Package ‘epos’

May 5, 2020

Type Package

Title Epilepsy Ontologies Similarities

Version 0.1.1

Author Bernd Mueller

Maintainer Bernd Mueller <bernd.mueller@zbmed.de>

Description Analysis and visualization of similarities between epilepsy ontologies based on text mining results by comparing ranked lists of co-occurring drug terms in the corpus of LIVIVO. The ranked result lists of neurological drug terms co-occurring with named entities from the epilepsy ontologies EpSO, ESSO, and EPILONT are aggregated in order to generate two different results: an overview table of drugs that are relevant to epilepsy created with the method createNeuroTable, and a plot of tanimoto similarity coefficients between the aggregated list of drug terms against the list of drug terms from each of the ontologies created with the method createTanimotoBaseline(). The alignment of the Top-K Ranked Lists is conducted using the R-package TopKLists <https://cran.r-project.org/package=TopKLists>. The source data to create the ranked lists of drug names is produced using the text mining workflows described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558> and Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>.

Depends R (>= 3.1.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

Imports hash, ggplot2, tidyr, dplyr, testthat, gridExtra, TopKLists,
stringr, xtable

Suggests knitr, rmarkdown

NeedsCompilation no

Repository CRAN

Date/Publication 2020-05-05 13:40:09 UTC

R topics documented:

`calcCosine` *Calculate the cosine similarity metric for two lists a and b*

Description

Calculate the cosine similarity metric for two lists a and b

Usage

```
calcCosine(a, b)
```

Arguments

a	list with elements that should be of same type as in list b
b	list with elements

Value

co list with length of set b containing the cosine similarity coefficient at each position

Examples

```
calcCosine(c(1,2), c(2,3))
```

---

calcDice *Calculate the dice similarity metric for two lists a and b*

Description

Calculate the dice similarity metric for two lists a and b

Usage

```
calcDice(a, b)
```

Arguments

a	list with elements that should be of same type as in list b
b	list with elements

Value

di list with length of set b containing the dice similarity coefficient at each list element

Examples

```
calcDice(c(1,2), c(2,3))
```

---

calcJaccard *Calculate the jaccard coefficient for two lists a and b*

Description

Calculate the jaccard coefficient for two lists a and b

Usage

```
calcJaccard(a, b)
```

Arguments

a	list with elements that should be of same type as in list b
b	list with elements

Value

ja list with length of set b containing the jaccard similarity coefficient for each list element

Examples

```
calcJaccard(c(1,2), c(2,3))
```

cosine *Calculate cosine similarity metric*

Description

Calculate cosine similarity metric

Usage

```
cosine(ainterb, lengtha, lengthb)
```

Arguments

`ainterb`	integer value with number of intersecting elements between set a and b
`lengtha`	integer value with the number of items in set a
`lengthb`	integer value with the number of items in set b

Value

cosine double vlaue with the cosine similarity coefficient

Examples

```
cosine(1,3,4)
```

createBaseTable *Create the final resulting data frame*

Description

Create the final resulting data frame

Usage

```
createBaseTable(neurospace, atchashda, atchashsec, dneuromaxk)
```

Arguments

`neurospace`	list of drug names that were aggregated using TopKLists::calculate.max topkspace
`atchashda`	hashmap retrieved from readAtcMapIntoHashMapDrugNamesAtcCodes
`atchashsec`	hashmap retrieved from readSecondLevelATC
`dneuromaxk`	data frame containing columns for each intersection, ATC class, and reference list

Value

data frame containing drug names with additional columns listing association to ATC classes

Examples

```
utils::data(rawDrugBankCoOcEpSO, package="epos")
utils::data(rawDrugBankCoOcESSO, package="epos")
utils::data(rawDrugBankCoOcEPILONT, package="epos")
atchashda <-
readAtcMapIntoHashMapDrugNamesAtcCodes(
  system.file("extdata", "db-atc.map", package = "epos"), "\t")
atchashaa <-
  readAtcMapIntoHashMapAtcCodesAtcNames(
    system.file("extdata", "db-atc.map", package = "epos"), "\t")
atchashsec <-
  readSecondLevelATC(
    system.file("extdata", "drugbankatc-secondlevel.map", package = "epos"), "\t")
tepso <- rawDrugBankCoOcEpSO[1:150]
tesso <- rawDrugBankCoOcESSO[1:150]
tepi <- rawDrugBankCoOcEPILONT[1:150]
lepso <- genDictListFromRawFreq(tepso)
neuroepso <- filterNeuroDrugs(lepso, atchashda)
lesso <- genDictListFromRawFreq(tesso)
neuroesso <- filterNeuroDrugs(lesso, atchashda)
lepi <- genDictListFromRawFreq(tepi)
neuroepi <- filterNeuroDrugs(lepi, atchashda)
dneuro <-
  data.frame(EpSO = neuroepso[1:15],
             ESSO = neuroesso[1:15],
             EPILONT = neuroepi[1:15])
dneuromaxk <- TopKLists::calculate.maxK(dneuro, 3, 5, 10)
neurospace <- as.character(dneuromaxk$topkspace)
neurotable <-
  createBaseTable(neurospace, atchashda, atchashsec, dneuromaxk)
```

