Package 'tfruns'

May 8, 2026

Type Package

Title Training Run Tools for 'TensorFlow'

Version 1.5.4

Description Create and manage unique directories for each 'TensorFlow' training run. Provides a unique, time stamped directory for each run along with functions to retrieve the directory of the latest run or latest several runs.

License Apache License 2.0

URL https://github.com/rstudio/tfruns

BugReports https://github.com/rstudio/tfruns/issues

Depends R (>= 3.1)

Imports utils, jsonlite (>= 1.2), base64enc, yaml, config, magrittr, whisker, tidyselect, rlang, rstudioapi (>= 0.7), reticulate

Suggests testthat, knitr, withr, here, rmarkdown

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation no

Author Tomasz Kalinowski [ctb, cre],
Daniel Falbel [ctb],
JJ Allaire [aut],
RStudio [cph, fnd],
Mike Bostock [cph] (D3 library - https://d3js.org/),
Masayuki Tanaka [cph] (C3 library - http://c3js.org/),
jQuery Foundation [cph] (jQuery library),
jQuery contributors [cph] (jQuery library; authors:
inst/views/components/jquery-AUTHORS.txt),
Shaun Bowe [cph] (jQuery visibilityChanged plugin),
Materialize [cph] (Materizlize library - https://materializecss.com/),
Yuxi You [cph] (Vue.js library - https://vuejs.org/),
Kevin Decker [cph] (jsdiff library -

https://github.com/kpdecker/jsdiff/),
Rodrigo Fernandes [cph] (diff2html library - https://diff2html.xyz/),
Ivan Sagalaev [cph] (highlight.js library - https://highlightjs.org/),
Yauheni Pakala [cph] (highlightjs-line-numbers library)

Maintainer Tomasz Kalinowski <tomasz@rstudio.com>

Repository CRAN

Date/Publication 2025-08-20 14:00:03 UTC

Contents

`clean_runs` *Clean run directories*

Description

Remove run directories from the filesystem.

Usage

```
clean_runs(
  runs = ls_runs(runs_dir = runs_dir),
  runs_dir = getOption("tfruns.runs_dir", "runs"),
  confirm = interactive()
)

purge_runs(
  runs_dir = getOption("tfruns.runs_dir", "runs"),
  confirm = interactive()
)
```

Arguments

runs	Runs to clean. Can be specified as a data frame (as returned by ls_runs()) or as a character vector of run directories.
runs_dir	Directory containing runs. Defaults to "runs" beneath the current working directory (or to the value of the tfruns.runs_dir R option if specified).
confirm	TRUE to confirm before performing operation

Details

The clean_runs() function moves the specified runs (by default, all runs) into an "archive" subdirectory of the "runs" directory.

The purge_runs() function permanently deletes the "archive" subdirectory.

See Also

Other run management: copy_run()

Examples

```
## Not run:
clean_runs(ls_runs(completed == FALSE))

## End(Not run)
```

---

compare_runs *Compare training runs*

Description

Render a visual comparison of two training runs. The runs are displayed with the most recent run on the right and the earlier run on the left.

Usage

```
compare_runs(runs = ls_runs(latest_n = 2), viewer = getOption("tfruns.viewer"))
```

Arguments

runs	Character vector of 2 training run directories or data frame returned from ls_runs() with at least 2 elements.
viewer	Viewer to display training run information within (default to an internal page viewer if available, otherwise to the R session default web browser).

copy_run *Copy run directories*

Description

Functions for exporting/copying run directories and run artifact files.

Usage

```
copy_run(run_dir, to = ".", rename = NULL)

copy_run_files(run_dir, to = ".", rename = NULL)
```

Arguments

run_dir	Training run directory or data frame returned from ls_runs().
to	Name of parent directory to copy run(s) into. Defaults to the current working directory.
rename	Rename run directory after copying. If not specified this defaults to the base-name of the run directory (e.g. "2017-09-24T10-54-00Z").

Details

Use copy_run to copy one or more run directories.

Use copy_run_files to copy only files saved/generated by training run scripts (e.g. saved models, checkpoints, etc.).

Value

Logical vector indicating which operation succeeded for each of the run directories specified.

