Package: epos (via r-universe)

September 11, 2024

Type Package

Title Epilepsy Ontologies' Similarities

Version 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 BioASQ corpus. The ranked result lists of neurological drug terms co-occurring with terms from the epilepsy ontologies EpSO, ESSO, EPILONT, EPISEM and FENICS undergo further analysis. 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>, Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>, Mueller, Bernd and Rebholz-Schuhmann, Dietrich (2020) <doi:10.1007/978-3-030-43887-6_52>, and Mueller, Bernd et al. (2022) <doi:10.1186/s13326-021-00258-w>.

Depends R (>= 3.6.0)

License LGPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

URL https://github.com/bernd-mueller/epos

BugReports https://github.com/bernd-mueller/epos/issues

Imports hash, ggplot2, testthat, gridExtra, TopKLists, stringr, xtable, mongolite, stats, VennDiagram, cowplot

Suggests knitr, rmarkdown

Repository https://bernd-mueller.r-universe.dev

RemoteUrl https://github.com/bernd-mueller/epos

RemoteRef HEAD

RemoteSha 2fc4cc21b140d38fafa171ca1f7b6290f1be0b32

Contents

calcCosine . . . 3
calcDice . . . 3
calcDSEA . . . 4
calcEnrichment . . . 4
calcJaccard . . . 5
cosine . . . 5
createBaseTable . . . 6
createDashVectorForATC . . . 6
createJaccardPlotDBMeSH . . . 7
createJaccardPlotMeSHFive . . . 8
createNeuroTable . . . 9
createTanimotoBaseline . . . 10
dice . . . 11
doFullPlot . . . 12
drawVenn4 . . . 13
drawVenn4Doc . . . 14
drawVenn4DrugDoc . . . 14
drawVenn4Syn . . . 15
drawVenn5 . . . 15
drawVenn5Doc . . . 16
drawVenn5DrugDoc . . . 16
drawVenn5Syn . . . 17
drawVennGrid . . . 17
filterApprovedDrugs . . . 18
filterNeuroDrugs . . . 19
genDictListFromRawFreq . . . 20
getRefAll . . . 21
getTermMatrix . . . 21
jaccard . . . 22
plotDSEA . . . 22
plotEnrichment . . . 23
printTop10Drugs . . . 25
rawDrugNamesCoOcEPILONT . . . 26
rawDrugNamesCoOcEPISEM . . . 27
rawDrugNamesCoOcEpSO . . . 27
rawDrugNamesCoOcESSO . . . 28
rawDrugNamesCoOcFENICS . . . 29
readAtcMapIntoHashMapAtcCodesAtcNames . . . 29
readAtcMapIntoHashMapDrugNamesAtcCodes . . . 30
readSecondLevelATC . . . 31
sortTableByRefMatches . . . 31

Index **33**

`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))
```

calcDSEA *Calculate dsea scores of one list in comparison to reference list*

Description

Calculate dsea scores of one list in comparison to reference list

Usage

```
calcDSEA(alist, N)
```

Arguments

alist	list of drug names to be used for calculating dsea
N	numeric value with maximum length of lists for dsea calculation

Value

list with dsea scores

Examples

```
calcDSEA(c("Valproic acid", "Lamotrigine", "Ketamin"), 3)
```

calcEnrichment *Calculate enrichment of one list in comparison to reference list*

Description

Calculate enrichment of one list in comparison to reference list

Usage

```
calcEnrichment(alist)
```

Arguments

alist	the list to compare

Value

list with calculated enrichment used for plotting

Examples

```
a <- calcEnrichment(c("Clobazam","Oxcarbazepine"))
```

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

Description

Main function to call everything and produce the results

Usage

```
createBaseTable(coocepso, coocesso, coocepi, coocepisem, coocfenics)
```

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
`coocepisem`	list of drug names sorted by frequency co-occuring with EPISEM
`coocfenics`	list of drug names sorted by frequency co-occuring with FENICS

Value

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

Examples

