Package 'TockyConvNetR'

February 20, 2025

Version 0.1.0

Date 2025-02-11

Title Convolutional Neural Network-
Based Machine Learning Methods for Analyzing Flow Cytometric Fluorescent Timer Data

Author Masahiro Ono [aut, cre]

Maintainer Masahiro Ono <monotockylab@gmail.com>

Description
This package provides Convolutional Neural Network (CNN)-based machine learning methods for analysing flow cytometric data from Fluorescent Timer reporters. Specifically, the package provides random forest-based methods. It is optimised to Timer-of-Cell-Kinetics-and-Activity (Tocky), which enables the analysis of temporal dynamics of individual cells in vivo.

Depends R (>= 4.2.0), utils, stats, graphics, grDevices

Imports dplyr, reticulate, fields, abind, TockyPrep, ggplot2, rlang, tidyr, gridExtra

Suggests knitr, rmarkdown, KernSmooth

VignetteBuilder knitr

URL https://github.com/MonoTockyLab/TockyMachineLearning/TockyConvNetR

BugReports
https://github.com/MonoTockyLab/TockyMachineLearning/TockyConvNetR/issues

Encoding UTF-8

RoxygenNote 7.3.2

Contents

`convert_to_image` *Convert Data to Image Matrices*

Description

This function processes a dataset containing multiple subsets, each identified by a unique 'file' identifier, and converts each subset into a matrix based on binned 'Angle' and 'Intensity' values. It ensures consistent binning across all subsets by using global minimum and maximum values, facilitating direct comparison between the resulting matrices.

Usage

```
convert_to_image(data, n_resolution = 100)
```

Arguments

`data`	A data frame or list of data frames containing at least three columns: 'file' (unique identifier for each subset), 'Angle', and 'Intensity'. Each row represents an observation with a specific angle and intensity value.
`n_resolution`	The number of bins

Value

A list containing: - matrices_list: A list where each element is a data frame corresponding to a subset identified by 'file', containing the original data. - bin_edges_Angle: Numeric vector of bin edges used for the 'Angle' dimension. - bin_edges_Intensity: Numeric vector of bin edges used for the 'Intensity' dimension. - counts_list: A list of matrices where each matrix represents the binned counts of 'Angle' and 'Intensity' for a subset.

Examples

```
## Not run:
# Assume 'data' is your dataset containing 'file', 'Angle', and 'Intensity' columns
result <- convert_to_image(data)

# Access the binned counts matrix for the first file
first_counts_matrix <- result$counts_list[[1]]

# Plot the matrix as an image
image(result$bin_edges_Angle, result$bin_edges_Intensity, first_counts_matrix)

## End(Not run)
```

`getImages` *Get CNN Images from TockyPrepData Object*

Description

Retrieves CNN images data from a TockyPrepData object and prints diagnostic information about the image data structure.

Usage

```
getImages(x)
```

Arguments

`x`	A `TockyPrepData` object containing processed CNN images data. This object must have already undergone ImageConversion.

Details

This function performs the following actions:

- Checks if the `TockyCNNimages` slot contains data
- Prints diagnostic information including:
  - Dimensions of the image arrays
  - Variables used in the analysis
  - Sample definitions

Value

The CNN images data stored in the object's `TockyCNNimages` slot. The returned value maintains the original structure from the slot, typically including:

- Image arrays in the first element
- Sample definitions in the second element
- Variables used in the third element

Examples

```
## Not run:
# After successful ImageConversion:
images_data <- getCNNimages(tocky_prep_object)
dim(images_data[[1]])  # Access image array dimensions

## End(Not run)
```

ImageConversion *Image Conversion for Direct Export of Samples*

Description

Image Conversion for Direct Export of Samples

Usage

```
ImageConversion(x, n_resolution = 100, selected_markers = NULL, output = NULL)
```

Arguments

x	A TockyPrepData object
n_resolution	The number of bins to be used. The default is 100, i.e. the resolution of output images is 100 x 100.
selected_markers	A character vector with the length two for defining marker data to be converted into 2D images.
output	A character to specify the output directory.

Value

An updated TockyPrepData object. In addition, the function exports image data as numpy array files, which are to be used for TockyMLPy analysis and TockyCNN model construction.

Examples

```
## Not run:
x <- ImageConversion(x, n_resolution = 100)

## End(Not run)
```

inverseGradCAM *Integrated Grad-CAM Feature Cell Identification*

Description

Identifies significant feature cells using Grad-CAM data either with visualization (gating mode) or as a lightweight operation using pre-converted images (base mode).

Usage

```
inverseGradCAM(
  x = NULL,
  results = NULL,
  feature_matrix,
  mode = c("gating", "base"),
  percentile = 0.9,
  n_resolution = 100,
  transpose = TRUE,
  filename = NULL,
  ncol = 2,
  nrow = 2
)
```

Arguments

x	TockyPrepData object (required for "gating" mode)
results	convert_to_image output (required for "base" mode)
feature_matrix	Feature intensity matrix from Grad-CAM analysis
mode	Operation mode ("gating" for visualization, "base" for lightweight analysis)
percentile	Significance threshold percentile (0-1). If NULL, grad-CAM values will be returned, instead of feature cell designation.
n_resolution	Binning resolution (for "gating" mode only)
transpose	Logical Whether to tranpose feature_matrix input. Note that TockyConvNetPy output Grad-CAM matrix for feature_matrix typically needs to be transposed. The default is TRUE.
filename	Optional PDF output path (for "gating" mode only)
ncol	The number of the columns for the output plot
nrow	The number of the rows for the output plot

Value

A binary numeric vector (1/0) for feature and other cells. If 'percentile = NULL', grad-CAM values are returned as a numeric vector.

