

Package ‘tensorflow’

August 19, 2021

Type Package

Title R Interface to 'TensorFlow'

Version 2.6.0

Description Interface to 'TensorFlow' <<https://www.tensorflow.org/>>, an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more 'CPUs' or 'GPUs' in a desktop, server, or mobile device with a single 'API'. 'TensorFlow' was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

License Apache License 2.0

URL <https://github.com/rstudio/tensorflow>

BugReports <https://github.com/rstudio/tensorflow/issues>

SystemRequirements TensorFlow (<https://www.tensorflow.org/>)

Encoding UTF-8

Depends R (>= 3.1)

Imports config, processx, reticulate (>= 1.10), tfruns (>= 1.0),
utils, yaml, tfautograph (>= 0.3), rstudioapi (>= 0.7)

Suggests testthat (>= 2.1.0), keras, callr

RoxygenNote 7.1.1

NeedsCompilation no

Author Daniel Falbel [ctb, cph],
JJ Allaire [aut, cph],
RStudio [cph, fnd],
Yuan Tang [aut, cph] (<<https://orcid.org/0000-0001-5243-233X>>),
Dirk Eddelbuettel [ctb, cph],

Nick Golding [ctb, cph],
 Tomasz Kalinowski [ctb, cph, cre],
 Google Inc. [ctb, cph] (Examples and Tutorials)

Maintainer Tomasz Kalinowski <tomasz.kalinowski@rstudio.com>

Repository CRAN

Date/Publication 2021-08-19 18:50:02 UTC

R topics documented:

all_dims	2
as_tensor	3
evaluate	4
export_savedmodel	4
install_tensorflow	5
install_tensorflow_extras	7
parse_arguments	7
parse_flags	8
set_random_seed	9
shape	10
tensorboard	10
tensorflow	11
tf	12
tfe_enable_eager_execution	12
tf_extract_opts	13
tf_function	15
tf_probability	15
train	16
train_and_evaluate	16
use_compat	17
use_session_with_seed	17
view_savedmodel	18
[.tensorflow.tensor	19
Index	22

all_dims	<i>All dims</i>
----------	-----------------

Description

This function returns an object that can be used when subsetting tensors with `[]`. If you are familiar with python, this is equivalent to the python Ellipsis `...`, (not to be confused with `...` in R).

Usage

```
all_dims()
```

Examples

```
## Not run:
# in python, if x is a numpy array or tensorflow tensor
x[..., i]
# the ellipsis means "expand to match number of dimension of x".
# to translate the above python expression to R, write:
x[all_dims(), i]

## End(Not run)
```

as_tensor

as_tensor

Description

Coerce objects to tensorflow tensors (potentially of a specific dtype). The provided default methods will call `tf.convert_to_tensor` and `tf.cast` as appropriate.

Usage

```
as_tensor(x, dtype = NULL, ..., name = NULL)

## Default S3 method:
as_tensor(x, dtype = NULL, ..., name = NULL)

## S3 method for class 'double'
as_tensor(x, dtype = NULL, ..., name = NULL)
```

Arguments

x	object to convert
dtype	NULL, a tensorflow dtype (<code>tf\$int32</code>), or something coercible to one (e.g. a string <code>"int32"</code>)
...	ignored
name	NULL or a string. Useful for debugging in graph mode, ignored while in eager mode.

Value

a tensorflow tensor

Examples

```
## Not run:
as_tensor(42, "int32")
as_tensor(as_tensor(42))

## End(Not run)
```

`evaluate`*Evaluate a Model*

Description

Evaluate a model object. See implementations in the [keras](#) and [tfestimators](#) packages.

Usage

```
evaluate(object, ...)
```

Arguments

<code>object</code>	An evaluable R object.
<code>...</code>	Optional arguments passed on to implementing methods.

Implementations

- [keras](#)
- [tfestimators](#)

`export_savedmodel`*Export a Saved Model*

Description

Serialize a model to disk. See implementations in the [keras](#) and [tfestimators](#) packages.

Usage

```
export_savedmodel(object, export_dir_base, ...)
```

Arguments

<code>object</code>	An R object.
<code>export_dir_base</code>	A string containing a directory in which to export the SavedModel.
<code>...</code>	Optional arguments passed on to implementing methods.

Value

The path to the exported directory, as a string.

Implementations

- [keras](#)
- [tfestimators](#)

install_tensorflow *Install TensorFlow and its dependencies*

Description

install_tensorflow() installs just the tensorflow python package and it's direct dependencies. For a more complete installation that includes additional optional dependencies, use keras::install_keras().

