Package ‘MetaNeighbor’

March 28, 2024

Type Package
Title Single cell replicability analysis
Version 1.22.0
Description MetaNeighbor allows users to quantify cell type replicability across datasets using neighbor voting.
biocViews ImmunoOncology, GeneExpression, GO, MultipleComparison, SingleCell, Transcriptomics
License MIT + file LICENSE
Depends R(>= 3.5)
Imports grDevices, graphics, methods, stats (>= 3.4), utils (>= 3.4), Matrix (>= 1.2), matrixStats (>= 0.54), beanplot (>= 1.2), gplots (>= 3.0.1), RColorBrewer (>= 1.1.2), SummarizedExperiment (>= 1.12), SingleCellExperiment, igraph, dplyr, tidyr, tibble, ggplot2
Suggests knitr (>= 1.17), rmarkdown (>= 1.6), testthat (>= 1.0.2), UpSetR
LazyData true
RoxygenNote 7.1.1
VignetteBuilder knitr
Maintainer Stephan Fischer <fischer@cshl.edu>
git_url https://git.bioconductor.org/packages/MetaNeighbor
git_branch RELEASE_3_18
git_last_commit a036c58
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-03-27
Author Megan Crow [aut, cre], Sara Ballouz [ctb], Manthan Shah [ctb], Stephan Fischer [ctb], Jesse Gillis [aut]
R topics documented:

extendClusterSet ................................................. 2
extractMetaClusters ............................................. 3
getCellType ....................................................... 4
getStudyId ......................................................... 4
ggPlotHeatmap ..................................................... 5
GOhuman ............................................................ 5
GOmouse ............................................................ 6
makeClusterGraph .................................................. 6
makeClusterName .................................................... 7
mergeSCE ............................................................ 7
MetaNeighbor ....................................................... 8
MetaNeighborUS .................................................... 9
mn_data ............................................................. 11
neighborVoting ..................................................... 11
orderCellTypes ..................................................... 13
plotBPlot ........................................................... 13
plotClusterGraph .................................................. 14
plotDotPlot ........................................................ 15
plotHeatmap ......................................................... 16
plotHeatmapPretrained ............................................ 16
plotMetaClusters ................................................... 17
plotUpset ........................................................... 18
scoreMetaClusters .................................................. 19
splitClusters ......................................................... 20
splitTestClusters .................................................... 20
splitTrainClusters ................................................... 21
standardizeLabel ................................................... 21
subsetClusterGraph ................................................ 22
topHits ............................................................... 23
topHitsByStudy ..................................................... 24
trainModel .......................................................... 25
variableGenes ....................................................... 26

Index 27

---

extendClusterSet  

Extend cluster set to nearest neighbors on cluster graph.

Description

Note that the graph is directed, i.e. neighbors are retrieved by following arrows that start from the initial clusters.

Usage

extendClusterSet(graph, initial_set, max_neighbor_distance = 2)
**extractMetaClusters**

**Arguments**

- **graph**
  Graph in igraph format generated by makeClusterGraph.

- **initial_set**
  Vector of cluster labels

- **max_neighbor_distance**
  Include more distantly related nodes by performing neighbor extension max_neighbor_distance rounds.

**Value**

Character vector including initial cluster set and all neighboring clusters (if any).

---

**Description**

Note that meta-clusters are *not* cliques, but connected components, e.g., if 1<->2 and 1<->3 are reciprocal top hits, 1, 2, 3 is a meta-cluster, independently from the relationship between 2 and 3.

**Usage**

```r
extractMetaClusters(best_hits, threshold = 0)
```

**Arguments**

- **best_hits**
  Matrix of AUROCs produced by MetaNeighborUS.

- **threshold**
  AUROC threshold. Two clusters belong to the same meta-cluster if they are reciprocal top hits and their similarity exceeds the threshold *both* ways (AUROC(1->2) > threshold *AND* AUROC(2->1) > threshold).