---

createDashVectorForATC

Creates a vector with an X at each position where a drug from the druglist matches the ATC class list slatc

---

Description

Creates a vector with an X at each position where a drug from the druglist matches the ATC class list slatc

Usage

```
createDashVectorForATC(druglist, atchashda, atchashsec, slatc)
```

Arguments

`druglist`	list of drug names
`atchashda`	hash retrieved from readAtcMapIntoHashMapDrugNamesAtcCodes
`atchashsec`	hash retrieved from readSecondLevelATC
`slatc`	list of ATC classes

Value

list with crosses if the drug in druglist matches at the position of the ATC class in slatc

Examples

```
## Not run:
createDashVectorForATC(druglist, atchashda, atchashsec, slatc)

## End(Not run)
```

`createJaccardPlotDBMeSH` *Creates the plot for all jaccard coefficients amongst the three epilepsy ontologies*

Description

Creates the plot for all jaccard coefficients amongst the three epilepsy ontologies

Usage

```
createJaccardPlotDBMeSH(jmeshepso, jmeshesso, jmeshepi)
```

Arguments

`jmeshepso`	list containing jaccard coefficients between mesh and epso for increasing k
`jmeshesso`	list containing jaccard coefficients between mesh and esso for increasing k
`jmeshepi`	list containing jaccard coefficients between mesh and epi for increasing k

Value

jaccardepilepsyplot the ggplot object

Examples

```
## Not run:
jaccardepilepsyplot <- createJaccardPlotAll(jaccardepso, jaccardesso)

## End(Not run)
```

`createJaccardPlotMeSHFive` *Creates the plot for all jaccard coefficients amongst the three epilepsy ontologies*

Description

Creates the plot for all jaccard coefficients amongst the three epilepsy ontologies

Usage

```
createJaccardPlotMeSHFive(jmeshepso, jmeshesso, jmeshepi, jmeshepilepsyand,
  jmeshepilepsyor)
```

Arguments

`jmeshepso`	list of jaccard coefficients between mesh and epso for increasing k
`jmeshesso`	list of jaccard coefficients between mesh and esso for increasing k
`jmeshepi`	list of jaccard coefficients between mesh and epi for increasing k
`jmeshepilepsyand`	list of jaccard coefficients between mesh and the intersection of epso, esso, and epi for increasing k
`jmeshepilepsyor`	list of jaccard coefficients between mesh and the union of epso, esso, and epi for increasing k

Value

jaccardepilepsyplot the ggplot object

Examples

```
## Not run:
jaccardepilepsyplot <- createJaccardPlotAll(jaccardepso, jaccardesso)

## End(Not run)
```

createNeuroTable *Main function to call everything and produce the results*

Description

Main function to call everything and produce the results

Usage

```
createNeuroTable(coocepso, coocesso, coocepi)
```

Arguments

coocepso	list of drug names sorted by frequency co-occuring with EpSO
coocesso	list of drug names sorted by frequency co-occuring with ESSO
coocepi	list of drug names sorted by frequency co-occuring with EPILONT

Value

result table containin the aggregated list of drug terms and their associations

Examples

```
utils::data(rawDrugBankCoOcEpSO, package="epos")
utils::data(rawDrugBankCoOcESSO, package="epos")
utils::data(rawDrugBankCoOcEPILONT, package="epos")
createNeuroTable(coocepso = rawDrugBankCoOcEpSO[1:150],
  coocesso=rawDrugBankCoOcESSO[1:150],
  coocepi=rawDrugBankCoOcEPILONT[1:150])
```

createTanimotoBaseline *Creates the plot for all jaccard coefficients amongst the three epilepsy ontologies*

Description

Creates the plot for all jaccard coefficients amongst the three epilepsy ontologies

Usage

```
createTanimotoBaseline(neuroepso, neuroesso, neuroepi, dneuromaxk)
```

Arguments

`neuroepso`	list of neuro drug names co-occurring with epso
`neuroesso`	list of neuro drug names co-occurring with esso
`neuroepi`	list of neuro drug names co-occurring with epi
`dneuromaxk`	object returned from TopKLists::calculate.maxkS