Usage

```
install_tensorflow(
  method = c("auto", "virtualenv", "conda"),
  conda = "auto",
  version = "default",
  envname = NULL,
  extra_packages = NULL,
  restart_session = TRUE,
  conda_python_version = "3.7",
  ...,
  python_version = conda_python_version
)
```

Arguments

method	Installation method. By default, "auto" automatically finds a method that will work in the local environment. Change the default to force a specific installation method. Note that the "virtualenv" method is not available on Windows.
conda	The path to a conda executable. Use "auto" to allow reticulate to automatically find an appropriate conda binary. See Finding Conda for more details.
version	TensorFlow version to install. Valid values include: <ul style="list-style-type: none"> • "default" installs 2.6 • "release" installs the latest release version of tensorflow (which may be incompatible with the current version of the R package) • A version specification like "2.4" or "2.4.0". Note that if the patch version is not supplied, the latest patch release is installed (e.g., "2.4" today installs version "2.4.2") • nightly for the latest available nightly build. • To any specification, you can append "-cpu" to install the cpu version only of the package (e.g., "2.4-cpu") • The full URL or path to an installer binary or python *.whl file.
envname	The name, or full path, of the environment in which Python packages are to be installed. When NULL (the default), the active environment as set by the RETICULATE_PYTHON_ENV variable will be used; if that is unset, then the r-reticulate environment will be used.
extra_packages	Additional Python packages to install along with TensorFlow.

```
restart_session
    Restart R session after installing (note this will only occur within RStudio).
...
    other arguments passed to reticulate::conda_install() or reticulate::virtualenv_install(),
    depending on the method used.
python_version, conda_python_version
    the python version installed in the created conda environment. Ignored when
    attempting to install with a Python virtual environment.
```

Details

You may be prompted you if you want it to download and install miniconda if reticulate did not find a non-system installation of python. Miniconda is the recommended installation method for most users, as it ensures that the R python installation is isolated from other python installations. All python packages will by default be installed into a self-contained conda or venv environment named "r-reticulate". Note that "conda" is the only supported method on Windows.

Custom Installation

`install_tensorflow()` or `keras::install_keras()` isn't required to use tensorflow with the package. If you manually configure a python environment with the required dependencies, you can tell R to use it by pointing reticulate at it, commonly by setting an environment variable:

```
Sys.setenv("RETICULATE_PYTHON" = "~/path/to/python-env/bin/python")
```

Apple Silicon

Tensorflow on Apple Silicon is not officially supported by the tensorflow maintainers. It is known that there can be issues running the official Tensorflow package under Rosetta as well. Fortunately, for the time being Apple has published a custom version of Tensorflow compatible with M1 macs. Installation instructions can be found at: <https://developer.apple.com/metal/tensorflow-plugin/>. Please note that this is an experimental build of both python and tensorflow. After following the instructions provided by Apple, you can advise reticulate to use that python installation by placing the following in your `.Renv` file:

```
RETICULATE_PYTHON = "~/miniforge3/bin/python"
```

Additional Packages

If you wish to add additional PyPI packages to your Keras / TensorFlow environment you can either specify the packages in the `extra_packages` argument of `install_tensorflow()` or `install_keras()`, or alternatively install them into an existing environment using the `reticulate::py_install()` function. Note that `install_keras()` includes a set of additional python packages by default, see `?keras::install_keras` for details.

See Also

```
keras::install_keras()
```

`install_tensorflow_extras`*(Defunct) Install additional Python packages alongside TensorFlow*

Description

This function is deprecated. Use the `extra_packages` argument to `install_tensorflow()` or `reticulate::py_install()` to install additional packages.

Usage

```
install_tensorflow_extras(packages, conda = "auto")
```

Arguments

<code>packages</code>	Python packages to install
<code>conda</code>	Path to conda executable (or "auto" to find conda using the PATH and other conventional install locations). Only used when TensorFlow is installed within a conda environment.

`parse_arguments`*Parse Command Line Arguments*

Description

Parse command line arguments of the form `--key=value` and `--key value`. The values are assumed to be valid `yaml` and will be converted using `yaml.load()`.

