Package: drugsens (via r-universe)

March 28, 2025

Title Automated Analysis of 'QuPath' Output Data and Metadata Extraction

Description A comprehensive toolkit for analyzing microscopy data output from 'QuPath' software. Provides functionality for automated data processing, metadata extraction, and statistical analysis of imaging results. The methodology implemented in this package is based on Labrosse et al. (2024) <doi:10.1016/j.xpro.2024.103274> ``Protocol for quantifying drug sensitivity in 3D patient-derived ovarian cancer models'', which describes the complete workflow for drug sensitivity analysis in patient-derived cancer models.

Version 0.1.0

BugReports `https://git.scicore.unibas.ch/ovca-research/drugsens/-/issues`

SystemRequirements QuPath™ 4.0.0 or higher

URL `https://git.scicore.unibas.ch/ovca-research/drugsens/`

Maintainer Flavio Lombardo <`flavio.lombardo@unibas.ch`>

License MIT + file LICENSE

Imports dplyr, ggplot2, ggpubr, knitr, roxygen2, stats, stringr, tidyr, tidyselect, utils, testthat (>= 3.0.0)

Depends R (>= 4.2)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/testthat/edition 3

Config/pak/sysreqs cmake make libicu-dev libxml2-dev

Repository https://flalom.r-universe.dev

RemoteUrl https://github.com/flalom/drugsens

RemoteRef HEAD

RemoteSha 46ea2fb92f8591331249436f7b34427f99f21b52

Contents

`change_data_format_to_longer` *Data format changer*

Description

This function gets the count data data.frame, that has a wider format and it returns a longer-formatted data.frame

Usage

```
change_data_format_to_longer(
  .data,
  pattern_column_markers = "_ratio_of_total_cells",
  unique_name_row_identifier = "filter_image",
  additional_columns = TRUE
)
```

Arguments

`.data`	The markers count dataframe that is coming from the processing of the microscopy data
`pattern_column_markers`	The markers' pattern name to obtain the column with ratios of the markers (it defaults to "_ratio_of_total_cells")
`unique_name_row_identifier`	String that indicates the unique identifier for each image, defaults as "filter_image"
`additional_columns`	columns that can be additionally added to the longer formatted data.frame, "Defaults as c("Treatment", "PID", "Image_number", "Tissue", "Concentration", "DOC")"

Details

Reformat the counts data in longer format

Value

A `dataframe/tibble`.

Examples

```
# Set up relabeling list
list_of_relabeling <- list(
  "PathCellObject" = "onlyDAPIPositve",
  "cCasp3" = "cCASP3",
  "E-Cadherin: cCASP3" = "E-Cadherin and cCASP3",
  "EpCAM_E-Cadherin" = "E-Cadherin",
  "EpCAM_E-Cadherin and cCASP3" = "E-Cadherin and cCASP3"
)

# Load and process example data
bind_data <- data_binding(
  path_to_the_projects_folder = system.file("extdata/to_merge/", package = "drugsens")
)
counts_dataframe <- make_count_dataframe(bind_data)

# Convert to long format
plotting_ready_dataframe <- change_data_format_to_longer(counts_dataframe)
```

`data_binding` *Merge all the dataframes coming out from the QuPath*

Description

This function identifies string patterns in the dataset, fills the dataframe with that information, and combines all data into a single file

Usage

```
data_binding(
  path_to_the_projects_folder,
  files_extension_to_look_for = "csv",
  recursive_search = FALSE,
  forcePath = NULL
)
```

Arguments

`path_to_the_projects_folder`	String/Path The path where the files coming out of QuPath are located
`files_extension_to_look_for`	String The extension of the file outputted from QuPath, (default is "csv")

`recursive_search`	Boolean, it defined the behavior of the file search, if recursive or not, (default is FALSE)
`forcePath`	String defining an alternative path to the confic file

Value

A concatenated dataframe from all the files within the indicated path

Examples

```
## Not run:
bind_data <- data_binding(path_to_the_projects_folder = system.file("extdata/to_merge/",
                          package = "drugsens"))

## End(Not run)
```

`generate_qupath_script` *Generate the groovy script used for the analysis*

Description

Generate a useful script to consistently save the output data from QuPath in .csv format following the naming conventions followed during the package development.