Value

jaccardepilepsyplot the ggplot object

Examples

```
utils::data(rawDrugBankCoOcEpSO, package="epos")
atchashda <-
  readAtcMapIntoHashMapDrugNamesAtcCodes(
    system.file("extdata", "db-atc.map", package = "epos"), "\t")
tepso <- genDictListFromRawFreq(rawDrugBankCoOcEpSO[1:150])
neuroepso <- filterNeuroDrugs(tepso, atchashda)
utils::data(rawDrugBankCoOcESSO, package="epos")
tesso <- genDictListFromRawFreq(rawDrugBankCoOcESSO[1:150])
neuroesso <- filterNeuroDrugs(tesso, atchashda)
utils::data(rawDrugBankCoOcEPILONT, package="epos")
tepi <- genDictListFromRawFreq(rawDrugBankCoOcEPILONT[1:150])
neuroepi <- filterNeuroDrugs(tepi, atchashda)
dneuro <-
  data.frame(EpSO = neuroepso[1:15],
             ESSO = neuroesso[1:15],
             EPILONT = neuroepi[1:15])
dneuromaxk <- TopKLists::calculate.maxK(dneuro, 3, 5, 10)
tanimotobaseline <- createTanimotoBaseline(neuroepso, neuroesso, neuroepi, dneuromaxk)
```

`dice` *Calculate dice similarity metric*

Description

Calculate dice similarity metric

Usage

```
dice(ainterb, lengtha, lengthb)
```

Arguments

`ainterb`	integer value with number of intersecting elements between set a and b
`lengtha`	integer value with the number of items in set a
`lengthb`	integer value with the number of items in set b

Value

dice double vlaue with the dice similarity coefficient

Examples

```
dice(1, 3, 4)
```

`doFullPlot` *Does the full plot on one page*

Description

Does the full plot on one page

Usage

```
doFullPlot(cosinemeshplot, cosinedrugbankplot, cosineepilepsyplot,
  dicemeshplot, dicedrugbankplot, diceepilepsyplot, jaccardmeshplot,
  jaccarddrugbankplot, jaccardepilepsyplot)
```

Arguments

`cosinemeshplot` plot with cosine coefficients against MeSH

`cosinedrugbankplot` plot with cosine coefficients against DrugBank

`cosineepilepsyplot` plot with cosine coefficients of Epilepsy Ontologies

`dicemeshplot` plot with dice coefficients against MeSH

`dicedrugbankplot` plot with dice coefficients against DrugBank

`diceepilepsyplot` plot with dice coefficients of Epilepsy Ontologies

`jaccardmeshplot` plot with jaccard coefficients against MeSH

`jaccarddrugbankplot` plot with jaccard coefficients against DrugBank

`jaccardepilepsyplot` plot with jaccard coefficients of Epilepsy Ontologies

Value

full

Examples

```
## Not run:
full <- doFullPlot (cosinemeshplot,
                    cosinedrugbankplot,
                    cosineepilepsyplot,
                    dicemeshplot,
                    dicedrugbankplot,
                    diceepilepsyplot,
                    jaccardmeshplot,
                    jaccarddrugbankplot,
                    jaccardepilepsyplot)

## End(Not run)
```

`filterApprovedDrugs` *Filter a given list of drug names for having an ATC code, if not they are dropped*

Description

Filter a given list of drug names for having an ATC code, if not they are dropped

Usage

```
filterApprovedDrugs(druglist, atchashda)
```

Arguments

`druglist`	a list of drug names
`atchashda`	a hash containing the drug names as keys

Value

approveddrugs a hash filtered for having an ATC code

Examples

```
utils::data(rawDrugBankCoOcEpSO, package="epos")
atchashda <-
  readAtcMapIntoHashMapDrugNamesAtcCodes(
  system.file("extdata", "db-atc.map", package = "epos"), "\t")
tepso <- genDictListFromRawFreq(rawDrugBankCoOcEpSO)
filterApprovedDrugs(tepso, atchashda)
```

filterNeuroDrugs *Filter a given list of drug names for having an ATC code starting with N indicating to be a drug for the Nervous System*

Description

Filter a given list of drug names for having an ATC code starting with N indicating to be a drug for the Nervous System

Usage

```
filterNeuroDrugs(druglist, atchashda)
```

Arguments

druglist	a list of drug names
atchashda	a hash containing the drug names as keys

Value

neurodrugs a hash filtered for having an ATC code starting with N

Examples

```
utils::data(rawDrugBankCoOcEpSO, package="epos")
atchashda <-
  readAtcMapIntoHashMapDrugNamesAtcCodes(
  system.file("extdata", "db-atc.map", package = "epos"), "\t")
tepso <- genDictListFromRawFreq(rawDrugBankCoOcEpSO)
nepso <- filterNeuroDrugs(tepso, atchashda)
```

genDictListFromRawFreq *Clears object that was loaded from harddrive into a list of terms sorted by frequency*

