Applied Deep Learning

Dr. Philippe Blaettchen
Bayes Business School (formerly Cass)

www.bayes.city.ac.uk

Learning objectives of today

Goals: Creating neural networks with Keras and TensorFlow

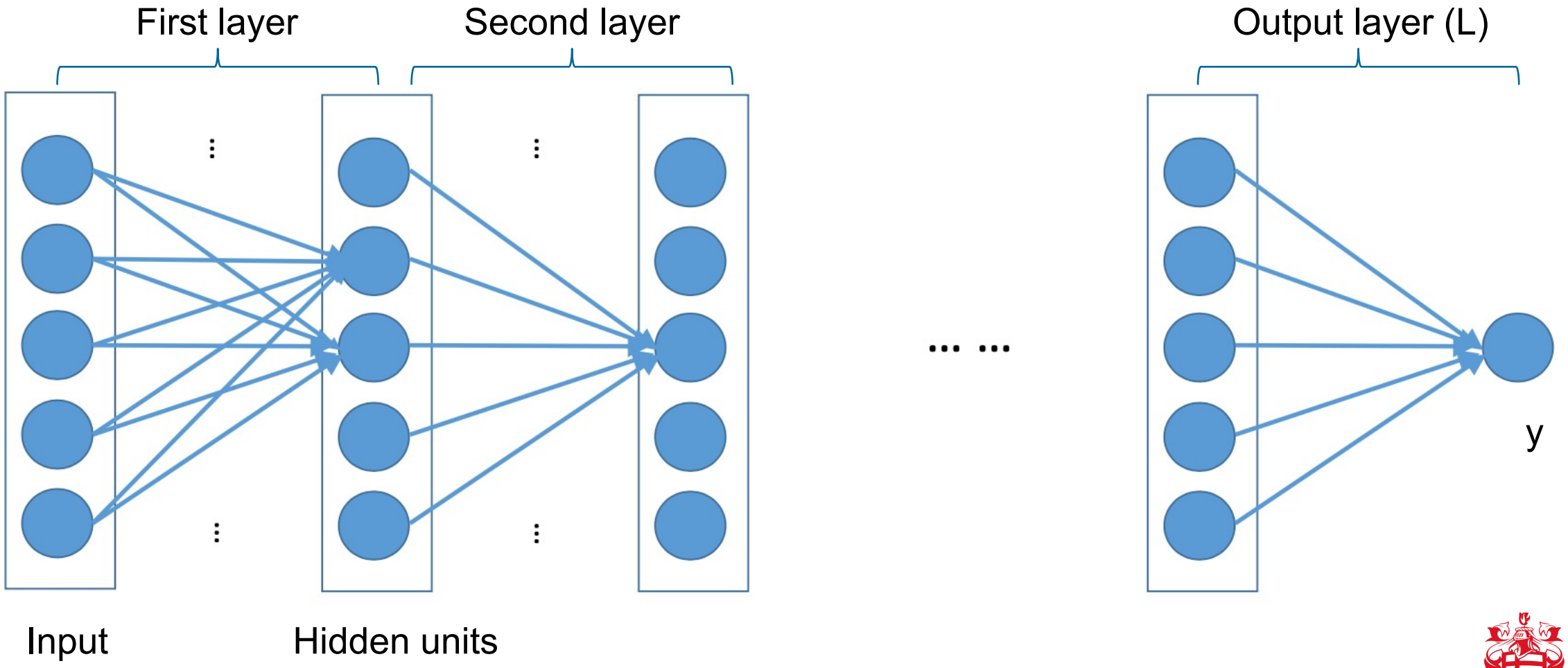
- Understanding the role that frameworks such as TensorFlow play in enabling the design, training, and use of arbitrarily complex neural networks
- Getting started on applying TensorFlow to create different types of feed-forward networks

How will we do this?