Usage

```
parse_arguments(arguments = NULL)
```

Arguments

<code>arguments</code>	A vector of command line arguments. When <code>NULL</code> (the default), the command line arguments received by the current R process are used.
------------------------	--

 parse_flags

Parse Configuration Flags for a TensorFlow Application

Description

Parse configuration flags for a TensorFlow application. Use this to parse and unify the configuration(s) specified through a `flags.yml` configuration file, alongside other arguments set through the command line.

Usage

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

Arguments

<code>config</code>	The configuration to use. Defaults to the active configuration for the current environment (as specified by the <code>R_CONFIG_ACTIVE</code> environment variable), or default when unset.
<code>file</code>	The configuration file to read.
<code>arguments</code>	The command line arguments (as a character vector) to be parsed.

Value

A named R list, mapping configuration keys to values.

Examples

```
## Not run:
# examine an example configuration file provided by tensorflow
file <- system.file("examples/config/flags.yml", package = "tensorflow")
cat(readLines(file), sep = "\n")

# read the default configuration
FLAGS <- tensorflow::parse_flags("default", file = file)
str(FLAGS)

# read the alternate configuration: note that
# the default configuration is inherited, but
# we override the 'string' configuration here
FLAGS <- tensorflow::parse_flags("alternate", file = file)
str(FLAGS)

# override configuration values using command
# line arguments (normally, these would be
```



```

# passed in through the command line invocation
# used to start the process)
FLAGS <- tensorflow::parse_flags(
  "alternate",
  file = file,
  arguments = c("--foo=1")
)
str(FLAGS)

## End(Not run)

```

set_random_seed	<i>Set random seed for TensorFlow</i>
-----------------	---------------------------------------

Description

Sets all random seeds needed to make TensorFlow code reproducible.

Usage

```
set_random_seed(seed, disable_gpu = TRUE)
```

Arguments

seed	A single value, interpreted as an integer
disable_gpu	TRUE to disable GPU execution (see <i>Parallelism</i> below).

Details

This function should be used instead of `use_session_with_seed()` if you are using TensorFlow ≥ 2.0 , as the concept of session doesn't really make sense anymore.

This functions sets:

- The R random seed with `set.seed()`.
- The python and Numpy seeds via (`reticulate::py_set_seed()`).
- The TensorFlow seed with (`tf$random$set_seed()`)

It also optionally disables the GPU execution as this is a potential source of non-reproducibility.

shape	<i>Tensor shape</i>
-------	---------------------

Description

Tensor shape

Usage

```
shape(...)
```

Arguments

... Tensor dimensions

tensorboard	<i>TensorBoard Visualization Tool</i>
-------------	---------------------------------------

Description

TensorBoard is a tool inspecting and understanding your TensorFlow runs and graphs.

Usage

```
tensorboard(
  log_dir,
  action = c("start", "stop"),
  host = "127.0.0.1",
  port = "auto",
  launch_browser = getOption("tensorflow.tensorboard.browser", interactive()),
  reload_interval = 5,
  purge_orphaned_data = TRUE
)
```

Arguments

log_dir	Directories to scan for training logs. If this is a named character vector then the specified names will be used as aliases within TensorBoard.
action	Specify whether to start or stop TensorBoard (TensorBoard will be stopped automatically when the R session from which it is launched is terminated).
host	Host for serving TensorBoard
port	Port for serving TensorBoard. If "auto" is specified (the default) then an unused port will be chosen automatically.

- `launch_browser` Open a web browser for TensorBoard after launching. Defaults to TRUE in interactive sessions. When running under RStudio uses an RStudio window by default (pass a function e.g. `utils::browseURL()` to open in an external browser). Use the `tensorflow.tensorboard.browser` option to establish a global default behavior.
- `reload_interval` How often the backend should load more data.
- `purge_orphaned_data` Whether to purge data that may have been orphaned due to TensorBoard restarts. Disabling `purge_orphaned_data` can be used to debug data disappearance.

Details

When TensorBoard is passed a `logdir` at startup, it recursively walks the directory tree rooted at `logdir` looking for subdirectories that contain tfevents data. Every time it encounters such a subdirectory, it loads it as a new run, and the frontend will organize the data accordingly.

The TensorBoard process will be automatically destroyed when the R session in which it is launched exits. You can pass `action = "stop"` to manually terminate TensorBoard.

Value

URL for browsing TensorBoard (invisibly).

tensorflow

TensorFlow for R

Description

TensorFlow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API.

Details

The **TensorFlow API** is composed of a set of Python modules that enable constructing and executing TensorFlow graphs. The `tensorflow` package provides access to the complete TensorFlow API from within R.

