Statistical Learning

Getting started with Keras

FRC-EVL

Master's degree in Statistics and Operations Research, UPC- UB

Table of Contents

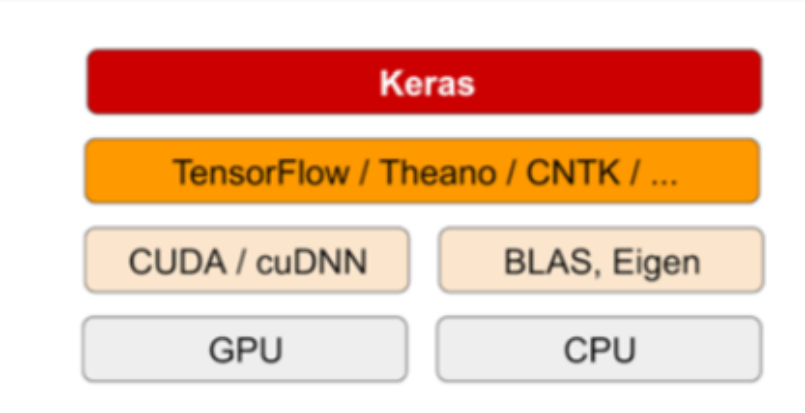
Introduction

Overview

Keras is an open-source high-level neural network API written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit (CNTK), Theano, or PlaidML. Its main features:

- Allows for easy and fast prototype of deep learning models.
- Supports arbitrary network architectures: multi-input or multi-output models, layer sharing, ...
- Runs seamlessly on CPU and GPU

For more information https://keras.io/



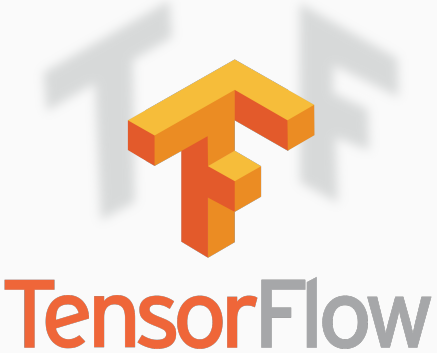
TensorFlow

TensorFlow is a lower level mathematical library for building deep neural network architecture using data flow graphs.

Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them.

TensorFlow was originally developed by researchers and engineers working on the Google Brain Team

For more information https://www.tensorflow.org/

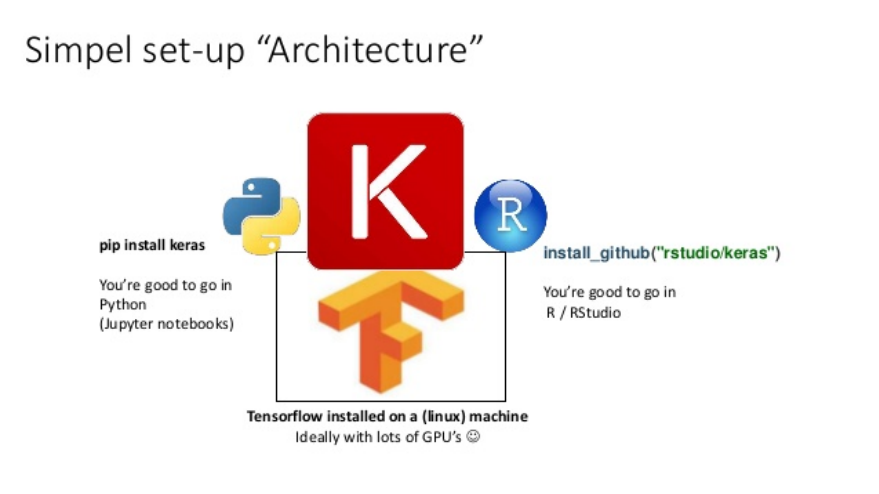


TensorFlow + Keras + R + Python

We work Keras + TensorFlow from R language so we need to have available python language too.

