Package 'ontoProc'

April 16, 2019

Title processing of ontologies of anatomy, cell lines, and so on

Description Support harvesting of diverse bioinformatic ontologies, making particular use of the ontologyIndex package on CRAN. We provide snapshots of key ontologies for terms about cells, cell lines, chemical compounds, and anatomy, to help analyze genome-scale experiments, particularly cell x compound screens. Another purpose is to strengthen development of compelling use cases for richer interfaces to emerging ontologies.

Version 1.4.0

Author Vince Carey <stvjc@channing.harvard.edu>

Imports Biobase, S4Vectors, methods, AnnotationDbi, stats, utils, shiny

Suggests knitr, org.Hs.eg.db, org.Mm.eg.db, testthat, BiocStyle

Depends R (>= 3.5), ontologyIndex

Maintainer VJ Carey <stvjc@channing.harvard.edu>

License Artistic-2.0

LazyLoad yes

LazyData yes

biocViews Infrastructure, GO

RoxygenNote 6.1.0

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/ontoProc

git_branch RELEASE_3_8

git_last_commit 1ae33ab

git_last_commit_date 2018-10-30

Date/Publication 2019-04-15

R topics documented:

`allGOterms` *allGOterms: data.frame with ids and terms*

Description

allGOterms: data.frame with ids and terms

Usage

```
allGOterms
```

Format

data.frame instance

Source

This is a snapshot of all the terms available from GO.db (3.4.2), August 2017, using keys(GO.db, keytype="TERM").

Examples

```
data(allGOterms)
head(allGOterms)
```

c,TermSet-method *combine TermSet instances*

Description

combine TermSet instances

Usage

```
## S4 method for signature 'TermSet'
c(x, ...)
```

Arguments

x	TermSet instance
...	additional instances

Value

TermSet instance

cellTypeToGO *utilities for approximate matching of cell type terms to GO categories and annotations*

Description

utilities for approximate matching of cell type terms to GO categories and annotations

Usage

```
cellTypeToGO(celltypeString, gotab, ...)

cellTypeToGenes(celltypeString, gotab, orgDb, cols = c("ENSEMBL",
  "SYMBOL"), ...)
```

Arguments

celltypeString	character atom to be used to search GO terms using
gotab	a data.frame with columns GO (goids) and TERM (term strings) agrep
...	additional arguments to agrep
orgDb	instances of orgDb
cols	columns to be retrieved in select operation

Value

data.frame

data.frame

Note

Very primitive, uses agrep to try to find relevant terms.

Examples

```
data(allGOterms)
library(org.Hs.eg.db)
head(cellTypeToGO("serotonergic neuron", allGOterms))
head(cellTypeToGenes("serotonergic neuron", allGOterms, org.Hs.eg.db))
```

---

cleanCLOnames *obtain named character vector of terms from Cell Line Ontology, omitting obsolete and trailing 'cell'*

Description

obtain named character vector of terms from Cell Line Ontology, omitting obsolete and trailing 'cell'

Usage

```
cleanCLOnames()
```

Value

character()

Examples

```
cleanCLOnames()[1:10]
```

---

demoApp *demonstrate the use of makeSelectInput*

Description

demonstrate the use of makeSelectInput

Usage

```
demoApp()
```

Value

Run only for side effect of starting a shiny app.

Examples

```
  if (interactive()) {
require(shiny)
print(demoApp())
}
```

dropStop *dropStop is a utility for removing certain words from text data*

Description

dropStop is a utility for removing certain words from text data

Usage

```
dropStop(x, drop, lower = TRUE, splitby = " ")
```

Arguments

x	character vector of strings to be cleaned
drop	character vector of words to scrub
lower	logical, if TRUE, x converted with tolower
splitby	character, used with strsplit to tokenize x

Value

a list with one element per input string, split by " ", with elements in drop removed

Examples

```
data(minicorpus)
minicorpus[1:3]
dropStop(minicorpus)[1:3]
```

fastGrep *some fields of interest are lists, and grep per se should not be used – this function checks and uses grep within vapply when appropriate*

Description

some fields of interest are lists, and grep per se should not be used – this function checks and uses grep within vapply when appropriate