For additional documentation on the `tensorflow` package see <https://tensorflow.rstudio.com>

tf *Main TensorFlow module*

Description

Interface to main TensorFlow module. Provides access to top level classes and functions as well as sub-modules (e.g. tf\$nn, tf\$contrib\$learn, etc.).

Usage

```
tf
```

Format

TensorFlow module

Examples

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

hello <- tf$constant('Hello, TensorFlow!')
zeros <- tf$Variable(tf$zeros(shape(1L)))

tf$print(hello)
tf$print(zeros)

## End(Not run)
```

tfe_enable_eager_execution

(Deprecated) Enables, for the rest of the lifetime of this program, eager execution.

Description

This function is no longer needed since Tensorflow 2.0, when eager execution became the default.

Usage

```
tfe_enable_eager_execution(
  config = NULL,
  device_policy = c("explicit", "warn", "silent")
)
```

Arguments

- `config` (Optional) A `tf$ConfigProto()` protocol buffer with configuration options for the Context. Note that a lot of these options may be currently unimplemented or irrelevant when eager execution is enabled.
- `device_policy` (Optional) What policy to use when trying to run an operation on a device with inputs which are not on that device. Valid values: "explicit": raises an error if the placement is not correct. "warn": copies the tensors which are not on the right device but raises a warning. "silent": silently copies the tensors. This might hide performance problems.

Details

If not called immediately on startup risks creating breakage and bugs.

After eager execution is enabled, operations are executed as they are defined and tensors hold concrete values, and can be accessed as R matrices or arrays with `as.matrix()`, `as.array()`, `as.double()`, etc.

Examples

```
## Not run:

# load tensorflow and enable eager execution
library(tensorflow)
tfe_enable_eager_execution()

# create a random 10x10 matrix
x <- tf$random$normal(shape(10, 10))

# use it in R via as.matrix()
heatmap(as.matrix(x))

## End(Not run)
```

tf_extract_opts	<i>Tensor extract options</i>
-----------------	-------------------------------

Description

Tensor extract options

Usage

```
tf_extract_opts(
  style = getOption("tensorflow.extract.style"),
  ...,
  one_based = getOption("tensorflow.extract.one_based", TRUE),
```

```

    inclusive_stop = getOption("tensorflow.extract.inclusive_stop", TRUE),
    disallow_out_of_bounds = getOption("tensorflow.extract.dissallow_out_of_bounds",
    TRUE),
    warn_tensors_passed_asis = getOption("tensorflow.extract.warn_tensors_passed_asis",
    TRUE),
    warn_negatives_pythonic = getOption("tensorflow.extract.warn_negatives_pythonic",
    TRUE)
  )

```

Arguments

style	one of NULL (the default) "R" or "python". If supplied, this overrides all other options. "python" is equivalent to all the other arguments being FALSE. "R" is equivalent to warn_tensors_passed_asis and warn_negatives_pythonic set to FALSE
...	ignored
one_based	TRUE or FALSE, if one-based indexing should be used
inclusive_stop	TRUE or FALSE, if slices like start:stop should be inclusive of stop
disallow_out_of_bounds	TRUE or FALSE, whether checks are performed on the slicing index to ensure it is within bounds.
warn_tensors_passed_asis	TRUE or FALSE, whether to emit a warning the first time a tensor is supplied to [that tensors are passed as-is, with no R to python translation
warn_negatives_pythonic	TRUE or FALSE, whether to emit a warning the first time a negative number is supplied to [about the non-standard (python-style) interpretation

Value

an object with class "tf_extract_opts", suitable for passing to [.tensorflow.tensor()

Examples

```

## Not run:
x <- tf$constant(1:10)

opts <- tf_extract_opts("R")
x[1, options = opts]

# or for more fine-grained control
opts <- tf_extract_opts(
  one_based = FALSE,
  warn_tensors_passed_asis = FALSE,
  warn_negatives_pythonic = FALSE
)
x[0:2, options = opts]

## End(Not run)

```

tf_function	<i>Creates a callable TensorFlow graph from an R function.</i>
-------------	--

Description

tf_function constructs a callable that executes a TensorFlow graph created by tracing the TensorFlow operations in f. This allows the TensorFlow runtime to apply optimizations and exploit parallelism in the computation defined by f.

Usage