**Value**

A named list, where names are default meta-cluster names, and values are vectors of cluster names, one vector per meta-cluster. The last element of the list is called "outliers" and contains all clusters that had no match in any other dataset.
getCellType  
*Return cell type from a label in format 'study_id|cell_type'*

**Description**  
Return cell type from a label in format 'study_id|cell_type'

**Usage**  
getCellType(cluster_name)

**Arguments**  
cluster_name  
Character vector containing cluster names in the format study_id|cell_type.

**Value**  
Character vector containing all cell type names.

getStudyId  
*Return study ID from a label in format 'study_id|cell_type'*

**Description**  
Return study ID from a label in format 'study_id|cell_type'

**Usage**  
getStudyId(cluster_name)

**Arguments**  
cluster_name  
Character vector containing cluster names in the format study_id|cell_type.

**Value**  
Character vector containing all study ids.
ggPlotHeatmap

Plots symmetric AUROC heatmap, clustering cell types by similarity.

Description
This function is a ggplot alternative to plotHeatmap (without the cell type dendrogram).

Usage
ggPlotHeatmap(aurocs, label_size = 10)

Arguments
aurocs A square AUROC matrix as returned by MetaNeighborUS.
label_size Font size of cell type labels along the heatmap (default is 10).

Value
A ggplot object.

See Also
plotHeatmap

GOhuman

Description
List containing gene symbols for 71 GO function

Usage
GOhuman

Format
genesets List containing gene symbols for 71 GO function (GO slim terms containing between
50 and 1,000 genes) downloaded from the Gene Ontology Consortium August 2015 http://
www.geneontology.org/page/download-annotations

Source
Dataset: https://github.com/mm-shah/MetaNeighbor/tree/master/data | Paper: https://
www.biorxiv.org/content/early/2017/06/16/150524
**makeClusterGraph**

*Description*

This representation is a useful alternative for heatmaps for large datasets and sparse AUROC matrices (MetaNeighborUS with one_vs_best = TRUE)

*Usage*

makeClusterGraph(best_hits, low_threshold = 0, high_threshold = 1)

*Arguments*

- **best_hits**: Matrix of AUROCs produced by MetaNeighborUS.
- **low_threshold**: AUROC threshold value. An edge is drawn between two clusters only if their similarity exceeds low_threshold.
- **high_threshold**: AUROC threshold value. An edge is drawn between two clusters only if their similarity is lower than high_threshold (enables focusing on close calls).

*Value*

A graph in igraph format, where nodes are clusters and edges are AUROC similarities.
**makeClusterName**

Make cluster names in format 'study_id|cell_type'

**Description**

Make cluster names in format 'study_id|cell_type'

**Usage**

```r
makeClusterName(study_id, cell_type)
```

**Arguments**

- `study_id` Character vector containing study ids.
- `cell_type` Character vector containing cell type names

**Value**

Character vector containing cluster names in the format `study_id|cell_type`.

---

**mergeSCE**

Merge multiple SingleCellExperiment objects.

**Description**

Merge multiple SingleCellExperiment objects.

**Usage**

```r
mergeSCE(sce_list)
```

**Arguments**

- `sce_list` A *named* list, where values are SingleCellExperiment objects and names are SingleCellExperiment objects.

**Value**

A SingleCellExperiment object containing the input datasets with the following limitations: (i) only genes common to all datasets are kept, (ii) only colData columns common to all datasets are kept, (iii) only assays common to all datasets (i.e., having the same name) are kept, (iv) all other slots (e.g., reducedDims or rowData) will be ignored and left empty. The SingleCellExperiment object contains a "study_id" column, mapping each cell to its original dataset (names in "sce_list").
MetaNeighbor

Description

For each gene set of interest, the function builds a network of rank correlations between all cells. Next, it builds a network of rank correlations between all cells for a gene set. Next, the neighbor voting predictor produces a weighted matrix of predicted labels by performing matrix multiplication between the network and the binary vector indicating cell type membership, then dividing each element by the null predictor (i.e., node degree). That is, each cell is given a score equal to the fraction of its neighbors (including itself), which are part of a given cell type. For cross-validation, we permute through all possible combinations of leave-one-dataset-out cross-validation, and we report how well we can recover cells of the same type as area under the receiver operator characteristic curve (AUROC). This is repeated for all folds of cross-validation, and the mean AUROC across folds is reported. Calls neighborVoting.

