

Package: actinet (via r-universe)

September 20, 2026

Title Estimate Human Activity from Accelerometry Data

Version 0.4.0

Description Interfaces the 'actinet' Python module
<<https://github.com/OxWearables/actinet>> for an activity
classification model based on self-supervised learning for
wrist-worn accelerometer data.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports assertthat, curl, lubridate, methods, readr, reticulate (>= 1.42.0), rlang

Suggests tidyr, dplyr, ggplot2, testthat (>= 3.0.0), callr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Language en-US

URL <https://github.com/jhuwit/actinet>,
<https://jhuwit.github.io/actinet/>

BugReports <https://github.com/jhuwit/actinet/issues>

Config/pak/sysreqs libpng-dev libssl-dev python3 libx11-dev

Repository <https://jhuwit.r-universe.dev>

Date/Publication 2026-08-21 21:02:12 UTC

RemoteUrl <https://github.com/jhuwit/actinet>

RemoteRef HEAD

RemoteSha 8203ccd780484fdc16f71fb03c5a8f18011df4d2

Contents

ac_load_model	2
acti_net	3
actinet	3
have_actinet	6
py_require_actinet	6
Index	7

ac_load_model	<i>Load Actinet Model</i>
---------------	---------------------------

Description

Load Actinet Model

Usage

```
ac_load_model(  
    classifier = c("walmsley", "willetts"),  
    model_path = NULL,  
    check_md5 = TRUE,  
    force_download = FALSE,  
    as_python = TRUE  
)  
  
ac_model_filename(classifier = c("walmsley", "willetts"))  
  
ac_download_model(  
    model_path,  
    classifier = c("walmsley", "willetts"),  
    check_md5 = TRUE,  
    ...  
)
```

Arguments

- | | |
|----------------|--|
| classifier | type of the model: either walmsley or willetts |
| model_path | the file path to the model. If on disk, this can be re-used and not re-downloaded. If NULL, will download to the temporary directory |
| check_md5 | Do a MD5 checksum on the file |
| force_download | force a download of the model, even if the file exists |
| as_python | Keep model object as a python object |
| ... | for ac_download_model, additional arguments to pass to curl::curl_download() |

Value

A model from Python. `ac_download_model` returns a model file path.

acti_net	<i>Minute Level actinet output</i>
----------	------------------------------------

Description

Minute Level actinet output

Usage

```
acti_net(...)  
  
py_acti_net(...)
```

Arguments

... arguments to pass to [actinet](#)

Value

A minute-level `data.frame`

actinet	<i>Run Actinet Model on Data</i>
---------	----------------------------------

Description

Run Actinet Model on Data
Run actinet with Python

Usage

```
actinet(  
  file,  
  outdir = tempfile(),  
  classifier = NULL,  
  sample_rate = NULL,  
  model_path = NULL,  
  pytorch_device = NULL,  
  no_hmm = FALSE,  
  require_sleep_above = NULL,  
  single_sleep_block = FALSE,  
  force_download = FALSE,
```

```

    exclude_first_last = NULL,
    exclude_wear_below = NULL,
    csv_start_row = NULL,
    csv_txyz = NULL,
    csv_txyz_idx = NULL,
    csv_date_format = NULL,
    calibration_stdtol_min = NULL,
    plot_activity = FALSE,
    cache_classifier = FALSE,
    verbose = TRUE
)

py_actinet(
  ...,
  pyenv_function = function() {
    actinet::py_require_actinet()
  },
  show = TRUE
)