```
utils::data(rawDrugNamesCoOcEpSO, package="epos")
utils::data(rawDrugNamesCoOcESSO, package="epos")
utils::data(rawDrugNamesCoOcEPILONT, package="epos")
utils::data(rawDrugNamesCoOcEPISEM, package="epos")
utils::data(rawDrugNamesCoOcFENICS, package="epos")
createBaseTable(coocepso = rawDrugNamesCoOcEpSO[1:150],
  coocesso=rawDrugNamesCoOcESSO[1:150],
  coocepi=rawDrugNamesCoOcEPILONT[1:150],
  coocepisem=rawDrugNamesCoOcEPISEM[1:150],
  coocfenics=rawDrugNamesCoOcFENICS[1:150])
```

`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 *Create the final resulting data frame*

Description

Create the final resulting data frame

Usage

```
createNeuroTable(atchashda, atchashsec, dneuromaxk)
```

Arguments

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(rawDrugNamesCoOcEpSO, package="epos")
utils::data(rawDrugNamesCoOcESSO, package="epos")
utils::data(rawDrugNamesCoOcEPILONT, package="epos")
utils::data(rawDrugNamesCoOcEPISEM, package="epos")
utils::data(rawDrugNamesCoOcFENICS, 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", "atc-secondlevel.map", package = "epos"), "\t")
epso <- rawDrugNamesCoOcEpSO
neuroepso <- filterNeuroDrugs(epso, atchashda)
esso <- rawDrugNamesCoOcESSO
neuroesso   <- filterNeuroDrugs(esso, atchashda)
epi <- rawDrugNamesCoOcEPILONT
neuroepi    <- filterNeuroDrugs(epi, atchashda)
episem <- rawDrugNamesCoOcEPISEM
neuroepisem <- filterNeuroDrugs(episem, atchashda)
fenics <- rawDrugNamesCoOcFENICS
neurofenics <- filterNeuroDrugs(fenics, atchashda)
mx <- max(
```

```
    c(length(neuroepso), length(neuroesso), length(neuroepi),
     length(neuroepisem), length(neurofenics)))
dneuro <-
  data.frame(EpSO = c(neuroepso, rep(1, (mx-length(neuroepso)))),
             ESSO = c(neuroesso, rep(1, (mx-length(neuroesso)))),
             EPILONT = c(neuroepi, rep(1, (mx-length(neuroepi)))),
             EPISEM = c(neuroepisem, rep(1, (mx-length(neuroepisem)))),
             FENICS = c(neurofenics, rep(1, (mx-length(neurofenics)))))
dneuromaxk <- TopKLists::calculate.maxK(dneuro, L=5, d=5, v=10)
neurotable <- createNeuroTable(atchashda, atchashsec, dneuromaxk)
```

`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.maxk

Value

jaccardepilepsyplot the ggplot object

Examples

```
utils::data(rawDrugNamesCoOcEpSO, package="epos")
utils::data(rawDrugNamesCoOcESSO, package="epos")
utils::data(rawDrugNamesCoOcEPILONT, package="epos")
utils::data(rawDrugNamesCoOcEPISEM, 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", "atc-secondlevel.map", package = "epos"), "\t")
tepso <- rawDrugNamesCoOcEpSO
tesso <- rawDrugNamesCoOcESSO
tepi <- rawDrugNamesCoOcEPILONT
tepisem <- rawDrugNamesCoOcEPISEM
tfenics <- rawDrugNamesCoOcFENICS

neuroepso <- filterNeuroDrugs(tepso, atchashda)
neuroesso <- filterNeuroDrugs(tesso, atchashda)
neuroepi <- filterNeuroDrugs(tepi, atchashda)
neuroepisem <- filterNeuroDrugs(tepisem, atchashda)
neurofenics <- filterNeuroDrugs(tfenics, atchashda)

dneuro <-
  data.frame(EpSO = neuroepso[1:210],
             ESSO = neuroesso[1:210],
             EPILONT = neuroepi[1:210],
             EPISEM = neuroepisem[1:210],
             FENICS = neurofenics[1:210])
dneuromaxk <- TopKLists::calculate.maxK(dneuro, 5, 5, 5)
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)
```

`drawVenn4` *Create quad Venn Diagramm for overlapping concepts between EpSO, ESSO, EPILONT and EPISEM*