Usage

MetaNeighbor(
  dat,
  i = 1,
  experiment_labels,
  celltype_labels,
  genesets,
  bplot = TRUE,
  fast_version = FALSE,
  node_degree_normalization = TRUE,
  batch_size = 10,
  detailed_results = FALSE
)

Arguments

dat A SummarizedExperiment object containing gene-by-sample expression matrix.
i default value 1; non-zero index value of assay containing the matrix data
experiment_labels A vector that indicates the source/dataset of each sample.
celltype_labels A character vector or one-hot encoded matrix (cells x cell type) that indicates the cell type of each sample.
genomesets Gene sets of interest provided as a list of vectors.
bplot default true, beanplot is generated
fast_version default value FALSE; a boolean flag indicating whether to use the fast and low memory version of MetaNeighbor
node_degree_normalization
default value TRUE; a boolean flag indicating whether to normalize votes by dividing through total node degree.

batch_size
Optimization parameter. Gene sets are processed in groups of size batch_size. The count matrix is first subset to all genes from these groups, then to each gene set individually.

detailed_results
Should the function return the average AUROC across all test datasets (default) or a detailed table with the AUROC for each test dataset?

Value
A matrix of AUROC scores representing the mean for each gene set tested for each celltype is returned directly (see neighborVoting). If detailed_results is set to TRUE, the function returns a table of AUROC scores in each test dataset for each gene set.

See Also
neighborVoting

Examples

data("mn_data")
data("GOmouse")
library(SummarizedExperiment)
AUROC_scores = MetaNeighbor(dat = mn_data,
    experiment_labels = as.numeric(factor(mn_data$study_id)),
    celltype_labels = metadata(colData(mn_data))[["cell_labels"]],
    genesets = GOmouse,
    bplot = TRUE)

Description
When it is difficult to know how cell type labels compare across datasets this function helps users to make an educated guess about the overlaps without requiring in-depth knowledge of marker genes.

Usage
MetaNeighborUS(
    var_genes = c(),
    dat, 
    i = 1, 
    study_id, 
    cell_type, 
    trained_model = NULL,
)
fast_version = FALSE,
node_degree_normalization = TRUE,
one_vs_best = FALSE,
symmetric_output = TRUE
)

**Arguments**

- **var_genes**
  vector of high variance genes.

- **dat**
  SummarizedExperiment object containing gene-by-sample expression matrix.

- **i**
  default value 1; non-zero index value of assay containing the matrix data

- **study_id**
  a vector that lists the Study (dataset) ID for each sample

- **cell_type**
  a vector that lists the cell type of each sample

- **trained_model**
  default value NULL; a matrix containing a trained model generated from MetaNeighbor::trainModel. If not NULL, the trained model is treated as training data and dat is treated as testing data. If a trained model is provided, fast_version will automatically be set to TRUE and var_genes will be overridden with genes used to generate the trained_model

- **fast_version**
  default value FALSE; a boolean flag indicating whether to use the fast and low memory version of MetaNeighbor

- **node_degree_normalization**
  default value TRUE; a boolean flag indicating whether to use normalize votes by dividing through total node degree.

- **one_vs_best**
  default value FALSE; a boolean flag indicating whether to compute AUROCs based on a best match against second best match setting (default version is one-vs-rest). This option is currently only relevant when fast_version = TRUE.

- **symmetric_output**
  default value TRUE; a boolean flag indicating whether to average AUROCs in the output matrix.

**Value**

The output is a cell type-by-cell type mean AUROC matrix, which is built by treating each pair of cell types as testing and training data for MetaNeighbor, then taking the average AUROC for each pair (NB scores will not be identical because each test cell type is scored out of its own dataset, and the differential heterogeneity of datasets will influence scores). If symmetric_output is set to FALSE, the training cell types are displayed as columns and the test cell types are displayed as rows. If trained_model was provided, the output will be a cell type-by-cell type AUROC matrix with training cell types as columns and test cell types as rows (no swapping of test and train, no averaging).

**Examples**

```r
data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
celltype_NV = MetaNeighborUS(var_genes = var_genes,
  dat = mn_data,
  fast_version = FALSE,
  node_degree_normalization = TRUE,
  one_vs_best = FALSE,
  symmetric_output = TRUE)
```
\textit{mn\_data}

\begin{verbatim}
  study_id = mn_data$study_id,
  cell_type = mn_data$cell_type)
\end{verbatim}

\texttt{celltype\_NV}

\begin{verbatim}
  mn_data
\end{verbatim}

\textbf{Description}

A SummarizedExperiment object containing: a gene matrix, cell type labels, experiment labels, sets of genes, sample ID, study id and cell types.

