GEO Web-based Technology Identifier

Identifying Transcriptomics Technology Types (Expression Array, Bulk RNA-seq, and Single-cell RNA-seq) from GEO Webpages using GEO Accession Numbers

Mohammad Reza Mohajeri

2023-8-25

Introduction

- Identifying the technology type employed in genomics datasets is pivotal for ensuring accurate and appropriate downstream analysis.

- This automation aids researchers in swiftly and efficiently classifying datasets and ensuring compatibility with their analytical requirements.

1. Technology Identification for a Single GEO Accession Number

(- NOTE This code is a working prototype and might require further modifications for enhanced accuracy.)

- Objective
Quickly discern the technology type (e.g., Single-cell RNA-seq, Bulk RNA-seq, Expression Array) of a GEO dataset using a given accession number.

- Workflow
- Fetches dataset description from the GEO website.
- Analyzes content for specific keywords indicating dataset technology.
- Classifies the dataset based on detected keywords.
- Returns a classification statement, listing reasons (detected keywords).

How the code works

1. Packages Used

- stringi for string/text processing
- rvest to scrape (or harvest) data and the content of the GEO webpage (web pages)
- dplyr for data manipulation.

2. Function (get dataset type)

URL Construction
- Constructs the GEO URL for the provided accession number.

Page Fetching

Retrieves the webpage content.
Text Extraction
- Grabs the main text content from the page and converts it to lowercase for case-insensitive matching.

Keyword Detection
- Uses string matching and regular expressions to search for specific keywords indicative of the dataset type.

Classification
- Based on the detected keywords, classifies the dataset and returns the result.

3. Classification Logic
- Initially checks for single-cell keywords. If matched, classifies as Single-cell RNA-seq.
- Checks for combined presence of array and expression for Expression Array classification.
- Otherwise, seeks several keywords related to Bulk RNA-seq to make its classification.
- If no relevant keywords are found, it's labeled as Other/Unknown.

4. Execution
- The function is then called with a sample GEO accession number (GSE48216), but users can replace this with any other valid accession number.

```r
# Load necessary libraries
library(stringi)
library(rvest)
library(dplyr)
```

```r
get_dataset_type <- function(accession_number) {
  # Construct the URL
  url <- paste0("https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=", accession_number)

  # Fetch the page content (rvest)
  page <- read_html(url)

  # Extract the text content from the page (rvest and dplyr)
  page_text <- page %>% html_text()

  # Convert to lower case for case-insensitive matching (base R)
  page_text_lower <- tolower(page_text)

  # Initialize an empty vector to hold matching keywords (base R)
```

```
  matching_keywords <- c()

  # Priority check for single-cell keywords (stringi)
  if (stri_detect_regex(page_text_lower, "single[- ]?cell") ||
      stri_detect_regex(page_text_lower, "sc[- ]?rna[- ]?seq")) {

    matching_keywords <- c("Single-cell RNA-seq")
    return(paste("The dataset", accession_number, "is of type: Single-cell RNA-seq, because of keywords
  }

  # Check for the keywords to determine the type of dataset
  if (stri_detect_fixed(page_text_lower, "array") &&
      stri_detect_fixed(page_text_lower, "expression")) {
    matching_keywords <- c(matching_keywords, "Array", "Expression")
    return(paste("The dataset", accession_number, "is of type: Expression Array, because of keywords:",
  }

  if (stri_detect_fixed(page_text_lower, "rna-seq")) {
    matching_keywords <- c(matching_keywords, "RNA-Seq")
  }

  if (stri_detect_fixed(page_text_lower, "rna sequencing")) {
    matching_keywords <- c(matching_keywords, "RNA Sequencing")
  }

  if (stri_detect_fixed(page_text_lower, "expression profiling") &&
      stri_detect_fixed(page_text_lower, "high throughput") &&
      stri_detect_fixed(page_text_lower, "illumina")) {
    matching_keywords <- c(matching_keywords, "Expression profiling", "High throughput", "Illumina")
  }

  if (stri_detect_fixed(page_text_lower, "bulkrnaseq")) {
    matching_keywords <- c(matching_keywords, "BulkRnaseq")
  }

  if (length(matching_keywords) > 0) {
    return(paste("The dataset", accession_number, "is of type: Bulk RNA-seq, because of keywords:", pas
  } else {
    return(paste("The dataset", accession_number, "is of type: Other/Unknown"))
  }
}

# Test the function with a GEO accession number
result <- get_dataset_type("GSE159282")  # Replace this with any GEO accession number
print(result)
```

```
## [1] "The dataset GSE159282 is of type: Single-cell RNA-seq, because of keywords: Single-cell RNA-seq
```

2. Technology Identification for Multiple GEO Accession Numbers from a TSV File

(- NOTE This code is a working prototype and might require further modifications for enhanced accuracy.)

