Package ‘tfevents’

May 6, 2024

Title Write Events for 'TensorBoard'

Version 0.0.3

Description Provides a convenient way to log scalars, images, audio, and histograms in the 'tfevent' record file format.
Logged data can be visualized on the fly using 'TensorBoard', a web based tool that focuses on visualizing the training
progress of machine learning models.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

LinkingTo Rcpp

Imports Rcpp, withr, fs, rlang, vctrs, ellipsis, blob, png, digest, cli, zeallot

Suggests testthat (>= 3.0.0), tibble, tidyr, reticulate, rmarkdown, ggplot2, tensorflow, wav

SystemRequirements libprotobuf, protobuf-compiler

URL https://github.com/mlverse/tfevents, https://mlverse.github.io/tfevents/

BugReports https://github.com/mlverse/tfevents/issues

Config/testthat/edition 3

Config/Needs/website tensorflow, webshot2, keras

NeedsCompilation yes

Author Daniel Falbel [aut, cre, cph],
Posit, PBC [cph],
The tl::optional authors [cph] (For the vendored tl::optional code.),
Mark Adler [cph] (For the included crc32c code.)

Maintainer Daniel Falbel <daniel@posit.co>

Repository CRAN

Date/Publication 2024-05-06 14:40:02 UTC

R topics documented:

`as_event` *Coerce an object to a event.*

Description

Coerce an object to a event.

Usage

```
as_event(x, step, wall_time, ...)
```

Arguments

`x`	Object that will be coerced to an event.
`step`	The step that will be used when the event is logged. This is used by TensorBoard when showing data.
`wall_time`	The all time the event will appended to the event. This field is used by TensorBoard when displaying information based on actual time.
`...`	currently unused.

Value

A event vctr with class <tfevents_event>.

Extending as_event

as_event is an S3 generic and you can implement method for your own class. We don't export the event constructor though, so you should implement it in terms of other as_event methods.

Examples

```
as_event(list(hello = 1), step = 1, wall_time = 1)
```

---

collect_events *Collect data from tfevents records*

Description

Collects all events of a kind in a single data.frame ready for analysis.

Usage

```
collect_events(
  logdir = get_default_logdir(),
  n = NULL,
  type = c("any", "summary", "scalar")
)

events_logdir(logdir = get_default_logdir())
```

Arguments

logdir	The log directory that you want to query events from. Either a file path or a connection created with events_logdir().
n	The maximum number of events to read from the connection. If NULL then all events are read, the default is NULL.
type	The kind of events that are to be read. By default all events are read. If a different type is specified, then the result can include other columns as well as more lines.

Value

A tibble with the collected events.

Functions

- events_logdir(): Creates a connection to a logdir that can be reused to read further events later.

Examples

```
temp <- tempfile()
with_logdir(temp, {
  for(i in 1:5) {
    log_event(my_log = runif(1))
  }
})
# collect all events in files, including file description events
collect_events(temp)
# collect summaries in the logdir
collect_events(temp, type = "summary")
# collect only scalar events
collect_events(temp, type = "scalar")
```

---

get_default_logdir *Query and modify the logdir*

Description

log_event() has a notion of default logdir, so you don't need to specify it at every call. These functions allow you to query and the current logdir.

Usage

```
get_default_logdir()

set_default_logdir(logdir = "logs")

with_logdir(logdir, code)

local_logdir(logdir, .env = parent.frame())
```

Arguments

logdir	The logdir that you want to set as default.
code	Expressions that will be evaluated in a context with the new logdir as the default logdir.
.env	Environment that controls scope of changes. For expert use only.

Value

The logdir for get_default_logdir() otherwise invisibly returns NULL

Functions

- set_default_logdir(): Modifies the default logdir.
- with_logdir(): Temporarily modify the default logdir.
- local_logdir(): Temporarily modify thedefault logdir.

Examples

```
temp <- tempfile()
get_default_logdir()
with_logdir(temp, {
 print(get_default_logdir())
})
```

`get_global_step` *Global step counters*

Description

Global step counters

Usage

```
get_global_step(increment = TRUE)

set_global_step(step)
```

Arguments

`increment`	Wether to increment the `step` when getting it.
`step`	New value for `step`.

Details

`tfevents` tracks and automatically increased the step counter whenever `log_event()` is called. Note that, it maintains a separate step counter for each root `logdir`, thus if you change the `logdir` using `set_default_logdir()` or `with_logdir()`, a different step counter will be used.

Value

The global step value for the default logdir, when `get_global_step`, otherwise returns `NULL` invisibly.

Functions

- `set_global_step()`: Set the global step.