Usage

```
generate_qupath_script(output_dir = NULL)
```

Arguments

`output_dir`	Directory where the script should be saved. If NULL, uses tempdir()

Value

Invisibly returns the path to the generated script file.

Examples

```
## Not run:
  # Generate script in a temporary directory
  generate_qupath_script()

  # Generate script in a specific directory
  output_dir <- tempdir()
  generate_qupath_script(output_dir = output_dir)

## End(Not run)
```

`get_QC_plots` *Plot some QC plots to define that everything ran correctly*

Description

Plot data to visualize immediate trends. This function expects data that has been processed through make_count_dataframe() and change_data_format_to_longer() to ensure the correct data structure for plotting.

Usage

```
get_QC_plots(
  .data,
  patient_column_name = "PID",
  colors = c("darkgreen", "red", "orange", "pink"),
  save_plots = FALSE,
  folder_name = NULL,
  x_plot_var = "Treatment_complete",
  isolate_a_specific_patient = NULL
)
```

Arguments

`.data`	The preprocessed data (after running make_count_dataframe() and change_data_format_to_longer()) merged data.frame that should be visualized
`patient_column_name`	The PID's column name in the merged data.frame (defaults to "PID")
`colors`	A list of colors to supply to personalize the plot, defaults to c("darkgreen", "red", "orange", "pink")
`save_plots`	A Boolean value indicating if the plots should be saved or not (default is FALSE)
`folder_name`	A string indicating the name of the folder where to save the plots if save_plots is TRUE
`x_plot_var`	A string indicating the treatment's full name for the QC plots (default is "Treatment_complete")
`isolate_a_specific_patient`	A string indicating the patient name to isolate for single plot case (default is NULL)

Value

Invisibly returns NULL, but saves plots to disk if save_plots is TRUE

Examples

```
# First process example data
example_path <- system.file("extdata/to_merge/", package = "drugsens")
raw_data <- data_binding(path_to_the_projects_folder = example_path)
count_data <- make_count_dataframe(raw_data)
processed_data <- change_data_format_to_longer(count_data)

# Create and save plots to temporary directory
temp_dir <- file.path(tempdir(), "qc_plots")
get_QC_plots(
  processed_data,
  save_plots = TRUE,
  folder_name = temp_dir
)

# Create plots for a specific patient
get_QC_plots(
  processed_data,
  isolate_a_specific_patient = "B39",
  save_plots = TRUE,
  folder_name = temp_dir
)
```

`get_QC_plots_parsed_merged_data`

Plot QC plots and calculate statistics for bound data

Description

This function creates quality control plots and calculates basic statistics for microscopy data. The plots provide visual insights into marker expression patterns and data quality.

Usage

```
get_QC_plots_parsed_merged_data(
  .data,
  list_of_columns_to_plot = NULL,
  save_plots = FALSE,
  saving_plots_folder = NULL,
  save_plots_in_patient_specific_subfolders = TRUE,
  fill_color_variable = NULL,
  PID_column_name = "PID",
  isolate_specific_drug = NULL,
  isolate_specific_patient = NULL,
  drug_column_name = "Treatment",
  save_list_of_plots = TRUE,
  p_height = 10,
```

```
  p_width = 10,
  verbose = TRUE
)
```

Arguments

`.data`	The preprocessed data frame to analyze
`list_of_columns_to_plot`	Columns to include in plots. If NULL, all numeric columns are used.
`save_plots`	Logical, whether to save plots to files. Defaults to FALSE.
`saving_plots_folder`	Directory for saving plots. If NULL and save_plots=TRUE, uses a subdirectory of tempdir().
`save_plots_in_patient_specific_subfolders`	Logical, whether to create patient subdirectories. Defaults to TRUE.
`fill_color_variable`	Variable name for plot color filling
`PID_column_name`	Column name for patient IDs. Defaults to "PID".
`isolate_specific_drug`	Drug name to subset data
`isolate_specific_patient`	Patient ID to subset data
`drug_column_name`	Column name for drug information. Defaults to "Treatment".
`save_list_of_plots`	Logical, whether to return list of plot objects. Defaults to TRUE.
`p_height`	Plot height in inches. Defaults to 10.
`p_width`	Plot width in inches. Defaults to 10.
`verbose`	Logical, whether to show progress messages. Defaults to TRUE.

