

Package ‘standardizeSnapshot’

December 22, 2022

Title Standardize and clean Snapshot data

Version 0.4.2

Description standardizeSnapshot is a R package to standardize camera trap records files from the Snapshot Safari project. Snapshot Safari data comes in different standards, following the method that was used to classify pictures (Zooniverse, TrapTagger or Digikam). This package allows to standardize all data sources to a unique file format and then cleans the files to homogenize records.

Depends R (>= 3.4.3)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

Roxygen list(markdown = TRUE)

Imports log4r, data.table, chron, lubridate, dplyr, magrittr, stringr, tidyselect, utils

Suggests rmarkdown,
knitr

VignetteBuilder knitr

R topics documented:

clean_species	2
create_logger	2
get_camnames	3
get_csv_files_and_folders	4
get_final_filename	5
guess_classifier	5
read_snapshot_files	6
standardize_snapshot_df	7
standardize_snapshot_list	8
write_log_message	10
write_standardized_df	11
write_standardized_list	12

clean_species	<i>Standardize species</i>
---------------	----------------------------

Description

Eliminate species duplicate names (things like birdofprey and birdsofprey)

Usage

```
clean_species(species)
```

Arguments

species	vector of species names
---------	-------------------------

Value

the vector of species names with standardized names, according to the species names that have already been encountered in the past datasets.

Examples

```
species <- c("zebraplains", "zebraburchells", "duiker",  
            "duikercommon", "aardvark", "lionfemale")  
clean_species(species)
```

create_logger	<i>Create logger</i>
---------------	----------------------

Description

Initializes a logger with a given logfile.

Usage

```
create_logger(  
  my_logfile,  
  threshold = c("INFO", "DEBUG", "WARNING", "ERROR", "FATAL"),  
  console = FALSE  
)
```

Arguments

my_logfile	path to the log file (character).
threshold	logging levels to write to the logger (see loglevel documentation in log4r). Defaults to INFO.
console	Write log messages to console?

Value

Returns a logger and creates a logfile at the given path. If the path given does not exist, also creates this path.

Examples

```
logger <- create_logger(tempfile())
```

get_camnames	<i>Get camera names</i>
--------------	-------------------------

Description

Subsets the locations IDs from the camera names vector.

Usage

```
get_camnames(cameras, locations, silence_warnings = FALSE, logger = NA)
```

Arguments

cameras	a character vector of camera names
locations	a character vector of locations (must be the same length as cameras)
silence_warnings	Should a warning be displayed when locationID is NA?
logger	a log4r logger object if you want logging (can be created with create_logger), else NA.

Value

A vector of camera names without the location prefix (if it was present)

Examples

```
get_camnames(c("APN_A01", "MAD_B01"), c("APN", "MAD"))
```

```
get_csv_files_and_folders
```

Get csv files and folder

Description

Get the names of all csv files from the input character vector.

Usage

```
get_csv_files_and_folders(input, except, basepath, logger = NA)
```

Arguments

input	a character vector of valid paths: can be files and/or folders
except	files to ignore (optional): all paths for (a part of) which match the exact expression in except will be ignored.
basepath	the part of the path that should be ignored when copying final files (i.e. absolute path inside one's computer that should not be copied in final file.)
logger	a log4r logger object if you want logging (can be created with create_logger), else NA.

Details

input can be a vector of files and/or folders. If an element of input is a folder, the function will list all files within input; if the element is a file, the function will only list this file.

Value

A dataframe with columns folders and files, where folders are the paths up to a given file (corresponding to the relative path to the file from basepath), and files are the files paths from folders.

Examples

```
## Not run:
get_csv_files_and_folders(input = c("path/to/datafolder/KGA",
                                     "path/to/datafolder/ATH_Roll1_Snapshot.csv"),
                          except = "KGA/KGA-KHO_together",
                          basepath = "path/to/datafolder")

## End(Not run)
```

get_final_filename	<i>Get final filemane</i>
--------------------	---------------------------

Description

Return the filename from the file columns.