See Also

Other run management: clean_runs()

Examples

```
## Not run:

# export a run directory to the current working directory
copy_run("runs/2017-09-24T10-54-00Z")

# export to the current working directory then rename
copy_run("runs/2017-09-24T10-54-00Z", rename = "best-run")

# export artifact files only to the current working directory then rename
copy_run_files("runs/2017-09-24T10-54-00Z", rename = "best-model")
```

```
# export 3 best eval_acc to a "best-runs" directory
copy_run(ls_runs(order = eval_acc)[1:3,], to = "best-runs")


## End(Not run)
```

`flags` *Flags for a training run*

Description

Define the flags (name, type, default value, description) which paramaterize a training run. Optionally read overrides of the default values from a "flags.yml" config file and/or command line arguments.

Usage

```
flags(
  ...,
  config = Sys.getenv("R_CONFIG_ACTIVE", unset = "default"),
  file = "flags.yml",
  arguments = commandArgs(TRUE)
)

flag_numeric(name, default, description = NULL)

flag_integer(name, default, description = NULL)

flag_boolean(name, default, description = NULL)

flag_string(name, default, description = NULL)
```

Arguments

`...`	One or more flag definitions
`config`	The configuration to use. Defaults to the active configuration for the current environment (as specified by the `R_CONFIG_ACTIVE` environment variable), or `default` when unset.
`file`	The flags YAML file to read
`arguments`	The command line arguments (as a character vector) to be parsed.
`name`	Flag name
`default`	Flag default value
`description`	Flag description

Value

Named list of training flags

Config File Flags

Config file flags are defined a YAML configuration file (by default named "flags.yml"). Flags can either appear at the top-level of the YAML or can be inclued in named configuration sections (see the config package for details).

Command Line Flags

Command line flags should be of the form `--key=value` or `--key value`. The values are assumed to be valid `yaml` and will be converted using `yaml::yaml.load()`.

Examples

```
## Not run:
library(tfruns)

# define flags and parse flag values from flags.yml and the command line
FLAGS <- flags(
  flag_numeric('learning_rate', 0.01, 'Initial learning rate.'),
  flag_integer('max_steps', 5000, 'Number of steps to run trainer.'),
  flag_string('data_dir', 'MNIST-data', 'Directory for training data'),
  flag_boolean('fake_data', FALSE, 'If true, use fake data for testing')
)

## End(Not run)
```

`is_run_active` *Check for an active training run*

Description

Check for an active training run

Usage

```
is_run_active()
```

Value

`TRUE` if a training tun is currently active

latest_run *Latest training run*

Description

Latest training run

Usage

```
latest_run(runs_dir = getOption("tfruns.runs_dir", "runs"))
```

Arguments

runs_dir	Directory containing runs. Defaults to "runs" beneath the current working directory (or to the value of the tfruns.runs_dir R option if specified).

Value

Named list with run attributes (or NULL if no runs found)

ls_runs *List or view training runs*

Description

List or view training runs

Usage

```
ls_runs(
  subset = NULL,
  order = "start",
  decreasing = TRUE,
  latest_n = NULL,
  runs_dir = getOption("tfruns.runs_dir", "runs")
)
```

Arguments

subset	Logical expression indicating rows to keep (missing values are taken as false). See subset().
order	Columns to order by (defaults to run start time)
decreasing	TRUE to use decreasing order (e.g. list most recent runs first)
latest_n	Limit query to the latest_n most recent runs
runs_dir	Directory containing runs. Defaults to "runs" beneath the current working directory (or to the value of the tfruns.runs_dir R option if specified).

Details

When printing the results of `ls_runs()`, only `run_dir`, `metric_loss`, `metric_val_loss`, and any columns specified in `order` will be printed.

To view all fields, use `View(ls_runs())`.

Value

Data frame with training runs

`run_dir` *Current run directory*

Description

Returns the current training run directory. If a training run is not currently active (see `is_run_active()`) then the current working directory is returned.

Usage

```
run_dir()
```

Value

Active run direcotry (or current working directory as a fallback)

`run_info` *Summary of training run*

Description

Summary of training run

Usage

```
run_info(run_dir)
```

Arguments

`run_dir`	Training run directory or data frame returned from `ls_runs()`.