```

Arguments

file	accelerometry file to process, including CSV, CWA, GT3X, and GENEActiv bin files
outdir	folder location to save output files
classifier	Enter custom activity classifier file to use. Default: walmsley (Walmsley (2020) doi:10.1136/bjsports-2021-104050 annotations of activity intensity). Can also enter path to local classifier (.joblib.lzma) file.
sample_rate	Sample rate for measurement, otherwise inferred.
model_path	the file path to the model. If on disk, this can be re-used and not re-downloaded. If NULL, will download to the temporary directory
pytorch_device	torch device to use, e.g.: 'cpu' or 'cuda:0'. Default: 'mps' if available, otherwise 'cpu'
no_hmm	Disable HMM post-processing
require_sleep_above	Require sleep blocks to exceed a minimum duration, otherwise be classified as sedentary. Pass values as strings, e.g.: '2H', '30min'. Default: None (no requirement)
single_sleep_block	Recognize only one sleep block per day, all other sleep blocks will be converted to sedentary
force_download	Force download of classifier file
exclude_first_last	first,last,both Exclude first, last or both days of data. Default: None (no exclusion)

exclude_wear_below	Exclude days with wear time below threshold. Pass values as strings, e.g.: '12H', '30min'. Default: None (no exclusion)
csv_start_row	Row number to start reading a CSV file. Default: 1 (First row)
csv_txyz	CSV_TXYZ Column names for time, x, y, z in CSV files. Comma_ separated string. Default: 'time,x,y,z'
csv_txyz_idx	Column indices for time,x,y,z (0_indexed, e.g., '0,1,2,3'). Overrides csv_txyz.
csv_date_format	Date time format for csv file when reading a csv file. See https://docs.python.org/3/library/datetime.html#strftime-and-strptime-format-codes for more possible codes. Default: '%Y-%m-%d %H:%M:%S.%f' (e.g. '2023-10-01 12:34:56.789')
calibration_stdtol_min	Minimum standard deviation tolerance (g) for detecting stationary periods for calibration. Default: None
plot_activity	Plot the predicted activity labels
cache_classifier	Download and cache classifier file and model modules for offline usage
verbose	print diagnostic messages
...	arguments to pass to actinet
pyenv_function	function that loads the forest Python package. By default, it uses <code>reticulate::py_import("actinet")</code> to import the package. If this function has an <code>args</code> argument, the output of <code>pyenv_function</code> will be re-assigned to <code>args</code> .
show	Logical, whether to show the standard output on the screen while the child process is running, passed to callr::r()

Value

A list of the results (data.frame), summary of the results, adjusted summary of the results, and information about the data.

Examples

```
actinet_check_result = function() {
  res = try({suppressWarnings(actinet_check())})
  if (inherits(res, "try-error")) {
    res = FALSE
  }
  res
}

file = system.file("extdata/P30_wrist100.csv.gz", package = "actinet")
if (actinet_check_result()) {
  out = try({actinet(file = file)})
  if (!inherits(out, "try-error")) {
    data = readr::read_csv(out$outfiles[1])
    daily_data = readr::read_csv(out$outfiles[3])
  }
}
```

```
    }  
}
```

have_actinet	<i>Check the actinet Python Module</i>
--------------	--

Description

Check the actinet Python Module

Usage

```
have_actinet()  
  
actinet_check()  
  
actinet_version()
```

Value

A logical value indicating whether the actinet Python module is available.

Examples

```
if (have_actinet()) {  
  actinet_version()  
}
```

py_require_actinet	<i>Command for py_require for actinet</i>
--------------------	---

Description

Command for py_require for actinet

Usage

```
py_require_actinet(...)
```

Arguments

```
... arguments to pass to reticulate::py\_require\(\)
```

Value

A logical value indicating whether the package is available.

Index

`ac_download_model (ac_load_model)`, [2](#)
`ac_load_model`, [2](#)
`ac_model_filename (ac_load_model)`, [2](#)
`acti_net`, [3](#)
`actinet`, [3](#), [3](#), [5](#)
`actinet_check (have_actinet)`, [6](#)
`actinet_version (have_actinet)`, [6](#)

`callr::r()`, [5](#)
`curl::curl_download()`, [2](#)

`have_actinet`, [6](#)

`py_acti_net (acti_net)`, [3](#)
`py_actinet (actinet)`, [3](#)
`py_require_actinet`, [6](#)

`reticulate::py_require()`, [6](#)