Usage

```
get_final_filename(df)
```

Arguments

df The dataframe to be copied. Must have columns locationID, season, roll.

Value

The filename for this file in the format locationID_Sseason_Rroll.csv It there are several values in locationID, season or roll, they are separated by a dash in the filename: locationID1-locationID2...

Examples

```
zooniverse_std <- standardize_snapshot_df(df = zooniverse, standard = standard)
get_final_filename(zooniverse_std)
```

guess_classifier	<i>Guess classifier</i>
------------------	-------------------------

Description

Guesses the classifier used to annotate the data based on the column names given in colnames_df.

Usage

```
guess_classifier(colnames_df, logger = NA)
```

Arguments

colnames_df A character vector of column names.

logger a log4r logger object if you want logging (can be created with create_logger), else NA.

Value

The classifier: either zooniverse, digikam or traptagger.

Examples

```
guess_classifier(colnames(zooniverse))
guess_classifier(colnames(traptagger))
guess_classifier(colnames(digikam))
```

read_snapshot_files	<i>Read Snapshot files</i>
---------------------	----------------------------

Description

Reads files from a vector of files and/or folders into a list of dataframes.

Usage

```
read_snapshot_files(input, except, basepath, verbose = TRUE, logger = NA)
```

Arguments

input	a character vector of valid paths: can be files and/or folders
except	files to ignore (optional): all paths for (a part of) which match the exact expression in except will be ignored.
basepath	the part of the path that should be ignored when copying final files (i.e. absolute path inside one's computer that should not be copied in final file.)
verbose	Should messages be displayed when reading a folder/file?
logger	a log4r logger object if you want logging (can be created with create_logger), else NA.

Details

The files are assumed to be comma or semicolon-separated CSV. If the filename is MAD_S2_full_report_0-50__agreement_corrected_fin.csv, the function only reads the first 26 columns as this file has empty columns.

Value

A named list of dataframes. Each element of the list is a dataframe with the contents of a file read from the files list given in input. The names of the list are the file names from the root of input: If the element from input is a file, it is a filename. If the element from input is a folder, it is the relative path from basepath to the file.

Examples

```
## Not run:
read_snapshot_files("path/to/datafolder/DHP",
                    basepath = "path/to/datafolder",
                    except = "DHP/DHP+OVE_same_file/*")

## End(Not run)
```

standardize_snapshot_df
Standardize dataframe

Description

Standardizes a dataframe to the Snapshot standard.

Usage

```
standardize_snapshot_df(
  df,
  standard_df,
  locationID_digikam,
  classifier,
  logger = NA,
  verbose = TRUE
)
```

Arguments

df	The dataframe to standardize. It is expected to match the data format for either Zooniverse, TrapTagger or Digikam processed data (i.e. have column names defined in standard_df).
standard_df	The standard dataframe to match column names to the new standard. A dataframe with ≥ 2 columns, one of which must be named zooniverse, digikam or traptagger and another one must be named new.
locationID_digikam	Optional character locationID to use for Digikam data (will display a warning if not provided for Digikam data.) Indeed, for Digikam data, the locationID cannot be inferred from other columns.
classifier	Optional character for the classifier.
logger	a log4r logger object if you want logging (can be created with create_logger), else NA.
verbose	Should a lot of details be given when executing function?

Details

Dates and times are standardized to YYYY-MM-DD and HH:MM:SS. The columns locationID, cameraID, roll, capture and season (if classifier is zooniverse) are filled with meaningful information. Species names (snapshotName) are standardized to match the existing, known species names. For column locationID:

- The location code DHP is replaced with OVE if the corresponding camera code starts with 'O'.
- The location code KGA is replaced with KHO if the corresponding camera code starts with 'KHO'.

For column cameraID:

- For TrapTagger data: will remove the leading location code part for all data (eg if location is ATH, will change cameras ATH_A01 -> A01). Also, if the location code is KHO, SAM or TSW: will remove the dash in the camera name (e.g KHO_E_A01 -> EA01)
- For Zooniverse data: if the location code is KHO, will replace KHOG with E and KHOL with M in cameras. If the location code is DHP, will remove leading D in cameraID. If the location code is OVE, will remove leading O in cameraID.
- For column eventID: the event ID formatted as season#cam_site#roll#event_no.