Description

Create quad Venn Diagramm for overlapping concepts between EpSO, ESSO, EPILONT and EPISEM

Usage

```
drawVenn4()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn4.png", plot = drawVenn4(), width=240, height=160,
  units = "mm", dpi = 300)

## End(Not run)
```

drawVenn4Doc *Create quintuple Venn Diagramm for shared documents with co-occurrences of drug names between EpSO, ESSO, EPILONT and EPISEM*

Description

Create quintuple Venn Diagramm for shared documents with co-occurrences of drug names between EpSO, ESSO, EPILONT and EPISEM

Usage

```
drawVenn4Doc()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn4doc.png", plot = drawVenn4Doc(), width=240, height=160,
  units = "mm", dpi = 300)

## End(Not run)
```

drawVenn4DrugDoc *Create quad Venn Diagramm for shared documents with co-occurrences of drug names between EpSO, ESSO, EPILONT and EPISEM*

Description

Create quad Venn Diagramm for shared documents with co-occurrences of drug names between EpSO, ESSO, EPILONT and EPISEM

Usage

```
drawVenn4DrugDoc()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn4drugdoc.png", plot = drawVenn4DrugDoc(), width=240,
  height=160, units = "mm", dpi = 300)

## End(Not run)
```

`drawVenn4Syn` *Create quad Venn Diagramm for shared synonyms between EpSO, ESSO, EPILONT and EPISEM*

Description

Create quad Venn Diagramm for shared synonyms between EpSO, ESSO, EPILONT and EPISEM

Usage

```
drawVenn4Syn()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn4syn.png", plot = drawVenn4Syn(), width=240,
  height=160, units = "mm", dpi = 300)

## End(Not run)
```

`drawVenn5` *Create quintuple Venn Diagramm for overlapping concepts between EpSO, ESSO, EPILONT, EPISEM and FENICS*

Description

Create quintuple Venn Diagramm for overlapping concepts between EpSO, ESSO, EPILONT, EPISEM and FENICS

Usage

```
drawVenn5()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn5.png", plot = drawVenn5(), width=240, height=160,
  units = "mm", dpi = 300)

## End(Not run)
```

`drawVenn5Doc` *Create quintuple Venn Diagramm for shared documents between EpSO, ESSO, EPILONT, EPISEM and FENICS*

Description

Create quintuple Venn Diagramm for shared documents between EpSO, ESSO, EPILONT, EPISEM and FENICS

Usage

```
drawVenn5Doc()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn5doc.png", plot = drawVenn5Doc(), width=240, height=160,
  units = "mm", dpi = 300)

## End(Not run)
```

`drawVenn5DrugDoc` *Create quintuple Venn Diagramm for shared documents with co-occurrences of drug names between EpSO, ESSO, EPILONT, EPISEM and FENICS*

Description

Create quintuple Venn Diagramm for shared documents with co-occurrences of drug names between EpSO, ESSO, EPILONT, EPISEM and FENICS

Usage

```
drawVenn5DrugDoc()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn5drugdoc.png", plot = drawVenn5DrugDoc(), width=240,
  height=160, units = "mm", dpi = 300)

## End(Not run)
```

`drawVenn5Syn` *Create quintuple Venn Diagramm for shared synonyms between EpSO, ESSO, EPILONT, EPISEM and FENICS*

Description

Create quintuple Venn Diagramm for shared synonyms between EpSO, ESSO, EPILONT, EPISEM and FENICS

Usage

```
drawVenn5Syn()
```

Value

plot object

Examples

```
## Not run:
ggplot2::ggsave("venn5syn.png", plot = drawVenn5Syn(), width=240,
  height=160, units = "mm", dpi = 300)

## End(Not run)
```