Value

Training run summary object with timing, flags, model info, training and evaluation metrics, etc. If more than one `run_dir` is passed then a list of training run summary objects is returned.

See Also

`view_run()`

save_run_comparison *Save a run comparison as HTML*

Description

Save a run comparison as HTML

Usage

```
save_run_comparison(runs = ls_runs(latest_n = 2), filename = "auto")
```

Arguments

runs	Character vector of 2 training run directories or data frame returned from ls_runs() with at least 2 elements.
filename	Path to save the HTML to. If no filename is specified then a temporary file is used (the path to the file is returned invisibly).

save_run_view *Save a run view as HTML*

Description

The saved view includes summary information (flags, metrics, model attributes, etc.), plot and console output, and the code used for the run.

Usage

```
save_run_view(run_dir = latest_run(), filename = "auto")
```

Arguments

run_dir	Training run directory or data frame returned from ls_runs().
filename	Path to save the HTML to. If no filename is specified then a temporary file is used (the path to the file is returned invisibly).

See Also

ls_runs(), run_info(), view_run()

training_run *Run a training script*

Description

Run a training script

Usage

```
training_run(
  file = "train.R",
  context = "local",
  config = Sys.getenv("R_CONFIG_ACTIVE", unset = "default"),
  flags = NULL,
  properties = NULL,
  run_dir = NULL,
  artifacts_dir = getwd(),
  echo = TRUE,
  view = "auto",
  envir = parent.frame(),
  encoding = getOption("encoding")
)
```

Arguments

file	Path to training script (defaults to "train.R")
context	Run context (defaults to "local")
config	The configuration to use. Defaults to the active configuration for the current environment (as specified by the R_CONFIG_ACTIVE environment variable), or default when unset.
flags	Named list with flag values (see flags()) or path to YAML file containing flag values.
properties	Named character vector with run properties. Properties are additional metadata about the run which will be subsequently available via ls_runs().
run_dir	Directory to store run data within
artifacts_dir	Directory to capture created and modified files within. Pass NULL to not capture any artifcats.
echo	Print expressions within training script
view	View the results of the run after training. The default "auto" will view the run when executing a top-level (printed) statement in an interactive session. Pass TRUE or FALSE to control whether the view is shown explictly. You can also pass "save" to save a copy of the run report at tfruns.d/view.html
envir	The environment in which the script should be evaluated
encoding	The encoding of the training script; see file().

Details

The training run will by default use a unique new run directory within the "runs" sub-directory of the current working directory (or to the value of the `tfruns.runs_dir` R option if specified).

The directory name will be a timestamp (in GMT time). If a duplicate name is generated then the function will wait long enough to return a unique one.

If you want to use an alternate directory to store run data you can either set the global `tfruns.runs_dir` R option, or you can pass a `run_dir` explicitly to `training_run()`, optionally using the `unique_run_dir()` function to generate a timestamp-based directory name.

Value

Single row data frame with run flags, metrics, etc.

`tuning_run` *Tune hyperparameters using training flags*

Description

Run all combinations of the specifed training flags. The number of combinations can be reduced by specifying the `sample` parameter, which will result in a random sample of the flag combinations being run.

Usage

```
tuning_run(
  file = "train.R",
  context = "local",
  config = Sys.getenv("R_CONFIG_ACTIVE", unset = "default"),
  flags = NULL,
  sample = NULL,
  properties = NULL,
  runs_dir = getOption("tfruns.runs_dir", "runs"),
  artifacts_dir = getwd(),
  echo = TRUE,
  confirm = interactive(),
  envir = parent.frame(),
  encoding = getOption("encoding")
)
```

Arguments

`file`	Path to training script (defaults to "train.R")
`context`	Run context (defaults to "local")
`config`	The configuration to use. Defaults to the active configuration for the current environment (as specified by the `R_CONFIG_ACTIVE` environment variable), or `default` when unset.

flags	Either a named list with flag values (multiple values can be provided for each flag) or a data frame that contains pre-generated combinations of flags (e.g. via base::expand.grid()). The latter can be useful for subsetting combinations. See 'Examples'.
sample	Sampling rate for flag combinations (defaults to running all combinations).
properties	Named character vector with run properties. Properties are additional metadata about the run which will be subsequently available via ls_runs().
runs_dir	Directory containing runs. Defaults to "runs" beneath the current working directory (or to the value of the tfruns.runs_dir R option if specified).
artifacts_dir	Directory to capture created and modified files within. Pass NULL to not capture any artifcats.
echo	Print expressions within training script
confirm	Confirm before executing tuning run.
envir	The environment in which the script should be evaluated
encoding	The encoding of the training script; see file().