\textbf{Usage}

\texttt{mn\_data}

\textbf{Format}

- **Gene matrix**  A gene-by-sample expression matrix consisting of 3157 rows (genes) and 1051 columns (cell types)
- **cell\_labels**  1051x1 binary matrix that indicates whether a cell belongs to the SstNos cell type (1=yes, 0 = no)
- **sample\_id**  A character vector of length 1051 that indicates the sample\_id of each sample
- **study\_id**  A character vector of length 1051 that indicates the study\_id of each sample (“GSE60361” = Zeisel et al, “GSE71585” = Tasic et al)
- **cell\_type**  A character vector of length 1051 that indicates the cell\_type of each sample

\textbf{Source}

Dataset: \url{https://github.com/mm-shah/MetaNeighbor/tree/master/data} 1. Zeisel et al. \url{http://science.sciencemag.org/content/347/6226/1138} 2. Tasic et al. \url{http://www.nature.com/neuro/journal/v19/n2/full/nn.4216.html}

\texttt{neighbor\_Voting} \textit{Runs the neighbor voting algorithm.}

\textbf{Description}

The function performs cell type identity prediction based on ‘guilt by association’ using cross validation. Performance is evaluated by calculating the AUROC for each cell type.
neighborVoting

Usage

neighborVoting(
  exp_labels,
  cell_labels,
  network,
  means = TRUE,
  node_degree_normalization = TRUE
)

Arguments

exp_labels A vector that indicates the dataset source of each sample

cell_labels sample by cell type matrix that indicates the cell type of each sample (0-absent; 1-present)

network sample by sample adjacency matrix, ranked and standardized between 0-1

means default TRUE, determines output formatting

node_degree_normalization default TRUE, should predictions be divided by node degree?

Value

If means = TRUE (default) a vector containing the mean of AUROC values across cross-validation folds will be returned. If FALSE a list is returned containing a cell type by dataset matrix of AUROC scores, for each fold of cross-validation. Default is over-ridden when more than one cell type is assessed.

See Also

MetaNeighbor

Examples

data("mn_data")
data("GOmouse")
library(SummarizedExperiment)
AUROC_scores = MetaNeighbor(dat = mn_data,
                           experiment_labels = as.numeric(factor(mn_data$study_id)),
                           celltype_labels = metadata(colData(mn_data))[["cell_labels"]],
                           genesets = GOmouse,
                           bplot = TRUE)

AUROC_scores
**orderCellTypes**

Order cell types based on AUROC similarity matrix.

**Description**

Order cell types based on AUROC similarity matrix.

**Usage**

```r
orderCellTypes(M, na_value = 0)
```

**Arguments**

- `M`: A square AUROC matrix as returned by MetaNeighborUS.
- `na_value`: Replace NA values with this value (default is 0).

**Value**

A hierarchical clustering object as returned by stats::hclust.

---

**plotBPlot**

Plot Bean Plot, showing how replicability of cell types depends on gene sets.

**Description**

Plot Bean Plot, showing how replicability of cell types depends on gene sets.

**Usage**

```r
plotBPlot(nv_mat, hvg_score = NULL, cex = 1)
```

**Arguments**

- `nv_mat`: A rectangular AUROC matrix as returned by MetaNeighbor, where each row is a gene set and each column is a cell type.
- `hvg_score`: Named vector with AUROCs obtained from a set of Highly Variable Genes (HVGs). The names must correspond to cell types from `nv_mat`. If specified, the HVG score is highlighted in red.
- `cex`: Size factor for row and column labels.
Examples

```r
data("mn_data")
data("GOmouse")
library(SummarizedExperiment)
AUROC_scores = MetaNeighbor(dat = mn_data,
                            experiment_labels = as.numeric(factor(mn_data$study_id)),
                            celltype_labels = metadata(colData(mn_data))[,"cell_labels"],
                            genesets = GOmouse,
                            bplot = FALSE)
plotBPlot(AUROC_scores)
```

---

**plotClusterGraph**  
*Plot cluster graph generated with makeClusterGraph.*

**Description**

In this visualization, edges are colored in black when AUROC > 0.5 and orange when AUROC < 0.5, edge width scales linearly with AUROC. Edges are oriented from training cluster towards test cluster. A black bidirectional edge indicates that two clusters are reciprocal top matches. Node radius reflects cluster size (small: up to 10 cells, medium: up to 100 cells, large: all other clusters).