`drawVennGrid` *Create plot_grid from multiple plots*

Description

Create plot_grid from multiple plots

Usage

```
drawVennGrid()
```

Value

plot object

Examples

```
## Not run:
  cowplot::plot_grid(drawVenn4 (), drawVenn4Syn(), drawVenn5Doc (),
    drawVenn5DrugDoc ())
  ggplot2::ggsave("vennAB.png", plot = cowplot::plot_grid(drawVenn4 (),
    drawVenn4Syn(), labels = c('A', 'B'), ncol = 1), width=240, height=320,
    units = "mm", dpi = 300)
  ggplot2::ggsave("vennAB.png", plot = cowplot::plot_grid(drawVenn4 (),
    drawVenn4Syn(), labels = c('Concepts:', 'Synonyms:'), ncol = 1), width=240,
    height=320,  units = "mm", dpi = 300)
  ggplot2::ggsave("vennCD.png", plot = cowplot::plot_grid(drawVenn5Doc (),
    drawVenn5DrugDoc(), labels = c('Documents with B-Terms:',
    'Documents with B- and C-Terms:'), ncol = 1), width=240, height=320,
    units = "mm", dpi = 300)
  ggplot2::ggsave("vennCD.png", plot = cowplot::plot_grid(drawVenn5Doc (),
    drawVenn5DrugDoc(), labels = c('Documents with B-Terms:',
   'Documents with B- and C-Terms:'), ncol = 1), width=240, height=320,  units = "mm",
   dpi = 300)
  ggplot2::ggsave("vennCD.png", plot = cowplot::plot_grid(drawVenn4Doc (),
    drawVenn4DrugDoc(), labels = c('Documents with B-Terms:',
    'Documents with B- and C-Terms:'), ncol = 1), width=240, height=320,
    units = "mm", dpi = 300)
  ggplot2::ggsave("vennCD.png", plot = cowplot::plot_grid(drawVenn4Doc (),
    drawVenn4DrugDoc(), labels = c('Documents\nwith B-Terms:             ',
    'Documents\nwith B- and C-Terms:'), ncol = 1), width=240, height=320,
    units = "mm", dpi = 300)
  ggplot2::ggsave("vennAB.png", plot = cowplot::plot_grid(drawVenn4 (),
    drawVenn4Syn(), labels = c('i) Concepts:', 'ii) Synonyms:'), ncol = 1),
    width=240, height=320,  units = "mm", dpi = 300)
  ggplot2::ggsave("vennCD.png", plot = cowplot::plot_grid(NULL,
    drawVenn4Doc (), drawVenn4DrugDoc(),
    labels = c('iii) Documents with B-Terms:',
    'iv) Documents with B- and C-Terms:'), ncol = 1,
    label_x = c(-0.105, -0.14), label_fontfamily = "Arial Nova Light",
    label_fontface = "bold"), width=240, height=320,  units = "mm", dpi = 300)

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

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

Usage

```
genDictListFromRawFreq(topfreqdictraw)

genDictListFromRawFreq(topfreqdictraw)
```

Arguments

`topfreqdictraw`	list with terms from a dictionary sorted by frequency

Value

a sorted list of terms

a sorted list of terms

Examples

```
## Not run:
genDictListFromRawFreq(epi)

## End(Not run)
utils::data(rawDrugNamesCoOcEpSO, package="epos")
genDictListFromRawFreq(rawDrugNamesCoOcEpSO)
```

getRefAll — *Retrieve the list of drugs from the union of all reference lists*

Description

Retrieve the list of drugs from the union of all reference lists

Usage

```
getRefAll()
```

Value

list of drugs from all reference lists

Examples

```
d <- getRefAll()
```

getTermMatrix — *Receives a sorted hashmap with found entities from a dictionary*

Description

Receives a sorted hashmap with found entities from a dictionary

Usage

```
getTermMatrix(dictionary, database, collection)
```

Arguments

dictionary	Character vector that is the name of a dicitonary having pre-calculated stats. This can be MeSH, DrugBank, Agrovoc, EpSO, ESSO, or EPILONT
database	the name of the MongoDB database to be used
collection	the name of the MongoDB collection to be used

Value

a sorted hashmap containing all found entities from the respective dictionaries with frequencies

Examples

```
## Not run:
mesh <- getTermMatrix("MeSH")

## End(Not run)
```

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)
```

plotDSEA *Plotting functions for DSEA lists*

Description

Plotting functions for DSEA lists

Usage

```
plotDSEA(dsepso, dsesso, dsepi, dsepisem, dsfenics, dsspace, k)
```

Arguments

dsepso	list with enrichment for EpSO
dsesso	list with enrichment for ESSO
dsepi	list with enrichment for EPILONT
dsepisem	list with enrichment for EPISEM
dsfenics	list with enrichment for FENICS
dsspace	list with enrichment for the combined ranked list
k	numeric value for the length to be plotted

