## Package ‘tfdeploy’

### June 14, 2019

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Deploy 'TensorFlow' Models</td>
</tr>
<tr>
<td>Version</td>
<td>0.6.1</td>
</tr>
<tr>
<td>Maintainer</td>
<td>Daniel Falbel <a href="mailto:daniel@rstudio.com">daniel@rstudio.com</a></td>
</tr>
<tr>
<td>Description</td>
<td>Tools to deploy 'TensorFlow' <a href="https://www.tensorflow.org/">https://www.tensorflow.org/</a> models across multiple services. Currently, it provides a local server for testing 'cloudml' compatible services.</td>
</tr>
<tr>
<td>License</td>
<td>Apache License 2.0</td>
</tr>
<tr>
<td>Encoding</td>
<td>UTF-8</td>
</tr>
<tr>
<td>LazyData</td>
<td>true</td>
</tr>
<tr>
<td>Imports</td>
<td>httpuv, htr, jsonlite, magrittr, reticulate, swagger, tensorflow</td>
</tr>
<tr>
<td>Suggests</td>
<td>cloudml, knitr, pixels, processx, testthat, yaml, stringr</td>
</tr>
<tr>
<td>RoxygenNote</td>
<td>6.1.1</td>
</tr>
<tr>
<td>VignetteBuilder</td>
<td>knitr</td>
</tr>
<tr>
<td>NeedsCompilation</td>
<td>no</td>
</tr>
<tr>
<td>Author</td>
<td>Javier Luraschi [aut, ctb], Daniel Falbel [cre, ctb], RStudio [cph]</td>
</tr>
<tr>
<td>Repository</td>
<td>CRAN</td>
</tr>
<tr>
<td>Date/Publication</td>
<td>2019-06-14 16:30:03 UTC</td>
</tr>
</tbody>
</table>

### R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>load_savedmodel</td>
<td>2</td>
</tr>
<tr>
<td>predict_savedmodel</td>
<td>3</td>
</tr>
<tr>
<td>predict_savedmodel.export_prediction</td>
<td>4</td>
</tr>
<tr>
<td>predict_savedmodel.graph_prediction</td>
<td>4</td>
</tr>
<tr>
<td>predict_savedmodel.webapi_prediction</td>
<td>5</td>
</tr>
<tr>
<td>serve_savedmodel</td>
<td>6</td>
</tr>
</tbody>
</table>

Index 7
load_savedmodel  

Load a SavedModel

Description

Loads a SavedModel using the given TensorFlow session and returns the model’s graph.

Usage

load_savedmodel(sess = NULL, model_dir = NULL)

Arguments

- **sess**: The TensorFlow session. NULL if using Eager execution.
- **model_dir**: The path to the exported model, as a string. Defaults to a "savedmodel" path or the latest training run.

Details

Loading a model improves performance over multiple `predict_savedmodel()` calls.

See Also

- `export_savedmodel()`, `predict_savedmodel()`

Examples

```r
## Not run:
# start session
sess <- tensorflow::tf$Session()

# preload an existing model into a TensorFlow session
graph <- tfdeploy::load_savedmodel(
  sess,
  system.file("models/tensorflow-mnist", package = "tfdeploy")
)

# perform prediction based on a pre-loaded model
tfdeploy::predict_savedmodel(
  list(rep(9, 784)),
  graph
)

# close session
sess$close()
```

## End(Not run)
**predict_savedmodel**

*Predict using a SavedModel*

**Description**

Runs a prediction over a saved model file, web API or graph object.

**Usage**

```r
predict_savedmodel(instances, model, ...)
```

**Arguments**

- **instances**: A list of prediction instances to be passed as input tensors to the service. Even for single predictions, a list with one entry is expected.
- **model**: The model as a local path, a REST url or graph object.
  A local path can be exported using `export_savedmodel()`, a REST URL can be created using `serve_savedmodel()` and a graph object loaded using `load_savedmodel()`.
  A type parameter can be specified to explicitly choose the type model performing the prediction. Valid values are export, webapi and graph.
- **...**: See `predict_savedmodel.export_prediction()`, `predict_savedmodel.graph_prediction()`, `predict_savedmodel.webapi_prediction()` for additional options.

```r
#' @section Implementations:
• predict_savedmodel.export_prediction()
• predict_savedmodel.graph_prediction()
• predict_savedmodel.webapi_prediction()
```

**See Also**

`export_savedmodel()`, `serve_savedmodel()`, `load_savedmodel()`

**Examples**

```r
## not run:
# perform prediction based on an existing model
tfdeploy::predict_savedmodel(
  list(rep(9, 784)),
  system.file("models/tensorflow-mnist", package = "tfdeploy")
)

## end(not run)
```
**predict_savedmodel.export_prediction**

*Predict using an Exported SavedModel*

**Description**

Performs a prediction using a locally exported SavedModel.