Examples

```
temp <- tempfile()
with_logdir(temp, {
  print(get_global_step())
  set_global_step(100)
  print(get_global_step())
})
print(get_global_step())
```

hparams_hparam *Defines a HParam*

Description

Hparam object are used to describe names and domains of hyperparameters so TensorBoard UI can show additional information about them.

Usage

```
hparams_hparam(name, domain = NA, display_name = name, description = name)
```

Arguments

name	Name of the hyperparameter.
domain	A list of values that can be assumed by the hyperparameter. It can be `character()`, `numeric()` or `logical()` vector. You can also pass a named numeric vector with eg `c(min_value = 0, max_value = 10)` in this case, any value in this range is accepted.
display_name	Display name of the hparameter for the TensorBoard UI. By default it's identical to the name.
description	Parameter description. Shown in tooltips around the TensorBoard UI.

Value

A `hparams_hparam` object.

Note

A list of `hparam` values can be passed to `log_hparams_config()` so you define the hyperparameters that are tracked by the experiment.

Examples

```
hparams_hparam("optimizer", domain = c("adam", "sgd"))
hparams_hparam("num_units", domain = c(128, 512, 1024))
hparams_hparam("use_cnn", domain = c(TRUE, FALSE))
hparams_hparam("dropout", domain = c(min_value = 0, max_value = 0.5))
```

hparams_metric *Defines a Metric*

Description

Metric objects are passed to log_hparams_config() in order to define the collection of scalars that will be displayed in the HParams tab in TensorBoard.

Usage

```
hparams_metric(
  tag,
  group = NA,
  display_name = tag,
  description = tag,
  dataset_type = NA
)
```

Arguments

tag	The tag name of the scalar summary that corresponds to this metric.
group	An optional string listing the subdirectory under the session's log directory containing summaries for this metric. For instance, if summaries for training runs are written to events files in ROOT_LOGDIR/SESSION_ID/train, then group should be "train". Defaults to the empty string: i.e., summaries are expected to be written to the session logdir.
display_name	An optional human-readable display name.
description	An optional Markdown string with a human-readable description of this metric, to appear in TensorBoard.
dataset_type	dataset_type: Either "training" or "validation, or NA.

Value

A hparams_metric object.

Examples

```
hparams_metric("loss", group = "train")
hparams_metric("acc")
```

log_event *Log event*

Description

Log event

Usage

```
log_event(..., step = get_global_step(increment = TRUE))
```

Arguments

...	Named values that you want to log. They can be possibly nested, in this case, the enclosing names are considered 'run' names by TensorBoard.
step	The step associated the logs. If NULL, a managed step counter will be used, and the global step is increased in every call to log_event().

Value

Invisibly returns the logged data.

Note

log_event() writes events to the default logdir. You can query the default logdir with get_default_logdir() and modify it with set_default_logdir(). You can also use the with_logdir() context switcher to temporarily modify the logdir.

Examples

```
temp <- tempfile()
with_logdir(temp, {
  log_event(
     train = list(loss = runif(1), acc = runif(1)),
     valid = list(loss = runif(1), acc = runif(1))
  )
})
```

log_hparams *Log hyperaparameters*

Description

Log hyperaparameters

Usage

```
log_hparams(..., trial_id = NA, time_created_secs = get_wall_time())

summary_hparams(..., trial_id = NA, time_created_secs = get_wall_time())
```

Arguments

...	Named values of hyperparameters.
trial_id	A name for the current trail. by default it's the hash of the hparams names and values.
time_created_secs	The time the experiment is created in seconds since the UNIX epoch.

Details

This function should only be called once in a logdir and it will record the set of hyperparameters used in that run. Undefined behavior can happen if it's called more than once in a logdir - specially how TensorBoard behaves during visualization.

Value

A hyperparameter summary. USed for the side effects of logging the hyperparameter to the logdir.

Functions

- `summary_hparams()`: For advanced users only. It's recommended to use the `log_hparams()` function instead. Creates a hyperparameter summary that can be written with `log_event()`.

See Also

`log_hparams_config()`

Examples

```
temp <- tempfile()
with_logdir(temp, {
  log_hparams(optimizer = "adam", num_units = 16)
})
```

`log_hparams_config` *Logs hyperparameters configuration*

Description

Logs the hyperaparameter configuration for a HyperParameter tuning experiment. It allows you to define the domain for each hyperparameters and what are the metrics that should be shown in the TensorBoard UI, along with configuring their display name and descriptions.

Usage

```
log_hparams_config(hparams, metrics, time_created_secs = get_wall_time())

summary_hparams_config(hparams, metrics, time_created_secs = get_wall_time())
```

Arguments

`hparams`	A list of `hparams` objects as created by `hparams_hparam()`.
`metrics`	A list of `metrics` objects as created by `hparams_metric()`. These metrics will be tracked by TensorBoard UI when displaying the hyperparameter tuning table.
`time_created_secs`	The time the experiment is created in seconds since the UNIX epoch.