Value

the plot object

Examples

```
utils::data(rawDrugNamesCoOcEpSO, package="epos")
utils::data(rawDrugNamesCoOcESSO, package="epos")
utils::data(rawDrugNamesCoOcEPILONT, package="epos")
utils::data(rawDrugNamesCoOcEPISEM, package="epos")
utils::data(rawDrugNamesCoOcFENICS, package="epos")
atchashda <-
    readAtcMapIntoHashMapDrugNamesAtcCodes(
        system.file("extdata", "db-atc.map", package = "epos"), "\t")
epso <- rawDrugNamesCoOcEpSO
neuroepso <- filterNeuroDrugs(epso, atchashda)
esso <- rawDrugNamesCoOcESSO
neuroesso   <- filterNeuroDrugs(esso, atchashda)
epi <- rawDrugNamesCoOcEPILONT
neuroepi    <- filterNeuroDrugs(epi, atchashda)
episem <- rawDrugNamesCoOcEPISEM
neuroepisem <- filterNeuroDrugs(episem, atchashda)
fenics <- rawDrugNamesCoOcFENICS
neurofenics <- filterNeuroDrugs(fenics, atchashda)
mx <- max(
    c(length(neuroepso), length(neuroesso), length(neuroepi),
      length(neuroepisem), length(neurofenics)))
dneuro <-
  data.frame(EpSO = c(neuroepso, rep("", (mx-length(neuroepso)))),
             ESSO = c(neuroesso, rep("", (mx-length(neuroesso)))),
             EPILONT = c(neuroepi, rep("", (mx-length(neuroepi)))),
             EPISEM = c(neuroepisem, rep("", (mx-length(neuroepisem)))),
             FENICS = c(neurofenics, rep("", (mx-length(neurofenics)))))
dneuromaxk <- TopKLists::calculate.maxK(dneuro, L=5, d=5, v=5)
neurospace <- as.character(dneuromaxk$topkspace)
dsepso <- calcDSEA(neuroepso, mx)
dsesso <- calcDSEA(neuroesso, mx)
dsepi <- calcDSEA(neuroepi, mx)
dsepisem <- calcDSEA(neuroepisem, mx)
dsfenics <- calcDSEA(neurofenics, mx)
dsspace  <-  calcDSEA (neurospace, mx)
p <- plotDSEA(dsepso, dsesso, dsepi, dsepisem, dsfenics, dsspace, dneuromaxk$maxK)
## Not run:
ggplot2::ggsave("dsea.png",
   p <- plotDSEA(dsepso, dsesso, dsepi, dsepisem, dsfenics, dsspace,
   dneuromaxk$maxK), width=480, height=320, units = "mm", dpi = 300)

## End(Not run)
```

---

plotEnrichment *Plotting functions for enrichment lists*

Description

Plotting functions for enrichment lists

Usage

```
plotEnrichment(enepso, enesso, enepi, enepisem, enfenics, enspace, k)
```

Arguments

enepso	list with enrichment for EpSO
enesso	list with enrichment for ESSO
enepi	list with enrichment for EPILONT
enepisem	list with enrichment for EPISEM
enfenics	list with enrichment for FENICS
enspace	list with enrichment for the combined ranked list
k	numeric value for the length to be plotted