- In high-throughput genomics studies, researchers often deal with multiple datasets.
- Automatically classifying multiple datasets saves time and ensures consistent criteria application.
- This script automates the dataset TECHNOLOGY TYPE (e.g. Expression Array, Bulk RNA-seq, or Single-cell RNA-seq) identification for a list of GEO accession numbers provided in a .tsv file.

How the code works

1. Packages Used
- stringi for string/text processing.
- rvest to scrape (or harvest) data and the content of the GEO webpage (web pages)
- dplyr for data manipulation.
- readr to read data from .tsv files.
- writexl to save results back to a .tsv file.

2. Function (get dataset type)
- This function works similarly to the one in the previous script but returns results in a slightly different format suitable for aggregation.
- Specifically, for each accession number, it provides both the dataset type and the reasoning (matching keywords).

3. Data Reading
- Reads a list of dataset IDs from a .tsv file. The path is specified and can be modified to point to the correct file location.

4. Classification Logic
- Iteratively classifies each dataset ID in the provided list using the get dataset type function.

5. Result Aggregation

- All results are aggregated in a data frame, with columns Dataset ID, Type, and Reason.

6. Data Writing

- Saves the aggregated results to a new .tsv file at a specified path.

7. Note on Execution

- Ensure the input file path correctly points to the .tsv file containing the list of GEO accession numbers. Likewise, set the desired path for the output file to save the results.

```r
# Load necessary libraries
library(stringi)
library(rvest)
library(dplyr)
library(readr)
library(writexl)
```

```r
 # options(timeout=300)  # sets timeout to 10 minutes

get_dataset_type <- function(accession_number) {
  url <- paste0("https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=", accession_number)
  page <- read_html(url)
  page_text <- page %>% html_text()
  page_text_lower <- tolower(page_text)
  matching_keywords <- c()

  if (stri_detect_regex(page_text_lower, "single[- ]?cell") ||
      stri_detect_regex(page_text_lower, "sc[- ]?rna[- ]?seq")) {
    matching_keywords <- c("Single-cell RNA-seq")
    return(c("Single cell RNA Seq", paste("Keywords:", paste(matching_keywords, collapse=", "))))
  }

  if (stri_detect_fixed(page_text_lower, "array") &&
      stri_detect_fixed(page_text_lower, "expression")) {
    matching_keywords <- c(matching_keywords, "Array", "Expression")
    return(c("Expression array", paste("Keywords:", paste(matching_keywords, collapse=", "))))
  }

  if (stri_detect_fixed(page_text_lower, "rna-seq")) {
    matching_keywords <- c(matching_keywords, "RNA-Seq")
  }

  if (stri_detect_fixed(page_text_lower, "rna sequencing")) {
    matching_keywords <- c(matching_keywords, "RNA Sequencing")
```

```
  }

  if (stri_detect_fixed(page_text_lower, "expression profiling") &&
      stri_detect_fixed(page_text_lower, "high throughput") &&
      stri_detect_fixed(page_text_lower, "illumina")) {
    matching_keywords <- c(matching_keywords, "Expression profiling", "High throughput", "Illumina")
  }

  if (length(matching_keywords) > 0) {
    return(c("Bulk RNA Seq", paste("Keywords:", paste(matching_keywords, collapse=", "))))
  } else {
    return(c("Other", "N/A"))
  }
}

# Read the list of dataset IDs from the TSV file
dataset_ids <- read_tsv("/home/omidmoh1980/Desktop/SLE_Dataset_IDs.tsv")$Dataset_ID
```

```
## Rows: 125 Columns: 1
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (1): <?xml version="1.0" encoding="UTF-8"?>
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

## Warning: Unknown or uninitialised column: `Dataset_ID`.
```

```
# Initialize an empty data frame to store the results
result_df <- data.frame(Dataset_ID=character(),
                        Type=character(),
                        Reason=character(),
                        stringsAsFactors=FALSE)

# Loop through each dataset ID and get its type and reason
for(id in dataset_ids) {


  Sys.sleep(5)  # Add a 5-second delay here

  type_and_reason <- get_dataset_type(id)
  result_df <- rbind(result_df, data.frame(Dataset_ID=id,
                                           Type=type_and_reason[1],
                                           Reason=type_and_reason[2],
                                           stringsAsFactors=FALSE))
}

# Save the result to a new TSV file
write_tsv(result_df, "/mnt/data/9-ClarivateTranscriptomicBioinformatician/Tips/GEO_Technology_Type_Resul
```