Value

The standardized dataframe: it has the same columns as specified in standard_df\$new, dates and times are standardized and some columns regarding information on the capture are filled. Species names (snapshotName) and cameraID and locationID are standardized (see details). Columns are in the same order as provided in standard_df\$new and the rows are ordered by ascending camera, date and time.

Examples

```
standardize_snapshot_df(zooniverse, standard)
standardize_snapshot_df(traptagger, standard)
standardize_snapshot_df(digikam, standard, locationID_digikam = "MOK")
```

standardize_snapshot_list

Standardize a list of dataframes

Description

Standardizes a list of dataframes to the Snapshot standard.

Usage

```
standardize_snapshot_list(
  df_list,
  standard_df,
  classifier,
  logger = NA,
  verbose = TRUE
)
```


Arguments

<code>df_list</code>	a list of dataframes.
<code>standard_df</code>	The standard dataframe to match column names to the new standard. A dataframe with ≥ 2 columns, one of which must be named <code>zooniverse</code> , <code>digikam</code> or <code>traptagger</code> and another one must be named <code>new</code> .
<code>classifier</code>	Optional character or vector of characters for the classifier.
<code>logger</code>	a <code>log4r</code> logger object if you want logging (can be created with <code>create_logger</code>), else NA.
<code>verbose</code>	Should a lot of details be given when executing function?

Details

Dates and times are standardized to YYYY-MM-DD and HH:MM:SS. The columns `locationID`, `cameraID`, `roll`, `capture` and `season` (if classifier is `zooniverse`) are filled with meaningful information. Species names (`snapshotName`) are standardized to match the existing, known species names. For column `locationID`:

- The location code DHP is replaced with OVE if the corresponding camera code starts with 'O'.
- The location code KGA is replaced with KHO if the corresponding camera code starts with 'KHO'.

For column `cameraID`:

- For `TrapTagger` data: will remove the leading location code part for all data (eg if `location` is `ATH`, will change cameras `ATH_A01` -> `A01`). Also, if the location code is `KHO`, `SAM` or `TSW`: will remove the dash in the camera name (e.g `KHO_E_A01` -> `EA01`)
- For `Zooniverse` data: if the location code is `KHO`, will replace `KHOG` with `E` and `KHOL` with `M` in cameras. If the location code is `DHP`, will remove leading `D` in `cameraID`. If the location code is `OVE`, will remove leading `O` in `cameraID`.
- For column `eventID`: the event ID formatted as `season#cam_site#roll#event_no`.

Value

The list of standardized dataframes: they the same columns as specified in `standard_df$new`, dates and times are standardized and some columns regarding information on the capture are filled. Species names (`snapshotName`) and `cameraID` and `locationID` are standardized (see details). Columns are in the same order as provided in `standard_df$new` and the rows are ordered by ascending camera, date and time.

Note

If the file name is `Marinmane NR _record_table_0min_deltaT_2021-06-10.csv`, will apply the location code `MAR`.

Examples

```
df_list <- list(zooniverse, digikam, traptagger)

# Digikam data must be named with at least the locationID code
names(df_list)[2] <- "MOK"
standardize_snapshot_list(df_list, standard)

# The name for Digikam data can also a filename starting with the locationID code
names(df_list)[2] <- "MOK_record_table_0min_deltaT_2021-05-07.csv"
standardize_snapshot_list(df_list, standard)
```

write_log_message	<i>Write log message</i>
-------------------	--------------------------

Description

Writes a log message. If a logger is provided, writes to that logger; if it is NA, displays a message.

Usage

```
write_log_message(
  message,
  logger = NA,
  level = c("info", "warn", "error", "debug")
)
```

Arguments

message	The message to display/write to the logger
logger	Logger to write to (log4r object of class logger) (defaults to NA)
level	Logging level: either info, warn, debug or error.