Value

the plot object

Examples

```
utils::data(rawDrugNamesCoOcEpSO, package="epos")
utils::data(rawDrugNamesCoOcESSO, package="epos")
utils::data(rawDrugNamesCoOcEPILONT, package="epos")
utils::data(rawDrugNamesCoOcEPISEM, package="epos")
utils::data(rawDrugNamesCoOcFENICS, package="epos")
atchashda <-
    readAtcMapIntoHashMapDrugNamesAtcCodes(
        system.file("extdata", "db-atc.map", package = "epos"), "\t")
epso <- rawDrugNamesCoOcEpSO
neuroepso <- filterNeuroDrugs(epso, atchashda)
esso <- rawDrugNamesCoOcESSO
neuroesso   <- filterNeuroDrugs(esso, atchashda)
epi <- rawDrugNamesCoOcEPILONT
neuroepi    <- filterNeuroDrugs(epi, atchashda)
episem <- rawDrugNamesCoOcEPISEM
neuroepisem <- filterNeuroDrugs(episem, atchashda)
fenics <- rawDrugNamesCoOcFENICS
neurofenics <- filterNeuroDrugs(fenics, atchashda)
mx <- max(
    c(length(neuroepso), length(neuroesso), length(neuroepi),
      length(neuroepisem), length(neurofenics)))
dneuro <-
  data.frame(EpSO = c(neuroepso, rep("", (mx-length(neuroepso)))),
             ESSO = c(neuroesso, rep("", (mx-length(neuroesso)))),
             EPILONT = c(neuroepi, rep("", (mx-length(neuroepi)))),
             EPISEM = c(neuroepisem, rep("", (mx-length(neuroepisem)))),
```

```
             FENICS = c(neurofenics, rep("", (mx-length(neurofenics)))))
dneuromaxk <- TopKLists::calculate.maxK(dneuro, L=5, d=5, v=5)
neurospace <- as.character(dneuromaxk$topkspace)
enepso <- calcEnrichment(neuroepso)
enesso <- calcEnrichment(neuroesso)
enepi <- calcEnrichment(neuroepi)
enepisem <- calcEnrichment(neuroepisem)
enfenics <- calcEnrichment(neurofenics)
enspace <- calcEnrichment (neurospace)
p <- plotEnrichment(enepso, enesso, enepi, enepisem, enfenics, enspace, dneuromaxk$maxK)
```

`printTop10Drugs` *Print Top 10 Drugs*

Description

Print Top 10 Drugs

Usage

```
printTop10Drugs(neuroepso, neuroesso, neuroepi, neuroepisem, neurofenics)
```

Arguments

`neuroepso`	Ranked list of drug names co-occurring with EpSO
`neuroesso`	Ranked list of drug names co-occurring with ESSO
`neuroepi`	Ranked list of drug names co-occurring with EPILONT
`neuroepisem`	Ranked list of drug names co-occurring with EPISEM
`neurofenics`	Ranked list of drug names co-occurring with FENICS

Value

data frame with top 10 drugs for each ontology

Examples

```
utils::data(rawDrugNamesCoOcEpSO, package="epos")
utils::data(rawDrugNamesCoOcESSO, package="epos")
utils::data(rawDrugNamesCoOcEPILONT, package="epos")
utils::data(rawDrugNamesCoOcEPISEM, package="epos")
utils::data(rawDrugNamesCoOcFENICS, 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", "atc-secondlevel.map", package = "epos"), "\t")
epso <- rawDrugNamesCoOcEpSO
neuroepso <- filterNeuroDrugs(epso, atchashda)
esso <- rawDrugNamesCoOcESSO
neuroesso   <- filterNeuroDrugs(esso, atchashda)
epi <- rawDrugNamesCoOcEPILONT
neuroepi    <- filterNeuroDrugs(epi, atchashda)
episem <- rawDrugNamesCoOcEPISEM
neuroepisem <- filterNeuroDrugs(episem, atchashda)
fenics <- rawDrugNamesCoOcFENICS
neurofenics <- filterNeuroDrugs(fenics, atchashda)
top10table <- printTop10Drugs(neuroepso, neuroesso, neuroepi, neuroepisem, neurofenics)
## Not run:
  print(xtable::xtable(top10table, type = "latex"),
    file = "top10table.tex")

## End(Not run)
```

`rawDrugNamesCoOcEPILONT` *List drug terms with their frequency co-occurring with terms from the EPILONT ontology in publications since 2015 from the BioASQ 2020 corpus.*

Description

List drug terms with their frequency co-occurring with terms from the EPILONT ontology in publications since 2015 from the BioASQ 2020 corpus.