Examples

```
## Not run:
# Gating mode with visualization
out <- inverseGradCAM(mode = "gating", x = prep_data,
                          feature_matrix = cam_matrix, filename = "output.pdf")

# Base mode with pre-converted results
img_data <- convert_to_image(merged_data)
out <- inverseGradCAM(mode = "base", results = img_data,
                          feature_matrix = cam_matrix)

## End(Not run)
```

plotGradCAMFeatureCells *Generate Plots for Analysing Feature Cell Abundance*

Description

This function processes clustering results, plots each cluster, and overlays each cluster's convex hull. It is adaptable to any number of cell_cluster_id.

Usage

```
plotGradCAMFeatureCells(
  x,
  feature_cells,
  p_adjust_method = "BH",
  ncol = 3,
  min_cells = 10,
  title = "GradCAM Feature Cells",
  Timer_positive = TRUE,
  ylim = NULL
)
```

Arguments

x	TockyPrepData object (required for "gating" mode)
p_adjust_method	A method for p-value adjustment in multiple testing using Mann Whitney. clusteringFeatureCells cen be used.
ncol	Number of columns in output figure panel.
min_cells	Numeric. The minimum nunmber of cells within a cluster to be analysed. The default is 10.
Timer_positive	Logical. Whether to remove Timer negative cells.
ylim	Optional. A numeric vector of the length 2 for specifying ylim.

Examples

```
## Not run:
  data <- data.frame(Angle = runif(100), Intensity = runif(100))
  cell_cluster_id <- dbscan(data, eps = 0.1, minPts = 5)$cluster
  plotGradCAMFeatureCells(data, cell_cluster_id)

## End(Not run)
```

`plotGradCAMFeatureMFI` *Generate a boxplot of MFI (median fluorescence intensity) for Grad-CAM feature cells, other Timer+ cells, and Timer negative cells following inverseGradCAM.*

Description

Generate a boxplot of MFI (median fluorescence intensity) for Grad-CAM feature cells, other Timer+ cells, and Timer negative cells following inverseGradCAM.

Usage

```
plotGradCAMFeatureMFI(
  x,
  feature_vector,
  group = NULL,
  select = FALSE,
  variables = NULL,
  title = "GradCAM Feature Cells"
)
```

Arguments

`x`	A 'TockyPrepData' object containing the original flow cytometry data.
`feature_vector`	A vector output from inverseGradCAM.
`group`	A character vector specifying the group(s) to use for analysis. If NULL, all groups are used.
`select`	A logical indicating whether to allow interactive selection of variables for analysis.
`variables`	A character vector specifying the variables to analyze. Only used if 'select' is TRUE.

Value

A list containing the following elements: * 'plot': A ggplot object of the boxplot. * 'summary_data': A data frame containing the summarized data used to create the plot.

Examples

```
## Not run:
  feature_matrix <- read.csv('heatmap.csv')
plotGradCAMFeatureMFI(x, feature_matrix)

## End(Not run)
```

`plotInverseGradCAM` *Plot GradCAM (or Heatmap) Values in Original Flow Cytometry Plot*

Description

This function obtains GradCAM heatmap values for each cell in the original flow cytometric space, mapping the 'Angle' and 'Intensity' image dimensions (pixels) back to the corresponding cells.

Usage

```
plotInverseGradCAM(
  x,
  feature_matrix,
  xaxis = "Red_log",
  yaxis = "Blue_log",
  xlim = NULL,
  ylim = NULL,
  title = "GradCAM",
  color_bar = TRUE,
  n_resolution = 100,
  transpose = TRUE
)
```

Arguments

`x`	A 'TockyPrepData' object containing the original flow cytometry data.
`feature_matrix`	A 100 x 100 matrix representing the GradCAM output.
`xaxis`	The name of the dataframe column to be used as x-axis in the plot (default 'Red_log').
`yaxis`	The name of the dataframe column to be used as y-axis in the plot (default 'Blue_log').
`xlim`	Optional vector of length 2 defining the x-axis limits.
`ylim`	Optional vector of length 2 defining the y-axis limits.
`title`	Character string specifying the title of the plot.
`color_bar`	Logical indicating whether to include a color bar in the plot (default TRUE).
`n_resolution`	Binning resolution. The default is 100.
`transpose`	Logical Whether to tranpose feature_matrix input. Note that TockyConvNetPy output Grad-CAM matrix for feature_matrix typically needs to be transposed. The default is TRUE.

Value

Invisibly returns the unmodified 'TockyPrepData' object.

Examples

```
## Not run:
  feature_matrix <- read.csv('heatmap.csv')
  par(mfrow= c(1,2))
  plotInverseGradCAM(x, feature_matrix, color_bar = TRUE)

## End(Not run)
```

`writeImages` *Export TockyCNN Image Data from TockyPrepData Object*

Description

Export CNN images data from a TockyPrepData object and prints diagnostic information about the image data structure.

Usage

```
writeImages(x, numpy = TRUE)
```

Arguments

`x`	A `TockyPrepData` object containing processed CNN images data. This object must have already undergone ImageConversion.
`numpy`	Logical. Whether to export your data in the numpy format. If FALSE, csv files are exported instead.

Details

This function performs the following actions:

- Checks if the `TockyCNNimages` slot contains data
- Prints diagnostic information including:
  - Dimensions of the image arrays
  - Variables used in the analysis
  - Sample definitions

Value

The CNN images data stored in the object's `TockyCNNimages` slot. The returned value maintains the original structure from the slot, typically including:

- Image arrays in the first element
- Sample definitions in the second element
- Variables used in the third element

Examples

```
## Not run:
# After successful ImageConversion:
images_data <- getCNNimages(tocky_prep_object)
dim(images_data[[1]])  # Access image array dimensions

## End(Not run)
```

Index