Package 'MetaboMapper'

April 1, 2025

Type Package

Title Metabolite Label Harmonization Tool

Version 0.0.1

Author person(``Mariam", ``Ait Oumelloul", email = ``mariam.aitoumelloul@epfl.ch", role = c(``cre",``aut")), person(``Tancredi", ``Guido Cogne", role = ``ctb"))

Maintainer The package maintainer <mariam.aitoumelloul@epfl.ch>

Description
A package designed for metabolite label matching, offering functions to map names or identifiers (e.g., HMDB, KEGG, CHEBI, INCHIKEY) to standardized HMDB identifiers. This facilitates the integration of datasets from diverse studies with varying annotations.

License MIT + file LICENSE

URL https://github.com/mariamaitoumelloul/MetaboMapper

BugReports https://github.com/mariamaitoumelloul/MetaboMapper/issues

Depends R (>= 4.3.0)

Imports dplyr, tidyr, xml2, pbapply, rlang, stats

Suggests pkgdown, testthat (>= 3.0.0)

Encoding UTF-8

RoxygenNote 7.3.2

Config/testthat/edition 3

R topics documented:

`clean_hmdb_data` *Clean and Process HMDB Data*

Description

Cleans and prepare the HMDB dataset by converting all columns to character, replacing NAs with empty strings, removing unnecessary characters, and creating needed columns for the next steps.

Usage

```
clean_hmdb_data(df_hmdb)
```

Arguments

`df_hmdb`	Dataframe generated by the first steps of hmdbextract function

Value

Processed dataframe for the next steps.

`extract_highest_star` *Extract Highest Star*

Description

Identifies and extracts the substring(s) with the highest number of trailing asterisks (‘*‘) from a delimited input string.

Usage

```
extract_highest_star(s)
```

Arguments

`s`	A string where items are delimited by ‘//‘, each possibly ending with one or more trailing asterisks (‘*‘).

Details

- The input string is split into individual items based on the ‘//‘ delimiter. - Each item is checked for trailing asterisks using a regular expression. - The function identifies the item(s) with the maximum number of trailing asterisks. - If multiple items tie for the highest count, they are concatenated using ‘/‘.

Value

A single string containing the substring(s) with the highest number of trailing asterisks, separated by ‘/‘ if there are ties.

`final_label_dataframe` *Generate a Final Labeled Dataframe*

Description

This function processes an input dataset by combining and selecting relevant columns (identifiers with highest confidence levels)

Usage

```
final_label_dataframe(
  data_input,
  study_cols,
  details = FALSE,
  original_colname
)
```

Arguments

`data_input`	A data frame containing input data with columns for accession and name information.
`study_cols`	A column name or a vector of column names in ‘df_study‘ corresponding to identifiers to be mapped.
`details`	Logical, indicating whether to include all details in the output. If ‘TRUE‘, the entire dataframe is returned with renamed columns for clarity. If ‘FALSE‘, only selected columns are returned.
`original_colname`	the name of the column containing the original metabolites label

Details

The function performs the following steps: 1. Identifies columns starting with ‘"accession_in_reference"‘ and ‘"name_in_reference"‘. 2. Creates new columns ‘HMDB_id_list‘ and ‘HMDB_name_list‘ by combining unique, non-NA values from the identified columns. 3. Extracts the highest confidence level (based on the number of stars) for ‘HMDB_id‘ and ‘HMDB_label‘ using the ‘extract_highest_star‘ function. 4. Calculates the number of stars (‘Num_Stars‘) for the label and maps star counts to corresponding confidence levels. 5. Depending on the ‘details‘ parameter, either the full processed dataframe or a subset with selected columns is returned.

Value

A data frame with the final processed results: - If ‘details = TRUE‘, all processed columns are returned with renamed columns for clarity. - If ‘details = FALSE‘, a subset with columns for the original study name, HMDB label, HMDB identifier, and confidence level is returned.

`hmdbextract` *Extract metabolite information from HMDB XML File*

Description

Parses an HMDB XML file to extract metabolite-related information, organizing it into a structured data frame that will be used as the HMDB reference for the next steps.

Usage

```
hmdbextract(in_file)
```

Arguments

`in_file`	A character string specifying the path to the HMDB XML file.

Details

This function parses an HMDB XML file (XML can be downloaded from the "Downloads" section here :https://hmdb.ca). It iterates through each metabolite node, extracts relevant fields, and returns the information as a data frame. The output will be used for the next steps as the reference.