**Usage**

```r
plotClusterGraph(
  graph,
  study_id = NULL,
  cell_type = NULL,
  size_factor = 1,
  label_cex = 0.2 * size_factor,
  legend_cex = 2,
  study_cols = NULL
)
```

**Arguments**

- **graph**: Graph in igraph format generated by `makeClusterGraph`.
- **study_id**: Vector with study IDs provided to `MetaNeighborUS` to compute AUROCs stored in graph (used to compute cluster size). If NULL, all nodes have medium size.
- **cell_type**: Vector with cell type labels provided to `MetaNeighborUS` to compute AUROCs stored in graph (used to compute cluster size). If NULL, all nodes have medium size.
- **size_factor**: Numeric value controlling the size of nodes and edges.
- **label_cex**: Numeric value controlling the size of cell type labels.
- **legend_cex**: Numeric value controlling the size of the legend.
- **study_cols**: Named vector where values are RGB colors and names are unique study identifiers. If NULL, a default color palette is used.
**plotDotPlot**

*Plot dot plot showing expression of a gene set across cell types.*

**Description**

The size of each dot reflects the number of cell that express a gene, the color reflects the average expression. Expression of genes is first average and scaled in each dataset independently. The final value is obtained by averaging across datasets.

**Usage**

```r
plotDotPlot(
  dat,
  experiment_labels,
  celltype_labels,
  gene_set,
  i = 1,
  normalize_library_size = TRUE,
  alpha_row = 10,
  average_expressing_only = FALSE
)
```

**Arguments**

- `dat`: A SummarizedExperiment object containing gene-by-sample expression matrix.
- `experiment_labels`: A vector that indicates the source/dataset of each sample.
- `celltype_labels`: A character vector that indicates the cell type of each sample.
- `gene_set`: Gene set vector that indicates the cell type of each sample.
- `i`: Default value 1; non-zero index value of assay containing the matrix data.
- `normalize_library_size`: Whether to apply library size normalization before computing average expression (set this value to FALSE if data are already normalized).
- `alpha_row`: Parameter controlling row ordering: a higher value of alpha_row gives more weight to extreme AUROC values (close to 1).
- `average_expressing_only`: Whether average expression should be computed based only on expressing cells (Seurat default) or taking into account zeros.

**Value**

A ggplot object.
**plotHeatmap**  
*Plots symmetric AUROC heatmap, clustering cell types by similarity.*

**Description**

Plots symmetric AUROC heatmap, clustering cell types by similarity.

**Usage**

```r
plotHeatmap(aurocs, cex = 1, margins = c(8, 8), ...)
```

**Arguments**

- `aurocs`: A square AUROC matrix as returned by `MetaNeighborUS`.
- `cex`: Size factor for row and column labels.
- `margins`: Size of margins (for row and column labels).
- `...`: Additional graphical parameters that are passed on to `gplots::heatmap.2` (allows customization of the heatmap).

**See Also**

- `ggPlotHeatmap`

**Examples**

```r
data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
celltype_NV = MetaNeighborUS(var_genes = var_genes,
                           dat = mn_data,
                           study_id = mn_data$study_id,
                           cell_type = mn_data$cell_type)
plotHeatmap(celltype_NV)
```

**plotHeatmapPretrained**  
*Plots rectangular AUROC heatmap, clustering train cell types (columns) by similarity, and ordering test cell types (rows) according to similarity to train cell types.*

**Description**

Plots rectangular AUROC heatmap, clustering train cell types (columns) by similarity, and ordering test cell types (rows) according to similarity to train cell types.
plotMetaClusters

Usage

plotHeatmapPretrained(
  aurocs,
  alpha_col = 1,
  alpha_row = 10,
  cex = 1,
  margins = c(8, 8)
)

Arguments

aurocs A rectangular AUROC matrix as returned by MetaNeighborUS.
alpha_col Parameter controlling column clustering: a higher value of alpha_col gives more weight to extreme AUROC values (close to 1).
alpha_row Parameter controlling row ordering: a higher value of alpha_row gives more weight to extreme AUROC values (close to 1).
cex Size factor for row and column labels.
margins Size of margins (for row and column labels).

