Universität Augsburg
Fakultät für Angewandte Informatik

T6: Introduction to Keras (With TensorFlow)

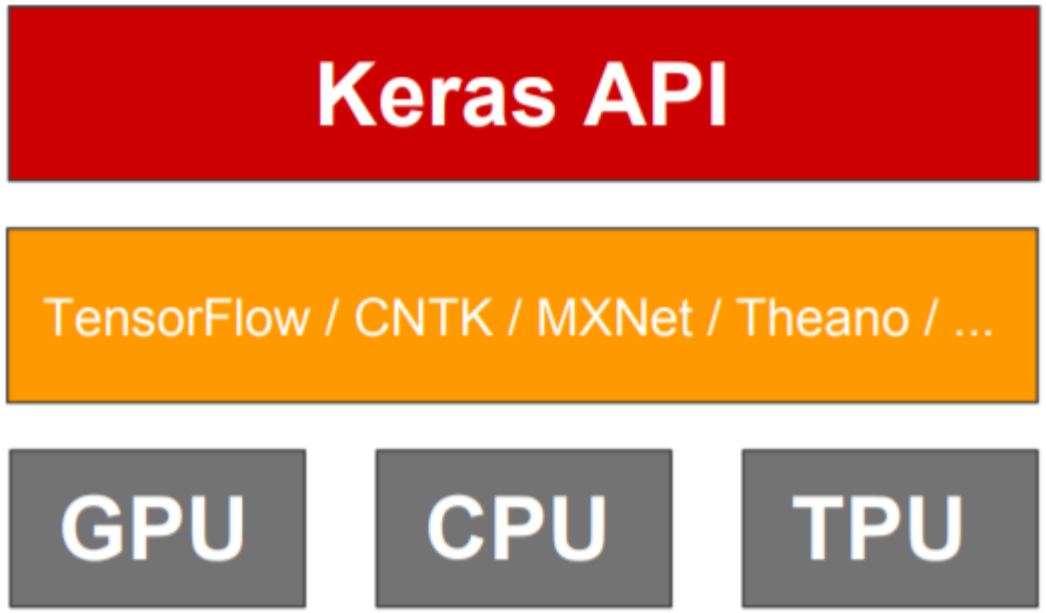
Thomas Wiest

Thomas.Wiest@informatik.uni-augsburg.de

Deep Learning – EIHW – University Augsburg

Keras & TensorFlow

How are they related?



TensorFlow

What is TensorFlow?

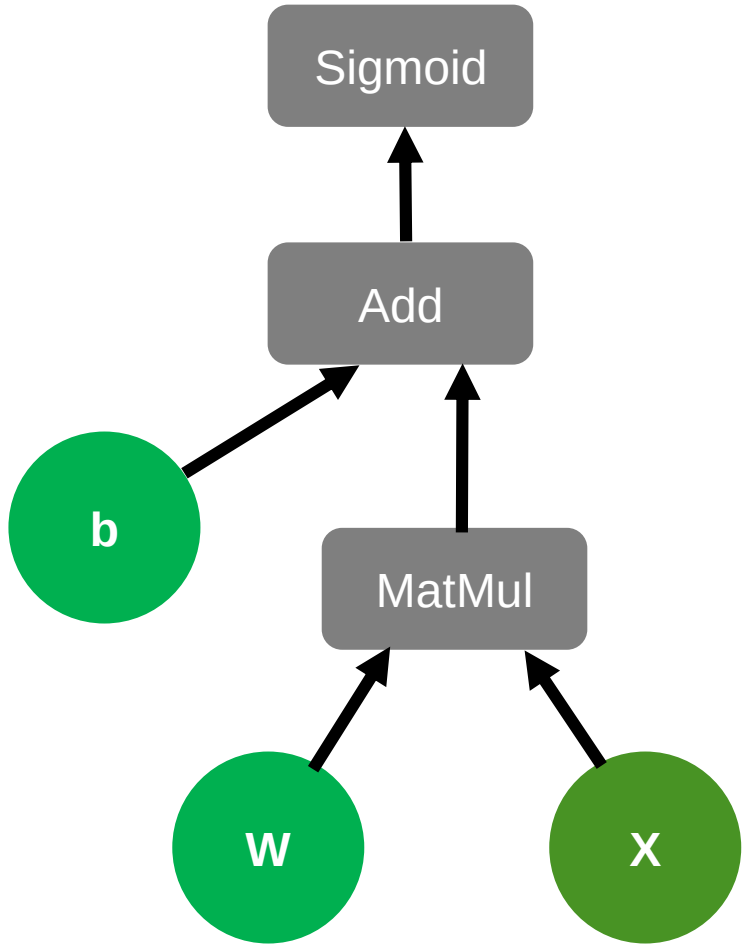
- Developed by Google
- One of the most popular libraries for implementing Machine Learning (and other) algorithms
- Provides primitives for defining functions on **tensors** and automatically computes their derivatives

TensorFlow

What is TensorFlow?