Value

Data frame with summary of all training runs performed during tuning.

Examples

```
## Not run:
library(tfruns)

# using a list as input to the flags argument
runs <- tuning_run(
  system.file("examples/mnist_mlp/mnist_mlp.R", package = "tfruns"),
  flags = list(
    dropout1 = c(0.2, 0.3, 0.4),
    dropout2 = c(0.2, 0.3, 0.4)
  )
)
runs[order(runs$eval_acc, decreasing = TRUE), ]

# using a data frame as input to the flags argument
# resulting in the same combinations above, but remove those
# where the combined dropout rate exceeds 1
grid <- expand.grid(
  dropout1 = c(0.2, 0.3, 0.4),
  dropout2 = c(0.2, 0.3, 0.4)
)
grid$combined_droput <- grid$dropout1 + grid$dropout2
grid <- grid[grid$combined_droput <= 1, ]
runs <- tuning_run(
  system.file("examples/mnist_mlp/mnist_mlp.R", package = "tfruns"),
  flags = grid[, c("dropout1", "dropout2")]
)

## End(Not run)
```

unique_run_dir *Create a unique run directory*

Description

Create a new uniquely named run directory within the specified `runs_dir`.

Usage

```
unique_run_dir(
  runs_dir = getOption("tfruns.runs_dir", "runs"),
  seconds_scale = 0
)
```

Arguments

`runs_dir`	Directory containing runs. Defaults to "runs" beneath the current working directory (or to the value of the `tfruns.runs_dir` R option if specified).
`seconds_scale`	Decimal scale for the seconds component of the timestamp. Defaults to 0 which results in only the rounded seconds value being used in the timestamp. Specify larger numbers to include a decimal component (useful if you need to create many unique run directories at the same time).

Details

The directory name will be a timestamp (in GMT time). If a duplicate name is generated then the function will wait long enough to return a unique one.

view_run *View a training run*

Description

View metrics and other attributes of a training run.

Usage

```
view_run(run_dir = latest_run(), viewer = getOption("tfruns.viewer"))
```

Arguments

`run_dir`	Training run directory or data frame returned from `ls_runs()`.
`viewer`	Viewer to display training run information within (default to an internal page viewer if available, otherwise to the R session default web browser).

See Also

ls_runs(), run_info()

---

view_run_metrics *View metrics for a training run*

Description

Interactive D3 visualization of metrics for a training run. Metrics will be displayed in the RStudio Viewer (if available), otherwise will be displayed in an external web browser.

Usage

```
view_run_metrics(metrics)

update_run_metrics(viewer, metrics)
```

Arguments

metrics	Data frame containing run metrics
viewer	Viewer object returned from view_run_metrics().

Metrics Data Frame

Metrics should be passed as a data frame with one column for each metric. If the metrics are not yet complete (e.g. only metrics for the first several epochs are provided) then metrics in yet to be completed epochs should use NA as their values. For example:

```
data.frame': 30 obs. of  4 variables:
$ loss    : num  0.423 0.201 NA NA NA ...
$ acc     : num  0.873 0.942 NA NA NA ...
$ val_loss: num  0.174 0.121 NA NA NA ...
$ val_acc : num  0.949 0.964 NA NA NA ...
```

If both metrics and validation metrics are provided, you should preface the name of the validation metric with "val_" (e.g. for a metric named "loss" provide validation metrics in "val_loss"). This indicates that the metrics are related which is useful e.g. when plotting metrics.

Realtime Updates

Metrics can be updated in real-time by calling the update_run_metrics() with the run viewer instance returned from view_run_metrics(). For example:

```
# view metrics
viewer <- view_run_metrics(metrics)

# update with new metrics
update_run_metrics(viewer, updated_metrics)
```

Note

Metrics named `"acc"` or `"accuracy"` will automatically use `1.0` as the maximum value on their y-axis scale.

See Also

write_run_metrics

Index