Examples

data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
celltype_NV = MetaNeighborUS(var_genes = var_genes,
  dat = mn_data,
  study_id = mn_data$study_id,
  cell_type = mn_data$cell_type,
  symmetric_output = FALSE)
keep_col = getStudyId(colnames(celltype_NV)) == "GSE71585"
keep_row = getStudyId(rownames(celltype_NV)) != "GSE71585"
plotHeatmapPretrained(celltype_NV[keep_row, keep_col])

plotMetaClusters

Plot meta-cluster badges, each badge is a small AUROC heatmap restricted to a specific meta-cluster.

Description

Plot meta-cluster badges, each badge is a small AUROC heatmap restricted to a specific meta-cluster.
Usage

plotMetaClusters(
  meta_clusters,
  best_hits,
  reorder = FALSE,
  cex = 1,
  study_cols = NULL,
  auroc_breaks = c(0, 0.5, 0.7, 0.9, 0.95, 0.99, 1),
  auroc_cols = (grDevices::colorRampPalette(c("white", "blue")))(length(auroc_breaks) - 1)
)

Arguments

meta_clusters Meta-cluster list generated by extractMetaClusters.
best_hits Matrix of AUROCs used to extract meta-clusters.
reorder Reorder datasets by similarity for each badge? By default, the same dataset ordering is used for each badge.
cex Size factor controlling label size.
study_cols Named vector where values are RGB colors and names are unique study identifiers (corresponding to study_id). If NULL, a default color palette is used.
auroc_breaks Numeric vector used to bin AUROC values for color coding.
auroc_cols Vector containing RGB colors used to encode AUROC levels. The length of auroc_cols must correspond to the length of auroc_breaks - 1.

plotUpset

Plot Upset plot showing how replicability depends on input dataset.

Description

Plot Upset plot showing how replicability depends on input dataset.

Usage

plotUpset(metacusters, min_recurrence = 2, outlier_name = "outliers")

Arguments

metacusters Metaclusters extracted from MetaNeighborUS analysis.
min_recurrence Only show replicability structure for metaclusters that are replicable across at least min_recurrence datasets.
outlier_name In metaclusters, name assigned to outliers (clusters that did not match with any other cluster)
Examples

```r
data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
celltype_NV = MetaNeighborUS(var_genes = var_genes,
   dat = mn_data,
   study_id = mn_data$study_id,
   cell_type = mn_data$cell_type,
   fast_version = TRUE, one_vs_best = TRUE)
mclusters = extractMetaClusters(celltype_NV)
plotUpset(mclusters)
```

scoreMetaClusters

Summarize meta-cluster information in a table.

**Description**

Summarize meta-cluster information in a table.

**Usage**

```r
scoreMetaClusters(meta_clusters, best_hits, outlier_label = "outliers")
```

**Arguments**

- `meta_clusters`: Meta-cluster list generated by `extractMetaClusters`.
- `best_hits`: Matrix of AUROCs used to extract meta-clusters.
- `outlier_label`: Element of meta-cluster list containing outlier clusters.

**Value**

A data.frame. Column "meta_cluster" contains meta-cluster names, "clusters" lists the clusters belonging to each meta-cluster, "n_studies" is the number of studies spanned by the meta-cluster, "score" is the average similarity between meta-cluster members (average AUROC, NAs are treated as 0).
**splitClusters**  
*Split clusters according to symmetric AUROC similarity.*

**Description**

This function computes hierarchical clustering to group similar clusters, interpreting the AUROC matrix as a similarity matrix, then uses a standard tree cutting algorithm to obtain groups of similar clusters. Note that the cluster hierarchy corresponds exactly to the dendrogram shown when using the plotHeatmap function.

