Package ‘tempaural’

August 16, 2020

Title TODO another todo for toing the todo

Date 2020-06-08

Version 0.0.1

Description Some description goes here TODO

Depends R (>= 3.00)

Imports behavr,
data.table,
seewave,
tuneR,
hms,
stringr,
lubridate,
memoise,
readr,
digest

Suggests testthat,
covr,
knitr

Remotes rethomics/tempaural

License GPL-3

Encoding UTF-8

LazyData true

URL `https://github.com/rethomics/tempaural`

BugReports `https://github.com/rethomics/tempaural/issues`

RoxygenNote 7.1.1

Roxygen list(markdown = TRUE)

R topics documented:

`extract_audio_chunk` *Extracts a section (chunk) of an audio file*

Description

Read an arbitrary section of an mp3 file as a Wave object

Usage

```
extract_audio_chunk(path, start, duration, quiet = TRUE)
```

Arguments

`path`	the path to the file
`start`	the staring point, in seconds
`duration`	the length of the chunk, in seconds
`quiet`	logical, whether to hide the verbose output

Value

a tuneR::Wave object corresponding to the parsed chunk

`file_duration` *Retrieve total audio file duration*

Description

Uses ffprobe to estimate the duration of an audio resource

Usage

```
file_duration(path)
```

Arguments

`path`	the path to the file

Value

a number of seconds

map_dir_chunks *Apply a user-defined function on target audio files*

Description

Uses a metadata table to query a local audio file library and apply an arbitrary function on consecutive time windows

Usage

```
map_dir_chunks(
  metadata,
  root_dir,
  FUN,
  chunk_duration,
  tz = "UTC",
  cache = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

`metadata`	a metadata table. See details.
`root_dir`	the root directory of the local library. See details.
`FUN`	a function to be run on each chunk. See details.
`chunk_duration`	the length of the chunk, in seconds
`tz,`	the timezone the files and provided dates are formatted in, default 'UTC'
`cache`	an optional directory, where the chunk is indefinitely stored
`verbose`	logical, whether to show progress and other messages
`...`	arguments to be passed to `FUN`

Details

Metadata is a table in which each row defines an individual. It must have the following columns:

- `id` – a character that uniquely identifies a biological individual. For each value of `id`, a sub-directory, with the same name, must exist in `root_dir`.
- `start_datetime` – formatted as YYYY-MM-DD HH:MM:SS The first requested time point for the individual. **Importantly,** `start_datetime` **will be used as the reference $t_0$.** in other words, if you want to express time relative to 10:00:00 – e.g. as it would be a ZT0 – you can specify `start_datetime` as "YYYY-MM-DD 10:00:00".
- `end_datetime` – the last requested time point for an individual. In addition, metadata can contain user-defined columns that will be used as metavariables (e.g. individual's genotype, treatment, ...)

Each id defined in metadata must correspond to a subdirectory in root_dir with the exact same name. For instance, if you have "animal_01" and "animal_02" in the id feild of your metadata, you then will have the directories "<root_dir>/animal_01" and "<root_dir>/animal_02".

The function FUN must use a tuneR::Wave as an input and output a named list. Each element of the list will be parsed as a new column in the resulting behavr::behavr table.

Processing long audio file might be very long, according to the funcion (FUN) that is mapped. If caching is turned on, the results of the computation on each chunk will be saved in a custom directory (cache). In other words, the first time FUN is run on a chink, the result is saved, and will not be recomputed, as long as the same function is applied on the same chunk, from the same file. Defining cache as a local directory turns caching on, and saves R objects accordingly. Deleting the content of this directory is safe, but implies subsequent calls to FUN will be reevaluated.

Value

a behavr::behavr table. The metadata corresponds to the user-provided metadata. The data has the variables:

- id – matching the id in the metadata.
- t – the time in second from the start_datetime requested for this id
- ... – variables computed by FUN

Examples

```
# get the path the the package-provided example directory
exple_dir = tempaural::tempaural_example_dir()
# show all the files in it
print(list.files(exple_dir, recursive = TRUE))
metadata = data.frame(id=c('ID1','ID2'),
                      start_datetime = c('2020-08-09 12:09:00','2020-08-09 17:08:00'),
                      end_datetime = c('2020-08-09 17:15:00', '2020-08-09 17:15:00'),
                      genotype=c('addesaf','fewsfr'))
print(metadata)
# a function that takes a wave as input and just outputs its duration
# and a random variable
my_function <- function(wave){
 out <- list(
   my_var = rnorm(1),
     duration = length(wave@left)/wave@samp.rate
     )
 out
}
# now we map this function to all the matching audio chunks
dt <- tempaural::map_dir_chunks(metadata, exple_dir,
                                FUN=my_function, chunk_duration = 60)
```

---

tempaural_example *Get path to tempaural example*

Description

tempaural comes with a sample audio files in its inst/extdata directory. tempaural allow make them easy to access.

Usage

```
tempaural_example(path = NULL)

tempaural_example_dir()
```

Arguments

`path`	Name of file. If `NULL`, the example files will be listed.

Examples

```
# list all files
tempaural_example()
```

Index