- **The core of TensorFlow is the Computational Graph**
  - Nodes are operations
  - Edges are Tensors

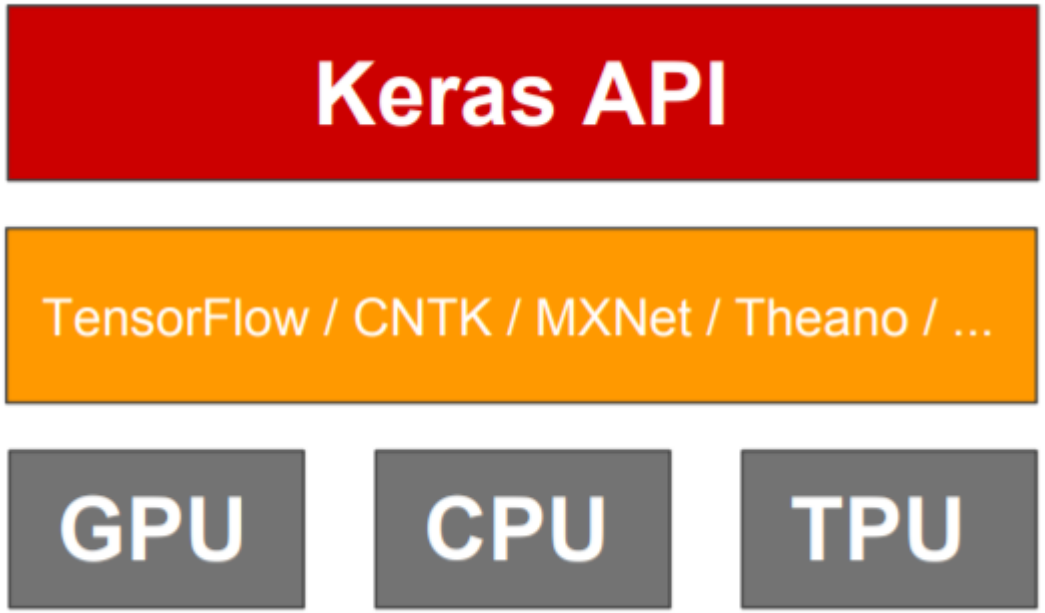
- **Typically a program is going through two phases**
  - Creating the **Computational Graph**
  - Pushing data through the graph



Keras

What is Keras?

- Official high-level API of TensorFlow
- Optimized for use with TensorFlow



Keras

Why Keras?

- **Simple API design**
  - Consistent and simple API
  - Reduces number of user interactions for common use cases

- **Easy to use while maintaining flexibility**
  - Fast prototyping due to simple code
  - Integrates TensorFlow Core API seamlessly

Keras

Who is behind Keras?

826 contributors

Keras

Programming models

- **Sequential Model**
  - Very simple
  - Single input, single output, sequential layer networks

- **Functional API**
  - Modular building block principle
  - Multi input, multi output, arbitrary graphs

- **Model subclassing**
  - Create own modules
  - High flexibility

Keras

Programming models

- **Sequential Model**
  - Very simple
  - Single input, single output, sequential layer networks

```
import keras
from keras.layers import Dense

model = keras.models.Sequential()
model.add(Dense(100, activation="relu", input_shape=(10,)))
model.add(Dense(100, activation="relu"))
model.add(Dense(10, activation="softmax"))

model.compile(...)
model.fit(...)
```

Keras

Programming models

- **Functional API**
  - Modular building block principle
  - Multi input, multi output, arbitrary graphs

```
import keras
from keras.layers import Dense

inputs = keras.Input((10,))
layer_1 = Dense(100, activation="relu")(inputs)
layer_2 = Dense(100, activation="relu")(layer_1)
output = Dense(10, activation="softmax")(layer_2)

model = Model(inputs=inputs, outputs=output)
model.compile(...)
model.fit(...)
```

Deep Learning

Online resources

- Becominghuman.ai
- Medium.com
- Towardsdatascience.com
- Hackeroon.com

https://keras.io/

Excercise

Introduction to Keras

- Advised to use Google Colaboratory with GPU support
  - Runtime -> Change Runtime Type -> GPU

- Exercise 2
  - Implement the same network as in Assignment 3 using the Sequential or the Functional API of Keras (using only Dense layers)

- Exercise 3
  - Implement a MNIST classifiers that uses at least 1 Convolutional Layer with a downsampling method and at most 2 Dense layers (Ît is possible to use only one)

- In both exercises use a train, validation and test partition of your dataset. Adjust your model using the train and validation set and perform your final evaluation on the test set.

Exercise

Programming

- **Hyperparameters**
  - Batch size
    - Size of the minibatch used for a training iteration; In real-world scenarios you can‘t just forward pass your whole dataset at once
  - Epochs
    - How often the whole dataset is passed for training; Too many epochs might result in overfitting, too few might result in the opposite
  - Optimizer
    - Adam, SGD, RMSprop
  - Learning Rate
    - Depends on problem and optimizer