Value

A data frame containing extracted metabolite information with the following columns:

- `accession`: Primary accession ID.
- `secondary_accessions`: Secondary accession IDs, concatenated into a single string.
- `name`: Name of the metabolite.
- `description`: Description of the metabolite.
- `synonyms`: Synonyms for the metabolite, concatenated into a single string.
- `InChIKey`: InChIKey of the metabolite.
- `description_taxonomy`: Description of the taxonomy.
- `kingdom`: Taxonomic kingdom classification.
- `super_class`: Taxonomic super class.
- `class`: Taxonomic class.
- `sub_class`: Taxonomic subclass.
- `direct_parent`: Taxonomic direct parent classification.
- `chebi_id`: ChEBI ID of the metabolite.
- `kegg_id`: KEGG ID of the metabolite.

Examples

```
## Not run:
  # Example usage
  extracted_data <- hmdbextract("path_to_hmdb.xml")
  head(extracted_data)

## End(Not run)
```

mapper_confidence_level *Mapper Confidence Level*

Description

Based on which the mapping strategy, give a confidence level for the metabolite annotation

Usage

```
mapper_confidence_level(data)
```

Arguments

data	A data frame containing metabolite identifiers to be updated with confidence score.

Details

- Columns ending with '_HMDB', '_inchikey', '_NAME', '_KEGG', or '_CHEBI' are appended with '***'. - Columns ending with '_skeleton' are appended with '**'. - Non-character columns matching the patterns are skipped with a warning.

modified_df <- mapper_confidence_level(df)

Value

A modified data frame where selected identifier columns with corresponding confidence level.

map_unique_identifier *Map Unique Identifiers*

Description

Maps a given metabolite identifier type in a cohort data frame to reference identifiers (e.g., INCHIKEY, NAME, HMDB, CHEBI, KEGG) by performing a series of processing, matching, and reporting steps.

Usage

```
map_unique_identifier(df_cohort, col_cohort, df_reference, identifier)
```

Arguments

df_cohort	A data frame containing cohort data with metabolite identifiers to be mapped.
col_cohort	A character string specifying the column in 'df_cohort' containing the identifier to be mapped.
df_reference	A data frame containing reference data with metabolite identifiers and corresponding metadata.
identifier	A character string specifying the type of identifier to map. Supported values are '"INCHIKEY"', '"NAME"', '"HMDB"', '"CHEBI"', and '"KEGG"'.

Details

- For ‘INCHIKEY‘, both the full and skeleton INCHIKEY identifiers are mapped. - For ‘NAME‘, metabolite names are processed and matched to all known labels, inclusing synonyms. - For ‘HMDB‘, ‘CHEBI‘, and ‘KEGG‘, the corresponding identifiers are matched directly to reference identifiers.

Value

A data frame with updated mapping information, including matched identifiers

---

`match_identifier` *Match Identifier*

Description

Matches metabolite identifiers from a cohort data set with a reference database, annotating the matches and providing additional information.

Usage

```
match_identifier(
  df_cohort,
  col_cohort,
  df_reference,
  col_reference,
  identifier
)
```

Arguments

`df_cohort`	A data frame containing cohort data with metabolite identifiers to be mapped.
`col_cohort`	A character string specifying the column in ‘df_cohort‘ containing the identifier to be mapped.
`df_reference`	A data frame containing reference data with metabolite identifiers and corresponding metadata.
`identifier`	A character string specifying the type of identifier to map. Supported values are ‘"INCHIKEY"‘, ‘"NAME"‘, ‘"HMDB"‘, ‘"CHEBI"‘, and ‘"KEGG"‘.

Details

- Counts and prints the number of available identifiers in the cohort data set before matching. - Renames columns in the reference data frame by appending ‘"_in_reference_"‘ and the ‘identifier‘ to avoid naming conflicts. - Merges the cohort and reference data frames based on the specified identifier columns. - Adds a new logical column (‘mapped_to_<col_reference>‘) to indicate if a match was found.

Value

A data frame that merges the cohort and reference data frames based on the specified identifier.Includes an additional column indicating whether a match was found.

MetaboMap *MetaboMap*

Description

Matches one or multiple metabolites identifiers of interest between the study dataframe and the generated reference dataframe from HMDB.

Usage

```
MetaboMap(
  df_study,
  study_cols,
  identifiers,
  df_reference,
  details = FALSE,
  original_colname
)
```

Arguments

df_study	A data frame containing metabolite information from the study, where each row corresponds to a metabolite and each column contain one identifier.
study_cols	A column name or a vector of column names in 'df_study' corresponding to identifiers to be mapped.
identifiers	An identifier or a vector of identifiers to be mapped; must be the same length and same order as 'study_cols'. It can takes values c("HMDB","NAME","KEGG","CHEBI","INCHIKEY
df_reference	Dataframe generated by the function hmdbextract().
details	A boolean- Default Value is FALSE. TRUE allows to have additional details for each mapping identifier.
original_colname	A column name in 'df_study' containing the original metabolites label in the study

Value

A data frame object containing the mapped identifiers with confidence levels.