Details

- if logger is not NA, writes a message to the logger file with the specified level.
- else, doesn't do anything.

Value

Will either write the message specified in message to the logger or do nothing if logger is NA.

Examples

```
logger <- create_logger(tempfile()) # Create a logger
write_log_message("Test", level = "debug", logger = logger)
write_log_message("Test", level = "info", logger = logger)
write_log_message("Test", level = "warn", logger = logger)
write_log_message("Test", level = "error", logger = logger)
```

write_standardized_df *Write the standardized file*

Description

Writes a file to a given location. If to does not exist, it is created, and if filename is not provided, a default standardized name is chosen.

Usage

```
write_standardized_df(
  df,
  to,
  filename,
  write = TRUE,
  return_path = ifelse(write, FALSE, TRUE),
  verbose = TRUE,
  logger = NA
)
```

Arguments

df	The standardized file
to	The target folder to copy data in. If it does not exist, will be created.
filename	The name to give to the file.
write	if TRUE (default), will write the df in the to folder, and df will be named filename.
return_path	Should the path be returned?
verbose	Should messages be displayed when creating a folder/file?
logger	a log4r logger object if you want logging (can be created with create_logger), else NA.

Value

Writes the file to the folder to/filename. Also returns the path to/filename if return_path == TRUE.

Examples

```
std_dat <- standardize_snapshot_df(zooniverse, standard)

# Don't write data
write_standardized_df(std_dat,
  to = "/home/lnicvert/Documents/PhD/Snapshot/data/2_standardized_data",
  write = FALSE)
# Don't write data and use custom name
```

```

write_standardized_df(std_dat,
                      to = "/home/lnicvert/Documents/PhD/Snapshot/data/2_standardized_data",
                      filename = "myname.csv",
                      write = FALSE)
# Write file to temporary location
write_standardized_df(std_dat,
                      to = tempdir())

```

```
write_standardized_list
```

Write the standardized files

Description

Writes a list of files to a given location. If to does not exist, it is created, and if filename is not provided, a default standardized name is chosen.

Usage

```

write_standardized_list(
  df_list,
  filenames,
  to,
  write = TRUE,
  return_path = ifelse(write, FALSE, TRUE),
  verbose = TRUE,
  logger = NA
)

```

Arguments

df_list	The standardized files list to write. If the list is named, the subdirectory structure in the names will be used to replicate the subdirectiry structure in the destination.
filenames	(Optional) vector of customized file names.
to	Destination folder to write to.
write	Should the result be written or only the path returned?
return_path	Should the path be returned?
verbose	Should messages be displayed when creating a folder/file?
logger	a log4r logger object if you want logging (can be created with create_logger), else NA.

Value

Writes the files to/filename1, to/filename2... Also returns the paths to/filename1, to/filename2... if return_path == TRUE.

Examples

```
# Example with a subdirectory structure (inferred from the filenames)
df_list <- list(zooniverse, digikam, traptagger)
names(df_list) <- c("APN/APN.csv", "MOK/MOK.csv", "ATH/ATH.csv")
std_list <- standardize_snapshot_list(df_list, standard)

# Don't write data
write_standardized_list(std_list,
                        to = "/home/lnicvert/Documents/PhD/Snapshot/data/2_standardized_data",
                        write = FALSE)
# Don't write data and use custom name
write_standardized_list(std_list,
                        to = "/home/lnicvert/Documents/PhD/Snapshot/data/2_standardized_data",
                        filenames = c("myname1.csv", "myname2.csv", "myname3.csv"),
                        write = FALSE)
# Write files to temporary location
write_standardized_list(std_list, to = tempdir())

# Without a subdirectory structure (and without list names)
df_list <- list(zooniverse, digikam, traptagger)
names(df_list)[2] <- "MOK"
std_list <- standardize_snapshot_list(df_list, standard)

# Write files to temporary location
write_standardized_list(std_list, to = tempdir())
```