Package 'getDEE2'

November 6, 2024

Title Programmatic access to the DEE2 RNA expression dataset

Version 1.16.0

Description Digital Expression Explorer 2 (or DEE2 for short) is a repository of processed RNA-seq data in the form of counts. It was designed so that researchers could undertake re-analysis and meta-analysis of published RNA-seq studies quickly and easily. As of April 2020, over 1 million SRA datasets have been processed. This package provides an R interface to access these expression data. More information about the DEE2 project can be found at the project homepage (http://dee2.io) and main publication (https://doi.org/10.1093/gigascience/giz022).

Depends R (>= 4.0)

Imports stats, utils, SummarizedExperiment, htm2txt

Suggests knitr, testthat, rmarkdown

License GPL-3

Encoding UTF-8

URL https://github.com/markziemann/getDEE2

LazyData true

RoxygenNote 7.1.1

biocViews GeneExpression, Transcriptomics, Sequencing

VignetteBuilder knitr

BugReport https://github.com/markziemann/getDEE2

git_url https://git.bioconductor.org/packages/getDEE2

git_branch RELEASE_3_20

git_last_commit a328347

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2024-11-05

Author Mark Ziemann [aut, cre],
Antony Kaspi [aut]

Maintainer Mark Ziemann <mark.ziemann@gmail.com>

Contents

`getDEE2` *getDEE2: Programmatic access to the DEE2 RNA expression dataset*

Description

Digital Expression Explorer 2 (or DEE2 for short) is a repository of processed RNA-seq data in the form of counts. It was designed so that researchers could undertake re-analysis and meta-analysis of published RNA-seq studies quickly and easily. This package provides an R interface to access these expression data. More information about the DEE2 project can be found at the project homepage (http://dee2.io) and main publication (https://doi.org/10.1093/gigascience/giz022).

The getDEE2 function fetches gene expression data from the DEE2 database of RNA sequencing data and returns it as a SummarizedExperiment object.

Usage

```
getDEE2(
  species,
  SRRvec,
  counts = "GeneCounts",
  metadata = NULL,
  outfile = NULL,
  legacy = FALSE,
  baseURL = "http://dee2.io/cgi-bin/request.sh?",
  ...
)
```

Arguments

species	A character string matching the species of interest.
SRRvec	A character string or vector of SRA run accession numbers.
counts	A string, either 'GeneCounts', 'TxCounts' or 'Tx2Gene'. When 'GeneCounts' is specified, STAR gene level counts are returned. When 'TxCounts' is specified, kallisto transcript counts are returned. When 'Tx2Gene' is specified, kallisto counts aggregated (by sum) on gene are returned. If left blank, "GeneCounts" will be fetched.
metadata	(Optional) name of R object for the meta data. Providing the metadata will speed up performance if multiple queries are made in a session. If left blank, the metadata will be fetched once again.
outfile	An optional file name for the downloaded dataset.
legacy	Whether data should be returned in the legacy (list) format. Default is FALSE. Leave this FALSE if you want to receive data as Summarized experiment.
baseURL	The base URL of the service. Leave this as the default URL unless you want to download from a 3rd party mirror.
...	Additional parameters to be passed to download.file.

Value

a SummarizedExperiment object.

Examples

```
# Example workflow
# Fetch metadata
mdat <- getDEE2Metadata("celegans")
# filter metadata for SRA project SRP009256
mdat1 <- mdat[which(mdat$SRP_accession %in% "SRP009256"),]
# create a vector of SRA run accessions to fetch
SRRvec <- as.vector(mdat1$SRR_accession)
# obtain the data as a SummarizedExperiment
x <- getDEE2("celegans",SRRvec,metadata=mdat,counts="GeneCounts")
# Next, downstream analysis with your favourite Bioconductor tools :)
x<-getDEE2("ecoli",c("SRR1613487","SRR1613488"))
```

---

getDEE2Metadata *Get DEE2 Metadata*

Description

This function fetches the short metadata for the species of interest.

