (>= 3.5), shiny, ontologyIndex, visNetwork

Suggests knitr, rmarkdown

NeedsCompilation no

Repository CRAN

Date/Publication 2021-04-29 07:10:02 UTC

R topics documented:

annot_all_chars .................................................. 2
annot_char_grep .................................................. 3
chars_per_term ................................................... 4
edges2list .......................................................... 4
exclude_terms ..................................................... 5
export_annotations .............................................. 5
export_cytoscape ................................................ 6
Annotate all character statements with ontology terms

**Description**

Matches character statement and returns most similar ontology terms using grep and distance-based matching

**Usage**

```r
annot_all_chars(ontology, use.synonyms = TRUE, min_set = TRUE)
```

**Arguments**

- **ontology**: ontology_index object with character names (ontology$name_characters) and ids (ontology$id_characters)
- **use.synonyms**: using synonyms list during search. It has to be included in ontology, see syn_extract()
- **min_set**: if TRUE eliminates higher order inferred ontology terms

**Value**

The list of matched ontology terms and their character ids.
Examples

```r
# getting ontology
data(HAO)
data(Sharkey_2011)
onontology<-HAO

# parsing synonyms
ontology$parsed_synonyms<-syn_extract(HAO)

# reading in characters
char_et_states<-Sharkey_2011

# embedding characters and character ids into ontology
id_characters<-paste("CHAR:","c(1:392), sep="")
name_characters<-char_et_states[,1]
names(name_characters)<-id_characters
ontology$name_characters<-name_characters
ontology$id_characters<-id_characters

# running annotations
auto_annotations<-annot_all_chars(ontology)
```

annot_char_grep

Annotate a character statement with ontology terms

Description

Matches character statement and returns most similar ontology terms

Usage

```r
annot_char_grep(ontology, char.statement, use.synonyms = TRUE, min_set = TRUE)
```

Arguments

- `ontology` : ontology.
- `char.statement` : character statement.
- `use.synonyms` : if TRUE then the synonyms are used during search. The synonyms have to be included in the ontology using syn_extract() function.
- `min_set` : if TRUE eliminates higher order inferred ontology terms

Value

The vector of matches ontology terms.

Examples

```r
data(HAO)
annot_char_grep(HAO, "Mola on right mandible")
```
# chars_per_term

*Get number of characters per each ontology term*

## Description

Returns matrix summarizing number of characters per each ontology terms in descending order.

## Usage

```
chars_per_term(ontology, annotations = "auto")
```

## Arguments

- `ontology`: ontology_index object with character annotations included (ontology$annot_characters).
- `annotations`: which annotations to use: "auto" means automatic annotations, "manual" means manual ones. Alternatively, any other list element containing annotations can be specified.

## Value

The matrix of ontology terms IDs, their names and character number.

## Examples

```
data(HAO)
ontology<-HAO
ontology$terms_selected_id<-list("CHAR:1"=c("HAO:0000653"), "CHAR:2"=c("HAO:0000653"))
chars_per_term(ontology, annotations="manual")
```

---

# edges2list

*Convert edge matrix to list*

## Description

Takes two-column edge matrix (columns from and two) and produces a list.

## Usage

```
edges2list(edge.matrix)
```

## Arguments

- `edge.matrix`: Two-column edge matrix.

## Value

The list.
**exclude_terms**

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>annot_list&lt;-list('CHAR:1'=c(&quot;HAO:0000933&quot;, &quot;HAO:0000958&quot;), 'CHAR:2'=c(&quot;HAO:0000833&quot;, &quot;HAO:0000258&quot;))</code></td>
</tr>
<tr>
<td><code>edge.matrix&lt;-list2edges(annot_list)</code></td>
</tr>
<tr>
<td><code>edges2list(edge.matrix)</code></td>
</tr>
</tbody>
</table>

**Description**

Ontology terms to exclude for sunburst plot

**Usage**

`exclude_terms`

**Format**

List containing ontological terms.

**Examples**

`exclude_terms`

**export_annotations**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>This function converts character annotations stored in shiny_in object to table format.</td>
</tr>
</tbody>
</table>

**Usage**

`export_annotations(ontology, annotations = "auto", incl.names = FALSE, sep.head = ",", sep.tail = NULL, collapse = NULL)`
export_cytoscape

Export to Cytoscape format

Description

This function converts character annotations to Cytoscape format. It returns a table that can be saved as in csv format and imported in Cytoscape. In Cytoscape choose File -> Import -> Network -> File. Then assign columns to nodes and edges. Do not select columns that enumerate the tables’ rows!