Usage

```
fastGrep(patt, onto, field, ...)
```

Arguments

patt	a regular expression whose presence in field should be checked
onto	an ontologyIndex instance
field	the ontologyIndex component to be searched
...	passed to grep

Value

logical vector indicating vector or list elements where a match is found

Examples

```
cheb = getChebiOnto()
ind = fastGrep("17-AAG", cheb, "synonym")
cheb$name[ind]
```

`getCellOnto` *load ontologies that may include non-ascii strings and therefore cannot be in data folder*

Description

load ontologies that may include non-ascii strings and therefore cannot be in data folder

Usage

```
getCellOnto()

getCellLineOnto()

getEFOOnto()

getChebiLite()

getCellosaurusOnto()

getUBERON_NE()

getChebiOnto()

getOncotreeOnto()

getDiseaseOnto()

getGeneOnto()

getHCAOnto()
```

Value

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex

instance of ontology_index (S3) from ontologyIndex,

Note

produced from HCAO.owl at https://github.com/HumanCellAtlas/ontology as of 15 Aug 2018

Examples

```
co = getCellOnto()
co
clo = getCellLineOnto()
length(clo$id)
che = getChebiLite()
length(che$id)
efo = getEFOOnto()
length(efo$id)
```

---

humrna *humrna: a data.frame of SRA metadata related to RNA-seq in humans*

Description

humrna: a data.frame of SRA metadata related to RNA-seq in humans

Usage

```
humrna
```

Format

data.frame

Note

arbitrarily chosen from RNA-seq studies for taxon 9606

Source

NCBI SRA

Examples

```
data(humrna)
names(humrna)
head(humrna[,1:5])
```

liberalMap *Produce a data.frame with a set of naive terms mapped to all matching ontology ids and their formal terms*

Description

Produce a data.frame with a set of naive terms mapped to all matching ontology ids and their formal terms

Usage

```
liberalMap(terms, onto, useAgrep = FALSE, ...)
```

Arguments

terms	character() vector, can use grep-compatible regular expressions
onto	an instance of ontologyIndex::ontology_index
useAgrep	logical(1) if TRUE, agrep will be used
...	passed to agrep if used

Value

a data.frame

Examples

```
cands = c("astrocyte$", "oligodendrocyte", "oligodendrocyte precursor",
   "neoplastic", "^neuron$", "^vascular", "badterm")
co = ontoProc::getCellOnto()
liberalMap(cands, co)
```

makeSelectInput *generate a selectInput control for an ontologyIndex slice*

Description

generate a selectInput control for an ontologyIndex slice

Usage

```
makeSelectInput(onto, term, type = "siblings", inputId, label,
  multiple = TRUE, ...)
```

Arguments

onto	ontologyIndex instance
term	character(1) term used as basis for term list option set in the control
type	character(1) 'siblings' or 'children', relationship to 'term' that the options will satisfy
inputId	character(1) for use in server
label	character(1) for labeling in ui
multiple	logical(1) passed to selectInput
...	additional parameters passed to selectInput

Value

a selectInput control

Examples

```
makeSelectInput
```

---

mapOneNaive *use grep or agrep to find a match for a naive token into ontology*

Description

use grep or agrep to find a match for a naive token into ontology

Usage

```
mapOneNaive(naive, onto, useAgrep = FALSE, ...)
```

Arguments

naive	character(1)
onto	an instance of ontologyIndex::ontology_index
useAgrep	logical(1) if TRUE, agrep will be used
...	passed to agrep if used

Value

if a match is found, the result of grep/agrep with value=TRUE is returned; otherwise a named NA_character_ is returned

named vector, names are ontology identifiers, values are matched strings

Examples

```
co = ontoProc::getCellOnto()
mapOneNaive("astrocyte", co)
```

`minicorpus` *minicorpus: a vector of annotation strings found in 'study title' of SRA metadata.*

Description

minicorpus: a vector of annotation strings found in 'study title' of SRA metadata.