Usage

```
getDEE2Metadata(species, outfile = NULL, ...)
```

Arguments

species	A character string matching a species of interest.
outfile	Optional filename.
...	Additional parameters to be passed to download.file.

Value

a table of metadata.

Examples

```
ecoli_metadata <- getDEE2Metadata("ecoli")
```

---

getDEE2_bundle *Get a DEE2 project bundle*

Description

The getDEE2_bundle function fetches gene expression data from DEE2. This function will only work if all SRA runs have been successfully processed for an SRA project. This function returns a SummarizedExperiment object.

Usage

```
getDEE2_bundle(
  species,
  query,
  col,
  counts = "GeneCounts",
  bundles = NULL,
  legacy = FALSE,
  baseURL = "http://dee2.io/huge/",
  ...
)
```

Arguments

species	A character string matching the species of interest.
query	A character string, such as the SRA project accession number or the GEO series accession number
col	the column name to be queried, usually "SRP_accession" for SRA project accession or "GSE_accession" for GEO series accession.
counts	A string, either 'GeneCounts', 'TxCounts' or 'Tx2Gene'. When 'GeneCounts' is specified, STAR gene level counts are returned. When 'TxCounts' is specified, kallisto transcript counts are returned. When 'Tx2Gene' is specified, kallisto counts aggregated (by sum) on gene are returned. If left blank, "GeneCounts" will be fetched.
bundles	optional table of previously downloaded bundles. providing this will speed up performance if multiple queries are made in a session. If left blank, the bundle list will be fetched again.

legacy	Whether data should be returned in the legacy (list) format. Default is FALSE. Leave this FALSE if you want to receive data as Summarized experiment.
baseURL	The base URL of the service. Leave this as the default URL unless you want to download from a 3rd party mirror.
...	Additional parameters to be passed to download.file.

Value

a SummarizedExperiment object.

Examples

```
x <- getDEE2_bundle("celegans", "SRP133403",col="SRP_accession")
```

---

list_bundles *Get a table of all completed projects at DEE2*

Description

This function fetches a table listing all completed projects that are available at DEE2

Usage

```
list_bundles(species)
```

Arguments

species	A character string matching a species of interest.

Value

a table of project bundles available at DEE2.io/huge

Examples

```
bundles <- list_bundles("celegans")
```

---

loadFullMeta *Load Full Metadata*

Description

This function loads the full metadata, which contains many fields.

Usage

```
loadFullMeta(zipname)
```

Arguments

zipname	Path to the zipfile.

Value

a dataframe of full metadata.

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),outfile="mydata.zip")
y <- loadFullMeta("mydata.zip")
```

`loadGeneCounts` *Load Gene Counts*

Description

This function loads STAR gene level counts from a downloaded zip file.

Usage

```
loadGeneCounts(zipname)
```

Arguments

`zipname`	Path to the zipfile.

Value

a dataframe of gene expression counts.

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),outfile="mydata.zip")
y <- loadGeneCounts("mydata.zip")
```

`loadGeneInfo` *Load Gene Info*

Description

This function loads gene information. This information includes gene names and lengths which is useful for downstream analysis.

Usage

```
loadGeneInfo(zipname)
```

Arguments

`zipname`	Path to the zipfile.

Value

a dataframe of gene information.

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),outfile="mydata.zip")
y <- loadGeneInfo("mydata.zip")
```

---

loadQcMx *Load Quality Control Info*

Description

This function loads quality control data. More information about the QC metrics is available from the project github page: https://github.com/markziemann/dee2/blob/master/qc/qc_metrics.md

Usage

```
loadQcMx(zipname)
```

Arguments

zipname	Path to the zipfile.

Value

a dataframe of quality control metrics.

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),outfile="mydata.zip")
y <- loadQcMx("mydata.zip")
```

---

loadSummaryMeta *Load Summary Metadata*

Description

This function loads the summary metadata, which are the most relevant SRA accession numbers.