Description

Clears object that was loaded from harddrive into a list of terms sorted by frequency

Usage

```
genDictListFromRawFreq(topfreqdictraw)
```

Arguments

topfreqdictraw	list with terms from a dictionary sorted by frequency

Value

a sorted list of terms

Examples

```
utils::data(rawDrugBankCoOcEpSO, package="epos")
genDictListFromRawFreq(rawDrugBankCoOcEpSO)
```

`jaccard` *Calculate jaccard similarity metric for two sets a and b*

Description

Calculate jaccard similarity metric for two sets a and b

Usage

```
jaccard(ainterb, aunionb, lengtha, lengthb)
```

Arguments

`ainterb`	integer value with number of intersecting elements between set a and b
`aunionb`	integer value with number of union elements between set a and b
`lengtha`	length of set a
`lengthb`	length of set b

Value

jac double value with the jaccard similarity coefficient

Examples

```
jaccard(1,3, 2, 3)
```

`rawDrugBankCoOcEPILONT` *List of frequency named by drug terms that co-occur with the EPILONT ontology in the corpus of LIVIVO.*

Description

List of frequency named by drug terms that co-occur with the EPILONT ontology in the corpus of LIVIVO.

Usage

```
rawDrugBankCoOcEPILONT
```

Format

A named list of drug term frequencies

Source

The source data to create the ranked lists of drug names is produced using the text mining workflows described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558> and Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>. The moment of execution for generating the data on the corpus of LIVIVO is June 26th 2018.

Examples

```
utils::data(rawDrugBankCoOcEPILONT, package="epos")
```

`rawDrugBankCoOcEpSO` *List of frequency named by drug terms that co-occur with the EpSO ontology in the corpus of LIVIVO.*

Description

List of frequency named by drug terms that co-occur with the EpSO ontology in the corpus of LIVIVO.

Usage

```
rawDrugBankCoOcEpSO
```

Format

A named list of drug term frequencies

Source

The source data to create the ranked lists of drug names is produced using the text mining workflows described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558> and Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>. The moment of execution for generating the data on the corpus of LIVIVO is June 26th 2018.

Examples

```
utils::data(rawDrugBankCoOcEpSO, package="epos")
```

`rawDrugBankCoOcESSO` *List of frequency named by drug terms that co-occur with the ESSO ontology in the corpus of LIVIVO.*

Description

List of frequency named by drug terms that co-occur with the ESSO ontology in the corpus of LIVIVO.

Usage

```
rawDrugBankCoOcESSO
```

Format

A named list of drug term frequencies

Source

The source data to create the ranked lists of drug names is produced using the text mining workflows described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558> and Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>. The moment of execution for generating the data on the corpus of LIVIVO is June 26th 2018.

Examples

```
utils::data(rawDrugBankCoOcESSO, package="epos")
```

readAtcMapIntoHashMapAtcCodesAtcNames

Processes the input file db-atc.map to form a HashMap containing the drug names with ATC codes

Description

Processes the input file db-atc.map to form a HashMap containing the drug names with ATC codes

Usage

```
readAtcMapIntoHashMapAtcCodesAtcNames(filename, seperator)
```

Arguments

filename	character vector with the file name of the file db-atc.map
seperator	character vector with the seperator used within the map-file

Value

atchashaa hash with atc codes as keys and atc names as values

Examples

```
atchashaa <-
  readAtcMapIntoHashMapAtcCodesAtcNames(
    system.file("extdata", "db-atc.map", package = "epos"), "\t")
```

readAtcMapIntoHashMapDrugNamesAtcCodes

Processes the input file db-atc.map to form a HashMap containing the drug names with ATC codes

Description

Processes the input file db-atc.map to form a HashMap containing the drug names with ATC codes

Usage

```
readAtcMapIntoHashMapDrugNamesAtcCodes(filename, seperator)
```

Arguments

filename	character vector with the file name of the file db-atc.map
seperator	character vector with the seperator used within the map-file

Value

atchashda a hash with drug names as keys and atc codes as values

Examples

```
atchashda <- readAtcMapIntoHashMapDrugNamesAtcCodes(
  system.file("extdata", "db-atc.map", package = "epos"), "\t")
```

---

readSecondLevelATC *Read the second level ATC classes from the file drugbankatc-secondlevel.map*

Description

Read the second level ATC classes from the file drugbankatc-secondlevel.map

Usage

```
readSecondLevelATC(filename, seperator)
```

Arguments

filename	the file name that is supposed to be drugbankatc-secondlevel.map
seperator	the csv file delimiter

Value

atchashsec a hash with second level ATC classes as keys and their names as values

Examples

```
atchashsec <-
  readSecondLevelATC(
  system.file("extdata", "drugbankatc-secondlevel.map", package = "epos"), "\t")
```

Index