**Usage**

```r
splitClusters(mn_scores, k)
```

**Arguments**

- `mn_scores`: A symmetric AUROC matrix as generated by MetaNeighborUS.
- `k`: The number of desired cluster sets.

**Value**

A list of cluster sets, each cluster set is a character vector containing cluster labels.

**See Also**

- `plotHeatmap`

---

**splitTestClusters**  
*Split test clusters according to AUROC similarity to train clusters.*

**Description**

This function computes hierarchical clustering to group similar test clusters, using similarity to train clusters as features, then uses a standard tree cutting algorithm to obtain groups of similar clusters. Note that the cluster hierarchy does *not* correspond to the row ordering of plotHeatmapPretrained function, which uses a different heuristic.

**Usage**

```r
splitTestClusters(mn_scores, k)
```

**Arguments**

- `mn_scores`: An AUROC matrix as generated by MetaNeighborUS, usually with the "trained_model" option.
- `k`: The number of desired cluster sets.
splitTrainClusters

Value

A list of cluster sets, each cluster set is a character vector containing cluster labels.

See Also

plotHeatmapPretrained

splitTrainClusters Split train clusters according to AUROC similarity to test clusters.

Description

This function computes hierarchical clustering to group similar train clusters, using similarity to test clusters as features, then uses a standard tree cutting algorithm to obtain groups of similar clusters. Note that the cluster hierarchy corresponds exactly to the column dendrogram shown when using the plotHeatmapPretrained function.

Usage

splitTrainClusters(mn_scores, k)

Arguments

mn_scores An AUROC matrix as generated by MetaNeighborUS, usually with the "trained_model" option.
k The number of desired cluster sets.

Value

A list of cluster sets, each cluster set is a character vector containing cluster labels.

See Also

plotHeatmapPretrained
standardizeLabel

Remove special characters ("|") from labels to avoid later conflicts

Usage

standardizeLabel(labels, replace = "|", with = ".")

Arguments

labels Character vector containing study ids or cell type names.
replace Special character to replace
with Character to use instead of special character

Value

Character vector with replaced special characters.

subsetClusterGraph

Subset cluster graph to clusters of interest.

Usage

subsetClusterGraph(graph, vertices)

Arguments

graph Graph in igraph format generated by makeClusterGraph.
vertices Vector of cluster labels

Value

Graph in igraph format, where nodes have been restricted to clusters of interests.

See Also

extendClusterSet
Find reciprocal top hits

Description
Identifies reciprocal top hits and high scoring cell type pairs. This function only look for the overall top hit for each cell type. We strongly recommend using topHitsByStudy instead, which looks for top hits in each target study, providing a more comprehensive view of replicability.

Usage
topHits(cell_NV, dat, i = 1, study_id, cell_type, threshold = 0.95)

Arguments
- cell_NV: matrix of celltype-to-celltype AUROC scores (output from MetaNeighborUS)
- dat: a SummarizedExperiment object containing gene-by-sample expression matrix.
- i: default value 1; non-zero index value of assay containing the matrix data
- study_id: a vector that lists the Study (dataset) ID for each sample
- cell_type: a vector that lists the cell type of each sample
- threshold: default value 0.95. Must be between [0,1]

Value
Function returns a dataframe with cell types that are either reciprocal best matches, and/or those with AUROC values greater than or equal to threshold value

See Also
topHitsByStudy

Examples
data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
celltype_NV = MetaNeighborUS(var_genes = var_genes,
dat = mn_data,
study_id = mn_data$study_id,
cell_type = mn_data$cell_type)
top_hits = topHits(cell_NV = celltype_NV,
dat = mn_data,
study_id = mn_data$study_id,
cell_type = mn_data$cell_type,
threshold = 0.9)
top_hits
topHitsByStudy

Find reciprocal top hits, stratifying results by study.

Description
This function looks for reciprocal top hits in each target study separately, allowing for as many reciprocal top hits as target studies. This is the recommended function for extracting top hits.