Usage

```
loadSummaryMeta(zipname)
```

Arguments

zipname	Path to the zipfile.

Value

a dataframe of summary metadata.

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),outfile="mydata.zip")
y <- loadSummaryMeta("mydata.zip")
```

loadTxCounts *Load Transcript Counts*

Description

This function loads Kallisto transcript level counts from a downloaded zip file.

Usage

```
loadTxCounts(zipname)
```

Arguments

zipname	Path to the zipfile.

Value

a dataframe of transcript expression counts.

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),outfile="mydata.zip")
y <- loadTxCounts("mydata.zip")
```

loadTxInfo *Load Transcript Info*

Description

This function loads transcript information. This information includes transcript lengths, corresponding parent gene accession and gene symbol that might be useful for downstream analysis.

Usage

```
loadTxInfo(zipname)
```

Arguments

zipname	Path to the zipfile.

Value

a dataframe of transcript info

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),outfile="mydata.zip")
y <- loadTxInfo("mydata.zip")
```

queryDEE2 *Query Whether a DEE2 Dataset is Available*

Description

This function sends a query to check whether a dataset is available or not.

Usage

```
queryDEE2(species, SRRvec, metadata = NULL, ...)
```

Arguments

species	A character string matching a species of interest.
SRRvec	A character string or vector thereof of SRA run accession numbers.
metadata	optional R object of DEE2 metadata to query.
...	Additional parameters to be passed to download.file.

Value

a list of datasets that are present and absent.

Examples

```
x <- queryDEE2("ecoli",c("SRR1067773","SRR5350513"))
```

query_bundles *Query whether a project bundle is available from DEE2*

Description

This function sends a query to check whether a dataset is available or not.

Usage

```
query_bundles(species, query, col, bundles = NULL)
```

Arguments

species	A character string matching a species of interest.
query	A character string, such as the SRA project accession number or the GEO series accession number
col	the column name to be queried, usually "SRP_accession" for SRA project accession or "GSE_accession" for GEO series accession.
bundles	optional table of previously downloaded bundles.

Value

a list of datasets that are present and absent.

Examples

```
query_bundles("celegans", c("SRP133403","SRP133439"), col = "SRP_accession")
```

`se` *Create summarizedExperiment object*

Description

This function creates a SummarizedExperiment object from a legacy getDEE2 dataset

Usage

```
se(x, counts = "GeneCounts")
```

Arguments

`x`	a getDEE2 object.
`counts`	select "GeneCounts" for STAR based gene counts, "TxCounts" for kallisto transcript level counts or "Tx2Gene" for transcript counts aggregated to gene level. Default is "GeneCounts"

Value

a SummarizedExperiment object

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),legacy=TRUE)
y <- se(x)
```

`srx_agg` *Summarized run data to experiments*

Description

Sometimes, each SRA experiment data is represented in two or more runs and they need to be aggregated.

Usage

```
srx_agg(x, counts = "GeneCounts")
```

Arguments

`x`	a getDEE2 object.
`counts`	select "GeneCounts" for STAR based gene counts, "TxCounts" for kallisto transcript level counts or "Tx2Gene" for transcript counts aggregated to gene level. Default is "GeneCounts"

Value

a dataframe with gene expression data summarised to SRA experiment accession numbers rather than run accession numbers.

Examples

```
x <- getDEE2("ecoli",c("SRR1613487","SRR1613488"),legacy=TRUE)
y <- srx_agg(x)
```

Tx2Gene *Aggregate Transcript Counts to Gene-Level Counts*

Description

This function converts Kallisto transcript-level expression estimates to gene-level estimates. Counts for each transcript are summed to get an aggregated gene level score.

Usage

```
Tx2Gene(x)
```

Arguments

x	a getDEE2 object.

Value

a dataframe of gene expression counts.

Examples

```
x <- getDEE2("scerevisiae",c("SRR1755149","SRR1755150"),legacy=TRUE)
x <- Tx2Gene(x)
```

Index