```
tf_function(f, input_signature = NULL, autograph = TRUE, ...)
```

Arguments

f	the function to be compiled
input_signature	A possibly nested sequence of tf\$TensorSpec objects specifying the shapes and dtypes of the tensors that will be supplied to this function. If NULL, a separate function is instantiated for each inferred input signature. If input_signature is specified, every input to f must be a tensor.
autograph	TRUE or FALSE. If TRUE (the default), you can use tensors in R control flow expressions if, while, for and break and they will be traced into the tensorflow graph. A guide to getting started and additional details can be found: here
...	additional arguments passed on to tf.function (vary based on Tensorflow version). See here for details.

Details

A guide to getting started with [tf.function](#) can be found [here](#).

tf_probability	<i>TensorFlow Probability Module</i>
----------------	--------------------------------------

Description

TensorFlow Probability Module

Usage

```
tf_probability()
```

Value

Reference to [TensorFlow Probability](#) functions and classes

Examples

```
## Not run:
library(tensorflow)
tfp <- tf_probability()
tfp$distributions$Normal(loc=0, scale=1)

## End(Not run)
```

train	<i>Train a Model</i>
-------	----------------------

Description

Train a model object. See implementation in the [tfestimators](#) package.

Usage

```
train(object, ...)
```

Arguments

object	A trainable R object.
...	Optional arguments passed on to implementing methods.

Implementations

- [tfestimators](#)

train_and_evaluate	<i>Simultaneously Train and Evaluate a Model</i>
--------------------	--

Description

Train and evaluate a model object. See implementation in the [tfestimators](#) package.

Usage

```
train_and_evaluate(object, ...)
```

Arguments

object	An R object.
...	Optional arguments passed on to implementing methods.

Implementations

- [tfestimators](#)

use_compat	<i>Use Compatibility</i>
------------	--------------------------

Description

Enables TensorFlow to run under a different API version for compatibility with previous versions. For instance, this is useful to run TensorFlow 1.x code when using TensorFlow 2.x.

Usage

```
use_compat(version = c("v1", "v2"))
```

Arguments

version The version to activate. Must be "v1" or "v2"

Examples

```
## Not run:  
library(tensorflow)  
use_compat("v1")  
  
## End(Not run)
```

use_session_with_seed	<i>Use a session with a random seed</i>
-----------------------	---

Description

Set various random seeds required to ensure reproducible results. The provided seed value will establish a new random seed for R, Python, NumPy, and TensorFlow. GPU computations and CPU parallelism will also be disabled by default.

Usage

```
use_session_with_seed(  
  seed,  
  disable_gpu = TRUE,  
  disable_parallel_cpu = TRUE,  
  quiet = FALSE  
)
```

Arguments

seed	A single value, interpreted as an integer
disable_gpu	TRUE to disable GPU execution (see <i>Parallelism</i> below).
disable_parallel_cpu	TRUE to disable CPU parallelism (see <i>Parallelism</i> below).
quiet	TRUE to suppress printing of messages.

Details

This function must be called at the very top of your script (i.e. immediately after `library(tensorflow)`, `library(keras)`, etc.). Any existing TensorFlow session is torn down via `tf$reset_default_graph()`.

This function takes all measures known to promote reproducible results from TensorFlow sessions, however it's possible that various individual TensorFlow features or dependent libraries escape its effects. If you encounter non-reproducible results please investigate the possible sources of the problem, contributions via pull request are very welcome!

Packages which need to be notified before and after the seed is set can register for the "tensorflow.on_before_use_session" and "tensorflow.on_use_session" hooks (see [setHook\(\)](#) for additional details on hooks).

Value

TensorFlow session object, invisibly

Parallelism

By default the `use_session_with_seed()` function disables GPU and CPU parallelism, since both can result in non-deterministic execution patterns (see <https://stackoverflow.com/questions/42022950/>). You can optionally enable GPU or CPU parallelism by setting the `disable_gpu` and/or `disable_parallel_cpu` parameters to `FALSE`.

Examples

```
## Not run:
library(tensorflow)
use_session_with_seed(42)

## End(Not run)
```

view_savedmodel

View a Saved Model

Description

View a serialized model from disk.

Usage

```
view_savedmodel(model_dir)
```

Arguments

model_dir The path to the exported model, as a string.

Value

URL for browsing TensorBoard (invisibly).

[.tensorflow.tensor *Subset tensors with* [

Description

Subset tensors with [

Usage

```
## S3 method for class 'tensorflow.tensor'

x[
  ...,
  drop = TRUE,
  style = getOption("tensorflow.extract.style"),
  options = tf_extract_opts(style)
]
```

Arguments

x Tensorflow tensor
 ... slicing specs. See examples and details.
 drop whether to drop scalar dimensions
 style One of "python" or "R".
 options An object returned by tf_extract_opts()

Examples