**Usage**

```r
## S3 method for class 'export_prediction'
predict_savedmodel(instances, model, signature_name = "serving_default", ...)
```

**Arguments**

- `instances`: A list of prediction instances to be passed as input tensors to the service. Even for single predictions, a list with one entry is expected.
- `model`: The model as a local path, a REST url or graph object. A local path can be exported using `export_savedmodel()`, a REST URL can be created using `serve_savedmodel()` and a graph object loaded using `load_savedmodel()`. A type parameter can be specified to explicitly choose the type model performing the prediction. Valid values are `export`, `webapi` and `graph`.
- `signature_name`: The named entry point to use in the model for prediction. See `predict_savedmodel.export_prediction()`, `predict_savedmodel.graph_prediction()`, `predict_savedmodel.webapi_prediction()` for additional options.

```r
# @section Implementations:
• predict_savedmodel.export_prediction()
• predict_savedmodel.graph_prediction()
• predict_savedmodel.webapi_prediction()
```

---

**predict_savedmodel.graph_prediction**

*Predict using a Loaded SavedModel*

**Description**

Performs a prediction using a SavedModel model already loaded using `load_savedmodel()`.

**Usage**

```r
## S3 method for class 'graph_prediction'
predict_savedmodel(instances, model, sess, signature_name = "serving_default", ...)
```
Arguments

instances A list of prediction instances to be passed as input tensors to the service. Even for single predictions, a list with one entry is expected.

model The model as a local path, a REST url or graph object. A local path can be exported using `export_savedmodel()` , a REST URL can be created using `serve_savedmodel()` and a graph object loaded using `load_savedmodel()` . A type parameter can be specified to explicitly choose the type model performing the prediction. Valid values are `export` , `webapi` and `graph` .

sess The active TensorFlow session.

signature_name The named entry point to use in the model for prediction.

... See `predict_savedmodel.export_prediction()` , `predict_savedmodel.graph_prediction()` , `predict_savedmodel.webapi_prediction()` for additional options.

#' @section Implementations:

• `predict_savedmodel.export_prediction()`
• `predict_savedmodel.graph_prediction()`
• `predict_savedmodel.webapi_prediction()`

predict_savedmodel.webapi_prediction

Predict using a Web API

Description

 Performs a prediction using a Web API providing a SavedModel.

Usage

## S3 method for class 'webapi_prediction'
predict_savedmodel(instances, model, ...)

Arguments

instances A list of prediction instances to be passed as input tensors to the service. Even for single predictions, a list with one entry is expected.

model The model as a local path, a REST url or graph object. A local path can be exported using `export_savedmodel()` , a REST URL can be created using `serve_savedmodel()` and a graph object loaded using `load_savedmodel()` . A type parameter can be specified to explicitly choose the type model performing the prediction. Valid values are `export` , `webapi` and `graph` .

... See `predict_savedmodel.export_prediction()` , `predict_savedmodel.graph_prediction()` , `predict_savedmodel.webapi_prediction()` for additional options.

#' @section Implementations:

• `predict_savedmodel.export_prediction()`
• `predict_savedmodel.graph_prediction()`
• `predict_savedmodel.webapi_prediction()`
Serve a SavedModel

Description

Serve a TensorFlow SavedModel as a local web api.

Usage

serve_savedmodel(model_dirL host = "127.0.0.1", port = 8089,
daemonized = FALSE, browse = !daemonized)

Arguments

model_dir The path to the exported model, as a string.
host Address to use to serve model, as a string.
port Port to use to serve model, as numeric.
daemonized Makes `httpuv` server daemonized so R interactive sessions are not blocked to handle requests. To terminate a daemonized server, call `httpuv::stopDaemonizedServer()` with the handle returned from this call.
browse Launch browser with serving landing page?

See Also

export_savedmodel()

Examples

## Not run:
# serve an existing model over a web interface
tfdeploy::serve_savedmodel(
  system.file("models/tensorflow-mnist", package = "tfdeploy")
)
## End(Not run)
Index

export_savedmodel(). 2, 3, 6
load_savedmodel. 2
load_savedmodel(). 3
predict_savedmodel. 3
predict_savedmodel(). 2
predict_savedmodel.export_prediction, 4
predict_savedmodel.export_prediction(). 3–5
predict_savedmodel.graph_prediction. 4
predict_savedmodel.graph_prediction(). 3–5
predict_savedmodel.webapi_prediction. 5
predict_savedmodel.webapi_prediction(). 3–5
serve_savedmodel. 6
serve_savedmodel(). 3