Value

If save_list_of_plots=TRUE, returns a named list of ggplot objects. Otherwise returns invisible(NULL).

Examples

```
# First load and process example data
example_path <- system.file("extdata/to_merge/", package = "drugsens")
raw_data <- data_binding(path_to_the_projects_folder = example_path)
count_data <- make_count_dataframe(raw_data)
processed_data <- change_data_format_to_longer(count_data)

# Basic usage - create plots for all patients
plots <- get_QC_plots_parsed_merged_data(processed_data)
```

```
# Save plots to a temporary directory
temp_dir <- file.path(tempdir(), "qc_plots")
plots <- get_QC_plots_parsed_merged_data(
  processed_data,
  save_plots = TRUE,
  saving_plots_folder = temp_dir
)

# Focus on a specific patient
plots <- get_QC_plots_parsed_merged_data(
  processed_data,
  isolate_specific_patient = "B39"
)

# Color plots by tissue type
plots <- get_QC_plots_parsed_merged_data(
  processed_data,
  fill_color_variable = "Tissue"
)
```

`make_count_dataframe` *Count the main marker expression*

Description

This function counts every single marker present in the "Name" column of the data.frame and return a dataframe of the counts per marker

Usage

```
make_count_dataframe(
  .data,
  unique_name_row_identifier = "filter_image",
  name_of_the_markers_column = "Name"
)
```

Arguments

`.data`	The dataframe that is coming from the processing of the microscopy data
`unique_name_row_identifier`	The name of the column of the .data where the unique name can be used to counts (it defaults to "filter_image")
`name_of_the_markers_column`	The name of the column of the .data where the marker names are expressed (ie E-Caderin, DAPI), "Defaults as Name"

Value

A `dataframe/tibble`.

Examples

```
# First load example data
pkg_path <- system.file("extdata/to_merge/", package = "drugsens")
bind_data <- data_binding(
  path_to_the_projects_folder = pkg_path,
  files_extension_to_look_for = "csv"
)

# Process the data
counts_dataframe <- make_count_dataframe(bind_data)

# Convert to plotting format
plotting_ready_dataframe <- change_data_format_to_longer(counts_dataframe)



# Example with custom parameters
make_count_dataframe(
  bind_data,
  name_of_the_markers_column = "Name",
  unique_name_row_identifier = "filter_image"
)
```

`make_run_config` *Generates and use a config txt file*

Description

When this function run the first time, it will generated a config.txt file in the user working directory. It will import the data config file into the use environment. This data will be used to change the column names of the imported dataset and change the name of the markers that is often incorrectly exported.

Usage

```
make_run_config(overwrite_config = FALSE, forcePath = NULL)
```

Arguments

`overwrite_config`	Boolean, if TRUE the `config_drugsens.txt` will be overwritten (default is FALSE)
`forcePath`	String, Define a custom path for the config file

Value

A `dataframe/tibble`.

Examples

```
# Generate config in temporary directory
make_run_config(forcePath = tempdir())
```

`string_parsing` *Parse image filenames to extract metadata*

Description

This function will parse the data from the Image name and will return the metadata there contained The metadata will be then associated to the count file as well

Usage

```
string_parsing(.data)
```

Arguments

`.data`	dataframe with parsed metadata

Value

A `dataframe/tibble`.

Examples

```
# Basic example with sample data
input_data <- data.frame(
  Image = "B516_Ascites_2023-11-25_DOC2020-12-14_dmso_rep_Ecad_cCasp3_(series 01).tif"
)
test <- drugsens::string_parsing(input_data)


# Example with actual data processing
data.parsed <- string_parsing(input_data)
```

Index