Usage

```
export_cytoscape(
    ontology,
    annotations = "auto",
    is_a = c("is_a"),
    part_of = c("BFO:0000050")
)
```
**get_ancestors_chars**

**Arguments**

- **ontology**
  - Description: Ontology
  - Value: which annotations to use: "auto" means automatic annotations, "manual" means manual ones. Alternatively, any other list containing annotations can be specified.

- **char_id**
  - Description: IDs of character.

- **annotations**
  - Description: which annotations to use: "auto" means automatic annotations, "manual" means manual ones. Alternatively, any other list containing annotations can be specified.

**Value**

- Returns a table

**Examples**

```r
data(HAO)
data(Sharkey_2011)
data(Sharkey_2011_annot)
ontology<-HAO
# processing ontology to incorporate character statements
ontology<-onto_process(ontology, Sharkey_2011[,1], do.annot = F)
# embedding manual annotations
ontology$annot_characters<-Sharkey_2011_annot
# exporting
cyto<-export_cytoscape(ontology, annotations = ontology$annot_characters, is_a = c("is_a"), part_of = c("BFO:0000050"))
#write.csv(cyto, file="cyto.csv")
```

---

**get_ancestors_chars**  
*Get ancestral ontology terms for a set of characters*

**Description**

- Returns all ontology terms which are ancestors of a given character set

**Usage**

```r
get_ancestors_chars(ontology, char_id, annotations = "auto")
```

**Arguments**

- **ontology**
  - Description: ontology_index object with character annotations included (ontology$annot_characters).

- **char_id**
  - Description: IDs of character.

- **annotations**
  - Description: which annotations to use: "auto" means automatic annotations, "manual" means manual ones. Alternatively, any other list element containing annotations can be specified.
**get_descendants_chars**

Get characters that descendants of selected ontology term

---

**Description**

Returns all characters located (associated) with given ontology term(s)

**Usage**

```
get_descendants_chars(ontology, annotations = "auto", terms, ...)
```

**Arguments**

- **ontology** ontology_index object.
- **annotations** which annotations to use: "auto" means automatic annotations, "manual" means manual ones. Alternatively, any other list element containing annotations can be specified.
- **terms** IDs of ontology terms for which descendants are queried.
- **...** other parameters for ontologyIndex::get_descendants() function

**Value**

The vector of character IDs.

**Examples**

```
data(HAO)
ontology<-'HAO'
ontology$terms_selected_id<-'list(`CHAR:1`='HAO:0000653`)', `CHAR:2`='HAO:0000653`)
get_descendants_chars(ontology, annotations="manual", "HAO:0000653")
```
get_onto_id

Get IDs for ontology names

Description

Returns IDs of ontology terms given terms’ names

Usage

get_onto_id(vec_name, ontology, names = FALSE)

Arguments

vec_name    names of terms
ontology    ontology
names        use element name

Value

vector of IDs.

Examples

data(HAO)
vec_name=c("ventral mesofurco-profurcal muscle", "anatomical entity")
get_onto_id(vec_name, HAO)

get_onto_name

Get names for ontology IDs

Description

Returns names of ontology terms for ontology IDs

Usage

get_onto_name(vec, onto, names = FALSE)

Arguments

vec    ID or a vector of IDs
onto   ontology
names  use element name
Value

vector of names.

Examples

data(HAO)
get_onto_name("HAO:0002272", HAO)

Description

Anatomy ontology of Hymenoptera. This ontology was imported into R using get_OBO() function from ontologyIndex package (see the examples).

Usage

HAO

Format

List containing various ontological relationships and terms.

References


Hymenoptera Anatomy Ontology Portal

Examples

data(HAO)
# you can also parse the original .obo file
get_OBO(system.file("data_onto", "HAO.obo", package = "ontoFAST"), extract_tags="everything", propagate_relationships = c("BFO:0000050", "is_a"))
list2edges  

Convert list to edge matrix

Description
Takes a list of character annotations and creates an edge matrix comprising two columns: from and to. The list to table conversion can be done using ldply function from plyr package: plyr::ldply(list, rbind).