Anaconda distribution is a good option to have the python language.

For more information https://www.anaconda.com/



R interfaces with Keras

Installation

We need install keras package which uses the python keras library.

```
install.packages("keras")
library(keras)
```

Next, to install the TensorFlow backend use the function:

```
install_keras()
```

For more information https://tensorflow.rstudio.com/install/

Installation problems (1/2)

Solution to problems in interoperability between Python and R:

```
install_packages("reticulate")
```

Some usefull functions are:

```
py_config() #Python configuration
py_available() #Check if Python is available on this system
#Discover the version of Python to use with reticulate
py_discover_config()
py_discover_config("keras")
py_discover_config("tensorflow")

#Configure which version of Python to use
use_condaenv("base", conda = "auto" )
```

More details in https://rstudio.github.io/reticulate/

Installation problems (2/2)

If `install_keras()` function has errors when it try to install r-reticulate (or r-tensorflow) environment. You can the following steps from Anaconda distribution:

1. Create a Rstudio environment.
2. Go back to run `install_keras()` function from Rstudio into Anaconda distribution.

Python from Rstudio

Python from Rstudio

From Rstudio using reticulate package, you can compile Python code. Also, you can create dynamic reports similar to R markdown. Even, it is possible to create a script with R and Python code.

1. Support for executing reticulated Python chunks within R Notebooks.
2. Line-by-line execution of Python code using the reticulate repl_python() function.
3. Sourcing Python scripts using the reticulate source_python() function.
4. Code completion and inline help for Python.
5. Display of matplotlib plots within both notebook and console execution modes.

More details in https://rstudio.github.io/reticulate/articles/rstudio_ide.html

Deep Learning Workflow

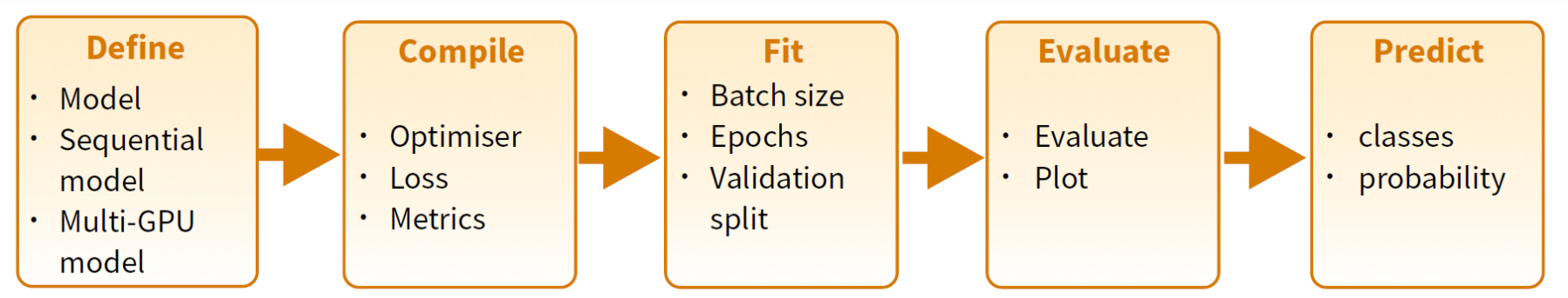
Keras Workflow (1/2)

This is the usual workflow with keras:

1. Define the training data
2. Define a neural network model
3. Configure the learning process
4. Train the model
5. Evaluate the model
6. Prediction new values

Keras Workflow (2/2)

This is a diagram of the process:



For more information https://www.rstudio.com/resources/cheatsheets/#keras

Examples

MNIST example

The MNIST database is a large database of handwritten digits that is commonly used for training various image processing systems. They are black and white images of 28×28 pixels.

Simple example

Objective: Vectorize images by rows, to get a 784-vector representation of each image. Implement a ANN with two hidden layers that uses such a vectors as input data for learning image categories, ten values from 0 to 9.