Usage

```
rawDrugNamesCoOcEPILONT
```

Format

A named list of drug term frequencies

Source

The text mining workflows for data generation are described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558>, Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>, and Mueller, Bernd and Rebholz-Schuhmann, Dietrich (2020) <doi:10.1007/978-3-030-43887-6_52>. The source data set for generating the data co-occurrence lists is the BioASQ 2020 corpus. The source ontology for the creation of the dictionary is the Epilepsy Ontology (EPILONT) from https://bioportal.bioontology.org/ontologies/EPILONT

Examples

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

rawDrugNamesCoOcEPISEM *List drug terms with their frequency co-occurring with terms from the EPISEM ontology in publications since 2015 from the BioASQ 2020 corpus.*

Description

List drug terms with their frequency co-occurring with terms from the EPISEM ontology in publications since 2015 from the BioASQ 2020 corpus.

Usage

```
rawDrugNamesCoOcEPISEM
```

Format

A named list of drug term frequencies

Source

The text mining workflows for data generation are described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558>, Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>, and Mueller, Bernd and Rebholz-Schuhmann, Dietrich (2020) <doi:10.1007/978-3-030-43887-6_52>. The source data set for generating the data co-occurrence lists is the BioASQ 2020 corpus. The source ontology for the creation of the dictionary is the Epilepsy Semiology Ontology (EPISEM) from https://bioportal.bioontology.org/ontologies/EPISEM

Examples

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

rawDrugNamesCoOcEpSO *List drug terms with their frequency co-occurring with terms from the EpSO ontology in publications since 2015 from the BioASQ 2020 corpus.*

Description

List drug terms with their frequency co-occurring with terms from the EpSO ontology in publications since 2015 from the BioASQ 2020 corpus.

Usage

```
rawDrugNamesCoOcEpSO
```

Format

A named list of drug term frequencies

Source

The text mining workflows for data generation are described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558>, Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>, and Mueller, Bernd and Rebholz-Schuhmann, Dietrich (2020) <doi:10.1007/978-3-030-43887-6_52>. The source data set for generating the data co-occurrence lists is the BioASQ 2020 corpus. The source ontology for the creation of the dictionary is the Epilepsy and Seizure Ontology (EpSO) from https://bioportal.bioontology.org/ontologies/EPSO

Examples

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

---

rawDrugNamesCoOcESSO *List drug terms with their frequency co-occurring with terms from the ESSO ontology in publications since 2015 from the BioASQ 2020 corpus.*

Description

List drug terms with their frequency co-occurring with terms from the ESSO ontology in publications since 2015 from the BioASQ 2020 corpus.

Usage

```
rawDrugNamesCoOcESSO
```

Format

An object of class `character` of length 8620.

Source

The text mining workflows for data generation are described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558>, Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>, and Mueller, Bernd and Rebholz-Schuhmann, Dietrich (2020) <doi:10.1007/978-3-030-43887-6_52>. The source data set for generating the data co-occurrence lists is the BioASQ 2020 corpus. The source ontology for the creation of the dictionary is Epilepsy Syndrome Seizure Ontology (ESSO) from https://bioportal.bioontology.org/ontologies/ESSO

Examples

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

rawDrugNamesCoOcFENICS *List drug terms with their frequency co-occurring with terms from the FENICS ontology in publications from the BioASQ 2020 corpus.*

Description

List drug terms with their frequency co-occurring with terms from the FENICS ontology in publications from the BioASQ 2020 corpus.

Usage

```
rawDrugNamesCoOcFENICS
```

Format

A named list of drug term frequencies

Source

The text mining workflows for data generation are described in Mueller, Bernd and Hagelstein, Alexandra (2016) <doi:10.4126/FRL01-006408558>, Mueller, Bernd et al. (2017) <doi:10.1007/978-3-319-58694-6_22>, and Mueller, Bernd and Rebholz-Schuhmann, Dietrich (2020) <doi:10.1007/978-3-030-43887-6_52>. The source data set for generating the data co-occurrence lists is the BioASQ 2020 corpus. The source ontology for the creation of the dictionary is the Functional Epilepsy Nomenclature for Ion Channels (FENICS) from https://bioportal.bioontology.org/ontologies/FENICS