Usage
list2edges(annotated.char.list, col_order_inverse = FALSE)

Arguments
annotated.char.list
Character list with ontology annotations.
col_order_inverse
The default creates the first columns consisting if character IDs and the second columns consisting of ontology annotations. The inverse order changes the columns order.

Value
Two-column matrix.

Examples
annot_list<-list("CHAR:1"=c("HAO:0000933", "HAO:0000958"), "CHAR:2"=c("HAO:0000833", "HAO:0000258"))
list2edges(annot_list)
# use plyr package and run
plyr::ldply(annot_list, rbind)

make_shiny_in  

Make an ontology object for visualization

Description
Make an ontology object for visualization in a separate environment "ontofast"

Usage
make_shiny_in(ontology)

Arguments
ontology
Ontology
onto_process

Value
Ontology index object named as shiny_in.

Examples

```r
data(Sharkey_2011)
data(HAO)
hao_obo<-onto_process(HAO, Sharkey_2011[,1], do.annot = FALSE)
ontofast <- new.env(parent = emptyenv())
ontofast$shiny_in <- make_shiny_in(hao_obo)
runOntoFast(is_a = c("is_a"), part_of = c("BFO:0000050"), shiny_in="shiny_in" )
```

onto_process

Shortcut to process characters and ontology

Description
This is a shortcut function to make characters and ontology suitable for visualization using onto-
FAST interactive tools.

Usage

```r
onto_process(ontology, name_characters, do.annot = TRUE, ...)```

Arguments

- `ontology`: Ontology
- `name_characters`: a vector of character names
- `do.annot`: specifies if you need to run automatic annotations or not
- `...`: other arguments for annot_all_chars() function

Value
Ontology index object named

Examples

```r
data(Sharkey_2011)
data(HAO)
hao_obo<-onto_process(HAO, Sharkey_2011[,1], do.annot = FALSE)
ontofast <- new.env(parent = emptyenv())
ontofast$shiny_in <- make_shiny_in(hao_obo)
runOntoFast(is_a = c("is_a"), part_of = c("BFO:0000050"), shiny_in="shiny_in" )
```
paths_sunburst

Return ontology paths for characters

Description

Returns ontology paths for all characters. These paths can be used to create a sunburst plot of ontological dependencies.

Usage

paths_sunburst(
  ontology,
  annotations = "auto",
  exclude.terms = NULL,
  include.terms = NULL,
  use.chars = TRUE,
  sep = "-"
)

Arguments

- **ontology**: ontology_index object with character annotations included.
- **annotations**: which annotations to use: "auto" means automatic annotations, "manual" means manual ones. Alternatively, any other list element containing annotations can be specified.
- **exclude.terms**: list of terms to exclude
- **include.terms**: list of terms to include
- **use.chars**: indicate whether character ids should be included in output
- **sep**: separator used to delimit ontology terms

Value

Table.

Examples

```r
ontology_partof=get_OBO(system.file("data_onto", "HAO.obo", package = "ontoFAST"),
  extract_tags="everything", propagate_relationships = c("BFO:0000050"))
ontology_partof<-onto_process(ontology_partof, Sharkey_2011[,1], do.annot = F)
ontology_partof$annot_characters<-Sharkey_2011_annot
tb<-paths_sunburst(ontology_partof,
  annotations = ontology_partof$annot_characters, exclude.terms=exclude_terms)
# library(sunburst)
sunburst(tb)
```
runOntoFast  

**Run ontoFAST interactively**

**Description**

This function runs ontoFAST in interactive mode. Interactive mode allows character annotation and navigation through ontology network.

**Usage**

```r
runOntoFast(
  is_a = c("is_a"),
  part_of = c("BFO:0000050"),
  nchar = "all",
  show.chars = TRUE,
  shiny_in = "shiny_in",
  file2save = "OntoFAST_shiny_in.RData",
  ...
)
```

**Arguments**

- **is_a** term for is_a relationships
- **part_of** term for part_of relationships
- **nchar** number of characters to show
- **show.chars** shows character statements
- **shiny_in** a name of ontology (shiny_in) object in "ontofast" enviroment
- **file2save** a name of the file to which shiny_in object is saved in interactive mode
- **...** other arguments

**Value**

runs ontoFAST interactively using Shiny.

**Examples**