Examples

```
study_data <- data.frame(
  metabolite_name = c("1-Methylhistidine", "2-Ketobutyric acid"),
  HMDB_id = c("HMDB0000001", "HMDB0000005")
)

reference_data <- data.frame(
  accession = c("HMDB0000001", "HMDB0000002", "HMDB0000005"),
  name = c("1-Methylhistidine", "1,3-Diaminopropane", "2-Ketobutyric acid"),
  all_HMDBs = c(
    "HMDB0000001|HMDB00001|HMDB0004935",
    "HMDB0000002|HMDB00002",
```

```
    "HMDB0000005|HMDB00005|HMDB0006544"
  ),
  all_names = c(
    "1-Methylhistidine|Pi-methylhistidine|",
    "1,3-Diaminopropane|1,3-Propanediamine|1,3-Propylenediamine",
    "2-Ketobutyric acid|2-Ketobutanoic acid|2-Oxobutyric acid"
  )
)

result <- MetaboMap(
  df_study = study_data,
  study_cols = c("metabolite_name", "HMDB_id"),
  identifiers = c("NAME", "HMDB"),
  df_reference = reference_data,
  details = FALSE,
  original_colname = "metabolite_name"
)
```

`process_inchikey` *Process InChIKey*

Description

Processes a specified column containing InChIKey strings by extracting the first ‘N_skeleton‘ characters to create a skeleton InChIKey and the first ‘N_inchikey‘ characters to create a processed InChIKey.

Usage

```
process_inchikey(df, col_name, N_skeleton = 14, N_inchikey = 23)
```

Arguments

`df`	A data frame containing the column with InChIKey strings.
`col_name`	The name of the column containing the InChIKey strings to be processed.
`N_skeleton`	An integer specifying the number of characters to extract for the skeleton InChIKey. Default is 14.
`N_inchikey`	An integer specifying the number of characters to extract for the processed InChIKey. Default is 23.

Details

- This function takes a column with InChIKey strings and extracts a skeleton version (first 14 characters) and a processed version (first 23 characters) of each InChIKey. - It returns the data frame with these new columns added.

Value

A data frame with two new columns: - ‘inchikey_skeleton‘: A new column containing the first ‘N_skeleton‘ characters of the specified InChIKey. - ‘inchikey_processed‘: A new column containing the first ‘N_inchikey‘ characters of the specified InChIKey.

process_labels *Process Labels*

Description

Cleans and standardizes labels by removing spaces, dashes, uppercase letters, special characters (such as single and double quotes, asterisks), and numbers in parentheses at the end of the label.

Usage

```
process_labels(df, col_name, reference)
```

Arguments

df	A data frame containing the labels to be processed.
col_name	The name of the column containing the labels to be processed.
reference	Boolean, if equal=TRUE the function will not remove the letters between parenthesis

Details

- The function performs several text cleaning operations: - It removes spaces and dashes. - It converts the text to lowercase. - It removes single and double quotes, and asterisks. - It uses an external function ‘remove_letters_between_parentheses‘ to remove shorter names within parentheses. - It also removes numeric values in parentheses at the end of the label (e.g., "(1)", "(2)"). - The cleaned labels are stored in a new column ‘label_processed‘.

Value

A data frame with a new column ‘label_processed‘, which contains the cleaned labels.

---

remove_letters_between_parentheses
Remove Letters Between Parentheses

Description

Removes any letters enclosed within parentheses from a given text string. It leaves the parentheses empty or removes them entirely if they contain only letters.

Usage

```
remove_letters_between_parentheses(text)
```

Arguments

text	A character string containing text where letters within parentheses will be removed.

Details

- The function uses a regular expression to match any sequence of letters inside parentheses and replace them with empty strings. - It ensures that parentheses without letters are also cleaned, resulting in no empty parentheses remaining.

Value

A character string with letters inside parentheses removed. The parentheses themselves are also removed.

---

remove_stars_after_NA *Remove Stars After "NA"*

Description

This function removes all asterisks (‘*‘) that appear immediately after the string "NA" in a given column of data. It uses a regular expression to identify and replace such patterns.

Usage

```
remove_stars_after_NA(column)
```

Arguments

column	A character vector representing a column of data. This can be a single column from a dataframe or a standalone vector.

Value

A character vector with all occurrences of "NA" followed by one or more asterisks (‘*‘) replaced with "NA".

---

safe_extract *safe_extract*

Description

Extracts the text content from an XML node using the specified XPath expression and namespace. Returns "NA" if an error occurs during extraction.

Usage

```
safe_extract(xml_node, xpath, ns)
```

Arguments

xml_node	An XML node object from which to extract content.
xpath	A character string specifying the XPath expression to locate the desired element within the XML node.
ns	A named list of namespace definitions to use when resolving the XPath expression.

Value

A character string containing the extracted text content. Returns "NA" if the specified XPath cannot be resolved or an error occurs.

Index