Usage

topHitsByStudy(
    auroc,
    threshold = 0.9,
    n_digits = 2,
    collapse_duplicates = TRUE
)

Arguments

- **auroc**: matrix of celltype-to-celltype AUROC scores (output from `MetaNeighborUS`)
- **threshold**: AUROC threshold, must be between [0,1]. Default is 0.9. Only top hits above this threshold are included in the result table.
- **n_digits**: Number of digits for AUROC values in the result table. Set to "Inf" to skip rounding.
- **collapse_duplicates**: Collapse identical pairs of cell types (by default), effectively averaging AUROCs when reference and target roles are reversed. Setting this option to FALSE makes it easier to filter results by study or cell type. If collapse_duplicates is set to TRUE, "Celltype_1" is the reference cell type and "Celltype_2" is the target cell type (relevant if MetaNeighborUS was run with symmetric_output = FALSE).

Value
Function returns a dataframe with cell types that are either reciprocal best matches, and/or those with AUROC values greater than or equal to threshold value.

See Also
topHits

Examples

data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
aurocs = MetaNeighborUS(var_genes = var_genes, dat = mn_data,
trainModel

```r
study_id = mn_data$study_id,
cell_type = mn_data$cell_type
```
top_hits = topHitsByStudy(aurocs)
top_hits

---

**trainModel**

*Pretrains model for the unsupervised version of MetaNeighbor*

**Description**

When comparing clusters to a large reference dataset, this function summarizes the gene-by-cell matrix into a much smaller highly variable gene-by-cluster matrix which can be fed as training data into MetaNeighborUS, resulting in substantial time and memory savings.

**Usage**

```r
trainModel(var_genes, dat, i = 1, study_id, cell_type)
```

**Arguments**

- `var_genes`: vector of high variance genes.
- `dat`: SummarizedExperiment object containing gene-by-sample expression matrix.
- `i`: default value 1; non-zero index value of assay containing the matrix data
- `study_id`: a vector that lists the Study (dataset) ID for each sample
- `cell_type`: a vector that lists the cell type of each sample

**Value**

The output is a gene-by-cluster matrix that contains all the information necessary to run MetaNeighborUS from a pre-trained model.

**Examples**

```r
data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
trained_model = trainModel(var_genes = var_genes,
                           dat = mn_data,
                           study_id = mn_data$study_id,
                           cell_type = mn_data$cell_type)
celltype_NV = MetaNeighborUS(trained_model = trained_model,
                             dat = mn_data,
                             study_id = mn_data$study_id,
                             cell_type = mn_data$cell_type)
celltype_NV
```
variableGenes  

Identify a highly variable gene set

Description

Identifies genes with high variance compared to their median expression (top quartile) within each experiment.

Usage

variableGenes(
  dat,
  i = 1,
  exp_labels,
  min_recurrence = length(unique(exp_labels)),
  downsampling_size = 10000
)

Arguments

dat         SummarizedExperiment object containing gene-by-sample expression matrix.
i           default value 1; non-zero index value of assay containing the matrix data
exp_labels   character vector that denotes the source (Study ID) of each sample.
min_recurrence Number of studies across which a gene must be detected as highly variable to be kept. By default, only genes that are variable across all studies are kept (intersection).
downsampling_size Downsample each study to downsampling_size samples without replacement. If set to 0 or value exceeds dataset size, no downsampling is applied.

Value

The output is a vector of gene names that are highly variable in every experiment (intersect)

Examples

data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
var_genes
Index

* datasets
  GOhuman, 5
  GOmouse, 6
  mn_data, 11
extendClusterSet, 2, 22
extractMetaClusters, 3
ggetCellType, 4
ggetCellStudyId, 4
ggPlotHeatmap, 5, 16
  GOhuman, 5
  GOmouse, 6
makeClusterGraph, 6
makeClusterName, 7
mergeSCE, 7
MetaNeighbor, 8, 12
MetaNeighborUS, 9, 23, 24
mn_data, 11
neighborVoting, 8, 9, 11
orderCellTypes, 13
plotBPlot, 13
plotClusterGraph, 14
plotDotPlot, 15
plotHeatmap, 5, 16, 20
plotHeatmapPretrained, 16, 21
plotMetaClusters, 17
plotUpset, 18
scoreMetaClusters, 19
splitClusters, 20
splitTestClusters, 20
splitTrainClusters, 21
standardizeLabel, 22
subsetClusterGraph, 22
topHits, 23, 24
topHitsByStudy, 23, 24
trainModel, 25
variableGenes, 26