```r
data(Sharkey_2011)
data(HAO)
hao_obo<onto_process(HAO, Sharkey_2011[,1], do.annot = FALSE)
ontofast <- new.env(parent = emptyenv())
ontofast$shiny_in <- make_shiny_in(hao_obo)
runOntoFast(is_a = c("is_a"), part_of = c("BFO:0000050"), shiny_in="shiny_in")
```
Description

Anatomy ontology of Hymenoptera + Dung beetles. This ontology was imported into R using get_OBO() function from ontologyIndex packages (see the examples). The added terms for dung beetle anatomy have prefix "SCR".

Usage

Scarab

Format

List containing various ontological relationships and terms.

References


Hymenoptera Anatomy Ontology Portal

Examples

data(Scarab)
#you can also parse the original .obo file
get_OBO(system.file("data_onto", "HAO4scarabs.obo", package = "ontoFAST"), extract_tags="everything", propagate_relationships = c("BFO:0000050", "is_a"))

Sharkey_2011

Description

A table of characters and character states from Hymenoptera character matrix (Sharkey et al., 2011: Cladistics). The table contains 392 character statements.

Usage

Sharkey_2011
Format

A data table with 392 rows; each row is a character statement with character states:

**CHARACTER STATEMENTS:** Character statements

**STATES** states of the character

**STATES** ....

References


Examples

```r
data(Sharkey_2011)
# read .csv file directly
char_et_states <- read.csv(system.file("data_onto", "Sharkey_2011.csv", package = "ontoFAST"), header = TRUE, stringsAsFactors = FALSE, na.strings = "")
```

---

**Sharkey_2011_annot**  
*Hymenoptera characters annotated with ontology terms*

Description

List of character IDs and their ontology annotations. The characters are from Hymenoptera phylogeny (Sharkey et al., 2011: Cladistics). The ontology annotations are from Hymenoptera Anatomy Ontology (HAO).

Usage

`Sharkey_2011_annot`

Format

List containing characters and ontology terms.

References


Examples

`Sharkey_2011_annot`
**syn_extract**

*Link synonyms with ontology terms*

**Description**

Extracts and parses synonyms from ontology to make them readable and searchable.

**Usage**

```r
syn_extract(ontology, list_id = "synonym")
```

**Arguments**

- `ontology`  
  ontology_index object.
- `list_id`  
  ID of list where synonyms are stored.

**Value**

vector of ontology IDs and synonym names.

**Examples**

```r
data(HAO)
syn_extract(HAO)
```

---

**table2list**  
*Converts a table to list*

**Description**

Takes a table where each row consists of character ID + an ontology annotation and returns a list. Each character is assigned its own ID CHAR:XXXX.

**Usage**

```r
table2list(table, id_col = c(1), descendants_cols = c(2:ncol(table)))
```

**Arguments**

- `table`  
  A character table with annotations.
- `id_col`  
  A column ID corresponding to character.
- `descendants_cols`  
  IDs of columns corresponding to character annotations.
Value

The list.

Examples

```
# converting Sharkey_2011 dataset to list of character states
table2list(Sharkey_2011)
```

---

**Tarasov_2017_annot**  
*Dung beetle characters annotated with ontology terms*

Description

List of character IDs and their ontology annotations. All 232 characters are from Scarabaeinae phylogeny (Tarasov, 2017: Zootaxa). The ontology annotations are from a modified version Hymenoptera Anatomy Ontology (HAO) that was enriched with additional terms (SCR:) to accommodate anatomy of the dung beetles.

Usage

**Tarasov_2017_annot**

Format

List containing characters and ontology terms.

References


(Read)

Examples

**Tarasov_2017_annot**
Index

* datasets
  exclude_terms, 5
  HAO, 10
  Scarab, 15
  Sharkey_2011, 15
  Sharkey_2011_annot, 16
  Tarasov_2017_annot, 18

  annot_all_chars, 2
  annot_char_grep, 3

  chars_per_term, 4

  edges2list, 4
  exclude_terms, 5
  export_annotations, 5
  export_cytoscape, 6

  get_ancestors_chars, 7
  get_descendants_chars, 8
  get_onto_id, 9
  get_onto_name, 9

  HAO, 10

  list2edges, 11

  make_shiny_in, 11

  onto_process, 12

  paths_sunburst, 13

  runOntoFast, 14

  Scarab, 15
  Sharkey_2011, 15
  Sharkey_2011_annot, 16
  syn_extract, 17

  table2list, 17
  Tarasov_2017_annot, 18