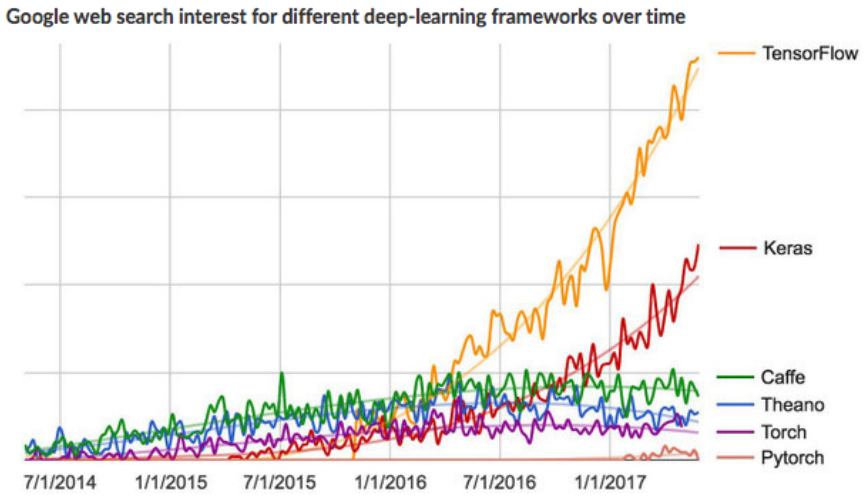
Deep Learning Lecture 1 - Keras Introduction

MA8701 General Statistical Methods

Thiago G. Martins, Department of Mathematical Sciences, NTNU

Spring 2019

We will use Keras R API to define and manipulate deep learning models.



Keras

Keras is a model-level library

- Provides high-level building blocks for developing deep-learning models.
- It does not handle low-level operations such as tensor manipulation and differentiation.

Backend engines

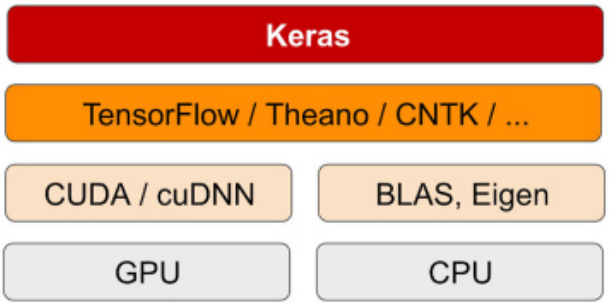
- Relies on a specialized and well-optimized tensor libraries to serve as backend engine of Keras.
- Supported backends: TensorFlow, Theano, and Microsoft Cognitive Toolkit (CNTK) backend.

CPU and GPU

- Via TensorFlow (or Theano, or CNTK), Keras is able to run seamlessly on both CPUs and GPUs.
- When running on CPU, TensorFlow is itself wrapping a low-level library for tensor operations, called Eigen.
- On GPU, TensorFlow wraps a library of well-optimized deep-learning operations called the NVIDIA CUDA Deep Neural Network library (cuDNN).

Deep Learning and GPUs

- Some applications will be excruciatingly slow on CPU, even a fast multicore CPU.
- Image processing with convolutional networks and sequence processing with recurrent neural networks
- For applications in general, speed increase by a factor of 5 or 10 by using a modern GPU.



Installing Keras

The steps below should be done only once within R console:

```
install.packages("keras") # Install R package Keras
library(keras)            # Load R package Keras
install_keras()           # Install Keras and dependencies
```

If you are running on a system with an NVIDIA GPU and properly configured CUDA and cuDNN libraries, you can install the GPU-based version of the TensorFlow backend engine as follows:

```
install_keras(tensorflow = "gpu")
```

Note: You do not need GPU to follow the lectures.

Typical Keras workflow

- Define your training data: input tensors and target tensors.
- Define a network of layers (or model) that maps your inputs to your targets.
- Configure the learning process by choosing a loss function, an optimizer, and some metrics to monitor.
- Iterate on your training data by calling the `fit()` method of your model.
- Use the trained model to make predictions, for example.

Reference material

This lecture note is based on (Chollet and Allaire 2018).

References

Chollet, F., and J. Allaire. 2018. *Deep Learning with R*. Manning Publications. https://books.google.no/books?id=xnIRtAEACAAJ.