Package ‘alignontology’

February 6, 2021

Type Package

Title An implementation of alignment in Network-Extracted Ontology algorithm

Version 0.1.0

Date 2021-01-21

Depends R (>= 4.0.2)

Description This package facilitates application of ontology alignment in Network-Extracted Ontology (NeXO) algorithm in R. This algorithm was originally implemented in C++ by Michael Kramer for NeXO algorithm found by Janusz Dutkowski. The inputs are computed and reference ontologies as source-target-relation data frames.

License GPL-3

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports devtools,
tidyverse

Suggests BiocStyle,
knitr,
rmarkdown,
kableExtra,
igraph,
magick,
testthat

URL https://github.com/herdiantrisufriyana/alignontology

BugReports https://github.com/herdiantrisufriyana/alignontology/issues

VignetteBuilder knitr

R topics documented:

alignontology 2
input_example 3

Index **4**

alignontology *Ontology alignment*

Description

This function runs alignment algorithm in Network-Extracted Ontology (NexO).

Usage

```
alignontology(
  computed,
  reference,
  min_similarity = 0.05,
  semantic_verification = "criss_cross",
  allow_multimap = F,
  features_as_terms = T,
  feature_name = "feature",
  n_rand_align = 100,
  effect_size = 0,
  pvals = T
)
```

Arguments

`computed` Computed ontology, a data frame with rows for ontologies and three columns for source, target, and relation. This is ontology that will be aligned to reference. Please see details below.

`reference` Reference ontology, a data frame with rows for ontologies and three columns for source, target, and relation. This is ontology which another is aligned to. Please see details below.

`min_similarity` A numeric from 0 to 1. A score that should be surpassed for mapping between two ontologies. A value of 0.05 works well. Please see https://pubmed.ncbi.nlm.nih.gov/23242164/.

`semantic_verification` A character of a setting that will switch the way looking for inconsistencies between pairs of mappings. Please see details below.

`allow_multimap` A logical whether the same term is allowed to be mapped twice (only in the case that two possible mappings are mathematically equivalent).

`features_as_terms` A logical whether features are included as terms.

`feature_name` A character to annotate feature (source)-ontology (target) relation in both of the input ontologies.

`n_rand_align` An integer of number of random alignments to perform for calculating false discovery rate (FDR) per pair of ontologies.

`effect_size` Effect size, a numeric default to 0. Otherwise, this will decrease actual score by effect size to ensure effect is not only statistically significant but also large.

`pvals` A logical whether using p-values or FDRs instead of minimum similarity. This means lower value is better.

Details

Feature (source)-ontology (target) relation is annotated as 'feature', while ontology-ontology relation is annotated as 'is_a'. To differentiate between feature and ontology names, a prefix followed by ':' precedes an ontology name. The prefix should be different for computed and reference ontology. All columns are characters.

For semantic verification, use `criss_cross` for parent-child criss-cross mappings, i.e. cases where e1<e2 and e1'<e2' but we have the mappings (e1,e2') and (e2,e1'). Use `strict_hierarchy` for considering it an inconsistency if we have the mappings (e1,e1') and (e2,e2') where and either e1<e2 or e1'<e2' but not both. This is more strict than criss_cross. This does not check if any of e1, e2, e1', or e2' is a feature, as this lead to an overly strict requirement that two concepts containing identical features to be mapped to each other. Use none to do neither of the methods. It still looks for double mapping inconsistencies. Use 'sib_sib' to allow two sibling terms in one ontology to be mapped to parent-child terms in the second ontology.

Value

Ontology alignment, a data frame with rows for ontologies and five columns for computed ontology, reference ontology, similarity, FDR, and terminal nodes.

Examples

```
## Create input example
input=input_example()

## Ontology alignment
ontology=alignontology(input$computed,input$reference)
```

`input_example` *Make an input example for alignontology package*

Description

This function create an input example for alignontology package.

Usage

```
input_example()
```

Value

output A list of inputs: 1) computed; and 2) reference. Both are data frames with rows for ontologies and three columns for source, target, and relation. Feature (source)-ontology (target) relation is annotated as 'feature', while ontology-ontology relation is annotated as 'is_a'. To differentiate between feature and ontology names, a prefix followed by ':' precedes an ontology name. The prefix should be different for computed and reference ontology. All columns are characters.

Examples

```
## Create input example
input=input_example()
```

Index

∗ **alignontology**
    alignontology, 2
∗ **data**
    input_example, 3
∗ **example**
    input_example, 3

alignontology, 2

input_example, 3