Examples

```
utils::data(rawDrugNamesCoOcFENICS, 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 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 atc-secondlevel.map*

Description

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

Usage

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

Arguments

filename	the file name that is supposed to be atc-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", "atc-secondlevel.map", package = "epos"), "\t")
```

sortTableByRefMatches *Sort table by scoring for each row*

Description

Sort table by scoring for each row

Usage

```
sortTableByRefMatches(dntk)
```

Arguments

dntk	the table returned from writeNeuroTable

Value

the sorted table

Examples

```
utils::data(rawDrugNamesCoOcEpSO, package="epos")
utils::data(rawDrugNamesCoOcESSO, package="epos")
utils::data(rawDrugNamesCoOcEPILONT, package="epos")
utils::data(rawDrugNamesCoOcEPISEM, package="epos")
utils::data(rawDrugNamesCoOcFENICS, 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", "atc-secondlevel.map", package = "epos"), "\t")
epso <- rawDrugNamesCoOcEpSO
neuroepso <- filterNeuroDrugs(epso, atchashda)
esso <- rawDrugNamesCoOcESSO
neuroesso   <- filterNeuroDrugs(esso, atchashda)
epi <- rawDrugNamesCoOcEPILONT
neuroepi    <- filterNeuroDrugs(epi, atchashda)
episem <- rawDrugNamesCoOcEPISEM
neuroepisem <- filterNeuroDrugs(episem, atchashda)
fenics <- rawDrugNamesCoOcFENICS
neurofenics <- filterNeuroDrugs(fenics, atchashda)
mx <- max(
    c(length(neuroepso), length(neuroesso), length(neuroepi),
      length(neuroepisem), length(neurofenics)))
dneuro <-
  data.frame(EpSO = c(neuroepso, rep("", (mx-length(neuroepso)))),
             ESSO = c(neuroesso, rep("", (mx-length(neuroesso)))),
             EPILONT = c(neuroepi, rep("", (mx-length(neuroepi)))),
             EPISEM = c(neuroepisem, rep("", (mx-length(neuroepisem)))),
             FENICS = c(neurofenics, rep("", (mx-length(neurofenics)))))
suppressWarnings(dneuromaxk <- TopKLists::calculate.maxK(dneuro, L=5, d=5, v=5))
neurotable <- createNeuroTable(atchashda, atchashsec, dneuromaxk)
sortedNeuroTable <- sortTableByRefMatches(neurotable)
print(xtable::xtable(sortedNeuroTable, type = "latex"),
  file = "sortedNeuroTable.tex",
  include.rownames=FALSE)
```

Index

∗ **datasets**
    rawDrugNamesCoOcEPILONT, 26
    rawDrugNamesCoOcEPISEM, 27
    rawDrugNamesCoOcEpSO, 27
    rawDrugNamesCoOcESSO, 28
    rawDrugNamesCoOcFENICS, 29

calcCosine, 3
calcDice, 3
calcDSEA, 4
calcEnrichment, 4
calcJaccard, 5
cosine, 5
createBaseTable, 6
createDashVectorForATC, 6
createJaccardPlotDBMeSH, 7
createJaccardPlotMeSHFive, 8
createNeuroTable, 9
createTanimotoBaseline, 10

dice, 11
doFullPlot, 12
drawVenn4, 13
drawVenn4Doc, 14
drawVenn4DrugDoc, 14
drawVenn4Syn, 15
drawVenn5, 15
drawVenn5Doc, 16
drawVenn5DrugDoc, 16
drawVenn5Syn, 17
drawVennGrid, 17

filterApprovedDrugs, 18
filterNeuroDrugs, 19

genDictListFromRawFreq, 20
getRefAll, 21
getTermMatrix, 21

jaccard, 22

plotDSEA, 22
plotEnrichment, 23
printTop10Drugs, 25

rawDrugNamesCoOcEPILONT, 26
rawDrugNamesCoOcEPISEM, 27
rawDrugNamesCoOcEpSO, 27
rawDrugNamesCoOcESSO, 28
rawDrugNamesCoOcFENICS, 29
readAtcMapIntoHashMapAtcCodesAtcNames, 29
readAtcMapIntoHashMapDrugNamesAtcCodes, 30
readSecondLevelATC, 31

sortTableByRefMatches, 31