Value

Invisibly returns the HParam conffuration data as a `summary` object.

Functions

- `summary_hparams_config()`: For advanced users only. Creates a hyperaparameter configuration summary that can be logged with `log_event()`.

Recommendations

When loging hyperparameter tuning experiments, the log directory organization is:

```
- root:
 - log_hparams_config(...)
 - run1:
   - log_hparams(...)
   - log_event(...)
 - run2:
   - log_hparams(...)
   - log_event(...)
```

Ie you should have a root logdir that will only contain the hyperaparameter config log, as created with `log_hparams_config()`. Then each run in the experiment will have it's own logdir as a child directory of the root logdir.

See Also

log_hparams()

---

summary_audio *Summary audio*

Description

Audio summaries can be played withing the TensorBoard UI.

Usage

```
summary_audio(audio, ..., metadata = NULL, tag = NA)

## S3 method for class 'array'
summary_audio(audio, ..., sample_rate = 44100, metadata = NULL, tag = NA)

## S3 method for class 'raw'
summary_audio(audio, ..., metadata = NULL, tag = NA)

## S3 method for class 'blob'
summary_audio(audio, ..., metadata = NULL, tag = NA)
```

Arguments

audio	Object that will be written as an audio event in the tfevents record.
...	Currently unused.
metadata	A metadata object, as created with summary_metadata(). In most cases you don't need to change the default.
tag	A tag that within the TensorBoard UI. See log_event() for other ways of specifying the tag attribute.
sample_rate	The sample rate in Hz associated to the audio values.

Value

An audio summary that can be logged with log_event().

Methods (by class)

- summary_audio(array): Creates a summary from a 3D array with dimensions (batch_size, n_samples, n_channel Values must be in the range [-1, 1].
- summary_audio(raw): Creates an audio summary from a raw vector containing a WAV encoded audio file.
- summary_audio(blob): Creates an audio summary from a blob (ie list of raw vectors) containing WAV encoded audio files.

See Also

Other summary: summary_histogram(), summary_image(), summary_scalar(), summary_text()

Examples

```
tmp <- tempfile()
with_logdir(tmp, {
  summary_audio(array(runif(100), dim = c(1,100, 1)))
})
```

---

summary_histogram *Creates an histogram summary*

Description

Writes an histogram for later analysis in TensorBoard's Histograms and Distributions tab.

Usage

```
summary_histogram(data, ..., metadata = NULL, tag = NA)

## S3 method for class 'numeric'
summary_histogram(data, ..., metadata = NULL, tag = NA, buckets = 30)

## S3 method for class 'array'
summary_histogram(data, ..., metadata = NULL, tag = NA, buckets = 30)
```

Arguments

data	A Tensor of any shape. The histogram is computed over its elements, which must be castable to float64.
...	Currently unused. To allow future expansion.
metadata	A metadata object, as created with summary_metadata(). In most cases you don't need to change the default.
tag	A tag that within the TensorBoard UI. See log_event() for other ways of specifying the tag attribute.
buckets	Optional positive int. The output will have this many buckets, except in two edge cases. If there is no data, then there are no buckets. If there is data but all points have the same value, then all buckets' left and right endpoints are the same and only the last bucket has nonzero count. Defaults to 30 if not specified.

Value

An histogram summary that can be logged with log_event().

Methods (by class)

- `summary_histogram(numeric)`: Creates an histogram summary for a numeric vector.
- `summary_histogram(array)`: Creates an histogram for array data.

See Also

Other summary: `summary_audio()`, `summary_image()`, `summary_scalar()`, `summary_text()`

Examples

```
temp <- tempfile()
with_logdir(temp, {
  for(i in 1:10) {
    log_event(x = summary_histogram(rnorm(10000)))
  }
})
```

---

summary_image *Creates a image summary*

Description

Creates a image summary

Usage

```
summary_image(img, ..., metadata = NULL, tag = NA)

## S3 method for class 'ggplot'
summary_image(img, ..., width = 480, height = 480, metadata = NULL, tag = NA)

## S3 method for class 'array'
summary_image(img, ..., metadata = NULL, tag = NA)

## S3 method for class 'blob'
summary_image(img, ..., width, height, colorspace, metadata = NULL, tag = NA)

## S3 method for class 'raw'
summary_image(img, ..., width, height, colorspace, metadata = NULL, tag = NA)
```

Arguments

`img`	An object that can be converted to an image.
`...`	Currently unused.
`metadata`	A `metadata` object, as created with `summary_metadata()`. In most cases you don't need to change the default.

tag	A tag that within the TensorBoard UI. See log_event() for other ways of specifying the tag attribute.
width	Width of the image.
height	Height of the image.
colorspace	Valid colorspace values are 1 - grayscale, 2 - grayscale + alpha, 3 - RGB, 4 - RGBA, 5 - DIGITAL_YUV, 6 - BGRA

Value

An image summary that can be logged with log_event().

