Package 'tensorflow'

July 27, 2017

Type Package

Title R Interface to 'TensorFlow'

Version 1.3

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

LazyData true

Depends R (>= 3.0)

Imports config, jsonlite, processx, reticulate (>= 1.0), tfruns, utils, yaml

Suggests testthat

RoxygenNote 6.0.1

NeedsCompilation no

Author JJ Allaire [aut, cre], RStudio [cph, fnd], Yuan Tang [aut, cph], Dirk Eddelbuettel [ctb, cph],

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

Maintainer JJ Allaire <jj@rstudio.com>

Repository CRAN

Date/Publication 2017-07-27 21:42:18 UTC

R topics documented:

`install_tensorflow` *Install TensorFlow and it's dependencies*

Description

Install TensorFlow and it's dependencies

Usage

```
install_tensorflow(method = c("auto", "virtualenv", "conda", "system"),
  version = "latest", gpu = FALSE, package_url = NULL, conda = "auto")
```

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 (as this isn't supported by TensorFlow). Note also that since this command runs without privillege the "system" method is available only on Windows.
`version`	TensorFlow version to install (must be either "latest" or a full major.minor.patch specification, e.g. "1.1.0").
`gpu`	Install the GPU version of TensorFlow
`package_url`	URL of the TensorFlow package to install (if not specified this is determined automatically). Note that if this parameter is provied then the `version` and `gpu` parameters are ignored.

conda	Path to conda executable (or "auto" to find conda using the PATH and other conventional install locations).

`install_tensorflow_extras` *Install additional Python packages alongside TensorFlow*

Description

Install additional Python packages alongside TensorFlow

Usage

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

Arguments

packages	Python packages to install
conda	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.

Details

This function requires a version of TensorFlow previously installed via the `install_tensorflow()` function.

For virtualenv and conda installations, the specified packages will be installed into the "r-tensorflow" environment. For system installations on Windows the specified packages will be installed into the system package library.

`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

arguments	A vector of command line arguments. When `NULL` (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

`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 configuration file to read.
`arguments`	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)
```

---

shape *Tensor shape*

Description

Tensor shape

Usage

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

Arguments

...	Tensor dimensions

---

tensorboard *TensorBoard Visualization Tool*

Description

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

Usage

```
tensorboard(log_dir = NULL, action = c("start", "stop"),
  host = "127.0.0.1", port = "auto", launch_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. The default is NULL, which will result in the active run_dir() (if available) and otherwise will use the current working directory.
action	Specify whether to start or stop TensorBoard for the given log_dir (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` `TRUE` to open a web browser for TensorBoard after launching.

`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)))

sess <- tf$Session()
sess$run(tf$global_variables_initializer())

sess$run(hello)
sess$run(zeros)

## End(Not run)
```

tf_imperative *Imperative style TensorFlow*

Description

The results of the computation are available right after the execution of a line of code. If the return value is Tensor, it can be converted to base R object automatically if `convert = TRUE` is specified. Users can also call `tensor$eval()` to convert any Tensor objects inside the `expr` scope to base R objects.

Usage

```
tf_imperative(expr, new_step = FALSE, convert = TRUE)
```

Arguments

expr	A block of code expession
new_step	A boolean indicating whether the expression is evaluated as a new step. A graph is constructed and kept around in the background, both for just executing using the standard TensorFlow runtime, and also for allowing automatic differentiation via `tf$gradients`. If this is set to `TRUE`, the graph as well as the cached tensors that have been kept around for gradient computation will be cleared after the expression is evaluated.
convert	A boolean indicating whether to convert the returned Tensor object to base R object.

Details

Note that this function is currently only experimental, meaning that the interface is subject to change or remove at later releases.

Examples

```
## Not run:

tf_imperative({
  a <- tf$constant(list(list(7), list(6)))
  b <- tf$constant(list(list(6, 7)))
  list(
   tf$matmul(a, b),
   a * 4
  )
})

# This is equivalent to the following:

tf <- tf$contrib$imperative
a <- tf$constant(list(list(7), list(6)))
b <- tf$constant(list(list(6, 7)))
res1 <- tf$matmul(a, b)
res2 <- a * 4
list(res1$eval(), res2$eval())


## End(Not run)
```

Index