Usage

```
minicorpus
```

Format

character vector

Note

arbitrarily chosen from titles of RNA-seq studies for taxon 9606

Source

NCBI SRA

Examples

```
data(minicorpus)
head(minicorpus)
```

`nomenCheckup` *repair nomenclature mismatches (to curated term set) in a vector of terms*

Description

repair nomenclature mismatches (to curated term set) in a vector of terms

Usage

```
nomenCheckup(cand, namedOffic, n = 1, tagcolname = "tag", ...)
```

Arguments

`cand`	character vector of candidate terms
`namedOffic`	named character vector of curated terms, the names are regarded as tags, intended to be identifiers in curated ontologies
`n`	numeric(1) number of nearest neighbors to return
`tagcolname`	character(1) prefix used to name columns for tags in output
`...`	passed to `adist`

Value

a data.frame instance with 2n+1 columns (column 1 is candidate, remaining n pairs of columns are (term, tag) for n nearest neighbors as measured by `adist`.

Examples

```
candidates = c("JHH7", "HUT102", "HS739T", "NCIH716")
# the candidates are cell line names returned in the text dump from
# https://portals.broadinstitute.org/ccle/page?gene=AHR
# note that one must travel to the third nearest neighbor
# to find the match (and tag) for Hs 739.T
# in this example, we compare to cell line names in Cell Line Ontology
nomenCheckup(candidates, cleanCLOnames(), n=3, tagcolname="clo")
```

---

secLevGen *simple generation of children of 'choices' given as terms, returned as TermSet*

Description

simple generation of children of 'choices' given as terms, returned as TermSet

Usage

```
secLevGen(choices, ont)
```

Arguments

`choices`	vector of terms
`ont`	instance of ontology_index (S3) from ontologyIndex package

Value

TermSet instance

Examples

```
efoOnto = getEFOOnto()
secLevGen( "disease", efoOnto )
```

`selectFromMap` *select a set of elements from a term 'map' and return a contribution to a data.frame*

Description

select a set of elements from a term 'map' and return a contribution to a data.frame

Usage

```
selectFromMap(namedvec, index)
```

Arguments

`namedvec`	named character vector, as returned from `mapOneNaive`
`index`	numeric() or integer(), typically of length one

Value

a data.frame; if `index` does not inherit from `numeric`, a data.frame of one row with columns 'ontoid' and 'term' populated with `NA_character_` is returned, otherwise a similarly named data.frame is returned with contents from the selected elements of `namedvec`

Examples

```
co = ontoProc::getCellOnto()
mast = mapOneNaive("astrocyte", co)
selectFromMap(mast, 1)
```

`siblings_TAG` *generate a TermSet with siblings of a given term, excluding that term by default*

Description

generate a TermSet with siblings of a given term, excluding that term by default

acquire the label of an ontology subject tag

acquire the labels of children of an ontology subject tag

Usage

```
siblings_TAG(Tagstring = "EFO:1001209", ontology, justSibs = TRUE)

label_TAG(Tagstring = "EFO:0000311", ontology)

children_TAG(Tagstring = "EFO:1001209", ontology)
```

Arguments

Tagstring	a character(1) that identifies a term
ontology	instance of ontology_index (S3) from ontologyIndex
justSibs	character(1)

Value

TermSet instance

character(1)

TermSet instance

Note

for label_TAG, Tagstring may be a vector

Examples

```
efoOnto = getEFOOnto()
siblings_TAG( "EFO:1001209", efoOnto )
efoOnto = getEFOOnto()
label_TAG( "EFO:0000311", efoOnto )
efoOnto = getEFOOnto()
children_TAG( ontology = efoOnto )
```

---

stopWords *stopWords: vector of stop words from xpo6.com*

Description

stopWords: vector of stop words from xpo6.com

Usage

```
stopWords
```

Format

character vector

Note

"Stop words" are english words that are assumed to contribute limited semantic value in the analysis of free text.

Source

http://xpo6.com/list-of-english-stop-words/

Examples

```
data(stopWords)
head(stopWords)
```

`TermSet-class` *manage ontological data with tags and a DataFrame instance*

Description

manage ontological data with tags and a DataFrame instance

abbreviated display for TermSet instances

Usage

```
## S4 method for signature 'TermSet'
show(object)
```

Arguments

`object`	instance of TermSet class

Value

instance of TermSet

Examples

```
efoOnto = getEFOOnto()
defsibs = siblings_TAG("EFO:1001209", efoOnto)
class(defsibs)
defsibs
```

Index