Methods (by class)

- summary_image(ggplot): Cretes an image summary from a ggplot2 graph object. The ... will be forwarded to grDevices::png().
- summary_image(array): Creates an image from an R array. The array should be numeric, with values between 0 and 1. Dimensions should be (batch, height, width, channels).
- summary_image(blob): Creates an image from blob::blob() vctr of PNG encoded images, (eg using png::writePNG()). width, height and colorspace are recycled thus they can be a single scalar or a vector the same size of the images blob.
- summary_image(raw): Creates an image from a png encoded image. Eg, created with png::writePNG(). In this case you need to provide width, height and colorspace arguments.

See Also

Other summary: summary_audio(), summary_histogram(), summary_scalar(), summary_text()

Examples

```
tmp <- tempfile()
with_logdir(tmp, {
  summary_image(array(runif(100), dim = c(1,10, 10, 1)))
})
```

---

summary_metadata *Summary metadata*

Description

Creates a summary metadata that can be passed to multiple summary_ functions.

Usage

```
summary_metadata(
  plugin_name,
  display_name = NA_character_,
  description = NA_character_,
  ...,
  plugin_content = NA
)
```

Arguments

plugin_name	The name of the TensorBoard plugin that might use the summary.
display_name	Display name for the summary.
description	A description of the summary.
...	Currently unused. For future expansion.
plugin_content	An optional plugin content. Note that it will only be used if the C++ function make_plugin_data is aware of plugin_content for the specified plugin name. For advanced use only.

Value

A summary_metadata object.

Examples

```
summary <- summary_scalar(1, metadata = summary_metadata("scalars"))
```

---

summary_scalar *Scalar event*

Description

Scalar event

Usage

```
summary_scalar(value, ..., metadata = NULL, tag = NA)
```

Arguments

value	A numeric scalar value to be logged.
...	Currently unused. To allow future expansion.
metadata	A metadata object, as created with summary_metadata(). In most cases you don't need to change the default.
tag	A tag that within the TensorBoard UI. See log_event() for other ways of specifying the tag attribute.

Value

A <scalar_event> object.

See Also

Other summary: summary_audio(), summary_histogram(), summary_image(), summary_text()

Examples

```
temp <- tempfile()
with_logdir(temp, {
  log_event(loss = summary_scalar(1))
})
```

---

summary_text *Creates a text summary*

Description

Creates a text summary

Usage

```
summary_text(txt, ..., metadata = NULL, tag = NA)

## S3 method for class 'character'
summary_text(txt, ..., metadata = NULL, tag = NA)
```

Arguments

txt	An object that can be converted to a text.
...	Currently unused.
metadata	A metadata object, as created with summary_metadata(). In most cases you don't need to change the default.
tag	A tag that within the TensorBoard UI. See log_event() for other ways of specifying the tag attribute.

Value

A summary that can be logged with log_event().

Methods (by class)

- summary_text(character): Creates a summary from a scalar character vector.

See Also

Other summary: summary_audio(), summary_histogram(), summary_image(), summary_scalar()

Examples

```
temp <- tempfile()
with_logdir(temp, {
  log_event(
    x = "hello world",
    y = summary_text("hello world")
  )
})
```

`value` *Extracts the value of a summary value*

Description

Summaries are complicated objects because they reflect the Protobuf object structure that are serialized in the tfevents records files. This function allows one to easily query vaues from summaries and will dispatch to the correct way to extract images, audio, text, etc from summary values.

Usage

```
value(x, ...)

## S3 method for class 'tfevents_summary_values'
value(x, ..., as_list = FALSE)
```

Arguments

`x`	A `tfevents_summary_values` object.
`...`	Currently unused. To allow future extension.
`as_list`	A boolean indicating if the results should be returned in a list. The default is to return a single value. If you need to extract values from multiple summaries use `as_list = TRUE`.

Value

Depending on the type of the summary it returns an image, audio, text or scalar.

Methods (by class)

- `value(tfevents_summary_values)`: Acess values from `summary_values`.

Examples

```
temp <- tempfile()
with_logdir(temp, {
  for(i in 1:5) {
    log_event(my_log = runif(1))
  }
})

# iterate over all events
summary <- collect_events(temp, n = 1, type = "summary")
value(summary$summary)
```

Index