```
## Not run:
sess <- tf$Session()

x <- tf$constant(1:15, shape = c(3, 5))
sess$run(x)
# by default, numerics supplied to `...` are interpreted R style
sess$run( x[,1] )# first column
sess$run( x[1:2,] ) # first two rows
```

```

sess$run( x[,1, drop = FALSE] )

# strided steps can be specified in R syntax or python syntax
sess$run( x[, seq(1, 5, by = 2)] )
sess$run( x[, 1:5:2] )
# if you are unfamiliar with python-style strided steps, see:
# https://docs.scipy.org/doc/numpy-1.13.0/reference/arrays.indexing.html#basic-slicing-and-indexing

# missing arguments for python syntax are valid, but they must be backticked
# or supplied as NULL
sess$run( x[, `::2`] )
sess$run( x[, NULL:NULL:2] )
sess$run( x[, `2:`] )

# Another python feature that is available is a python style ellipsis `...`
# (not to be confused with R dots `...`)
# a all_dims() expands to the shape of the tensor
y <- tf$constant(1:(3^5), shape = c(3,3,3,3,3))
identical(
  sess$run( y[all_dims(), 1] ),
  sess$run( y[,,,1] )
)

# tf$newaxis are valid
sess$run( x[,, tf$newaxis] )

# negative numbers are always interpreted python style
# The first time a negative number is supplied to `[`, a warning is issued
# about the non-standard behavior.
sess$run( x[-1,] ) # last row, with a warning
sess$run( x[-1,] ) # the warning is only issued once

# specifying `style = 'python'` changes the following:
# + zero-based indexing is used
# + slice sequences in the form of `start:stop` do not include `stop`
#   in the returned value
# + out-of-bounds indices in a slice are valid

# The style argument can be supplied to individual calls of `[` or set
# as a global option

# example of zero based indexing
sess$run( x[0, , style = 'python'] ) # first row
sess$run( x[1, , style = 'python'] ) # second row

# example of slices with exclusive stop
options(tensorflow.extract.style = 'python')
sess$run( x[, 0:1] ) # just the first column
sess$run( x[, 0:2] ) # first and second column

# example of out-of-bounds index
sess$run( x[, 0:10] )
options(tensorflow.extract.style = NULL)

```

```
# slicing with tensors is valid too, but note, tensors are never
# translated and are always interpreted python-style.
# A warning is issued the first time a tensor is passed to `[`
sess$run( x[, tf$constant(0L):tf$constant(2L)] )
# just as in python, only scalar tensors are valid
# https://www.tensorflow.org/api\_docs/python/tf/Tensor#\_\_getitem\_\_

# To silence the warnings about tensors being passed as-is and negative numbers
# being interpreted python-style, set
options(tensorflow.extract.style = 'R')

# clean up from examples
options(tensorflow.extract.style = NULL)

## End(Not run)
```

Index

- * **datasets**
 - tf, [12](#)
 - [.tensorflow.tensor, [19](#)
- all_dims, [2](#)
- as.array(), [13](#)
- as.double(), [13](#)
- as.matrix(), [13](#)
- as_tensor, [3](#)
- evaluate, [4](#)
- export_savedmodel, [4](#)
- install_tensorflow, [5](#)
- install_tensorflow_extras, [7](#)
- keras, [4](#)
- parse_arguments, [7](#)
- parse_flags, [8](#)
- reticulate::conda_install(), [6](#)
- reticulate::py_install(), [6](#)
- reticulate::py_set_seed(), [9](#)
- reticulate::virtualenv_install(), [6](#)
- set.seed(), [9](#)
- set_random_seed, [9](#)
- setHook(), [18](#)
- shape, [10](#)
- tensorboard, [10](#)
- tensorflow, [11](#)
- tf, [12](#)
- tf_extract_opts, [13](#)
- tf_function, [15](#)
- tf_probability, [15](#)
- tfe_enable_eager_execution, [12](#)
- tfestimators, [4](#), [16](#)
- train, [16](#)
- train_and_evaluate, [16](#)
- use_compat, [17](#)
- use_session_with_seed, [17](#)
- use_session_with_seed(), [9](#)
- utils::browseURL(), [11](#)
- view_savedmodel, [18](#)
- yaml.load(), [7](#)