- We start with a quick recap of the functioning of neural networks
- We then introduce TensorFlow and Keras as programming frameworks for neural networks
- We create a number of neural networks, using TensorFlow's Sequential API

Recap

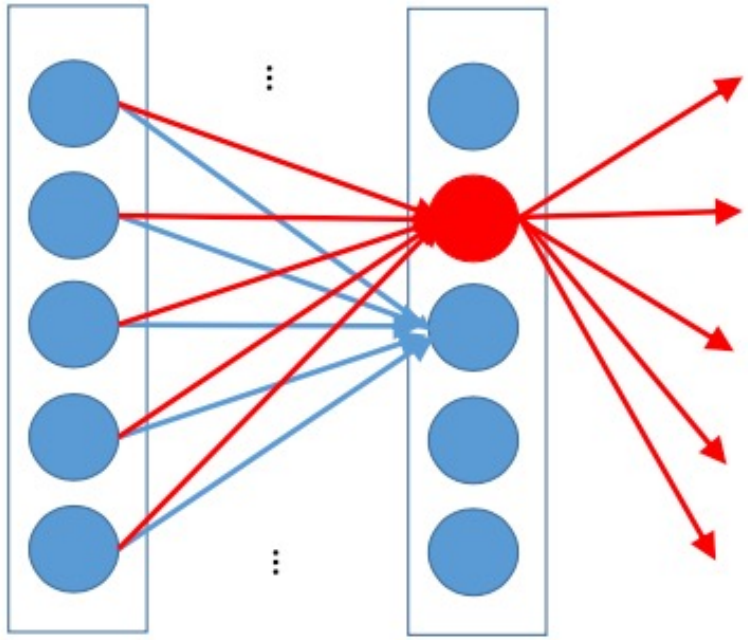
Components



Source: Liang

Hidden layers

$$\text{“}\boldsymbol{x}\text{”} = \boldsymbol{a}^{[l-1]} \qquad z = \boldsymbol{a}^{[\boldsymbol{l-1}]}\boldsymbol{w} + \boldsymbol{b} \qquad f(\mathrm{z})$$

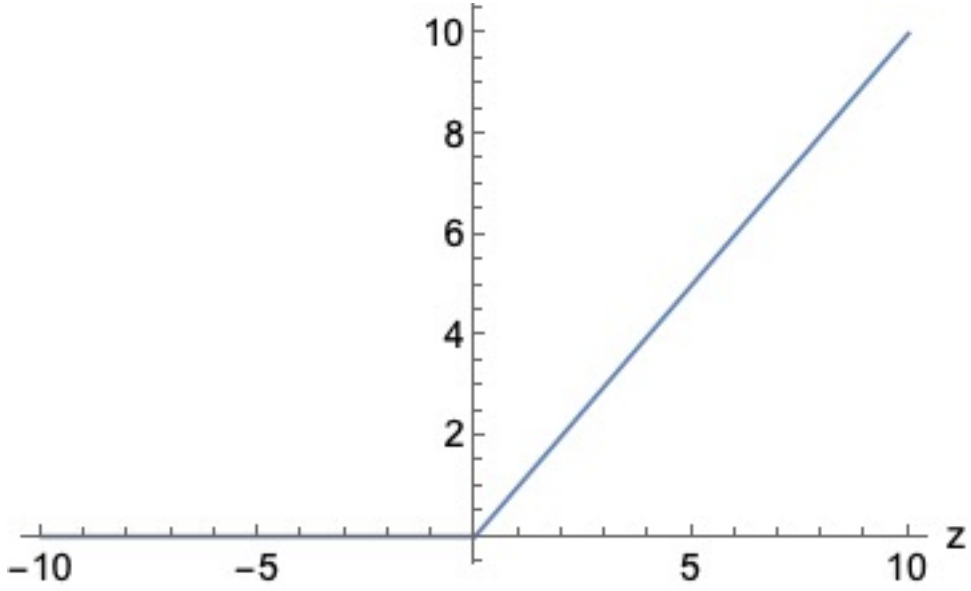


- f is what we call an “activation function”
- There are many activation functions, and new ones are invented all the time
- Many of these functions do just fine, or slightly better than existing ones

Source: Liang

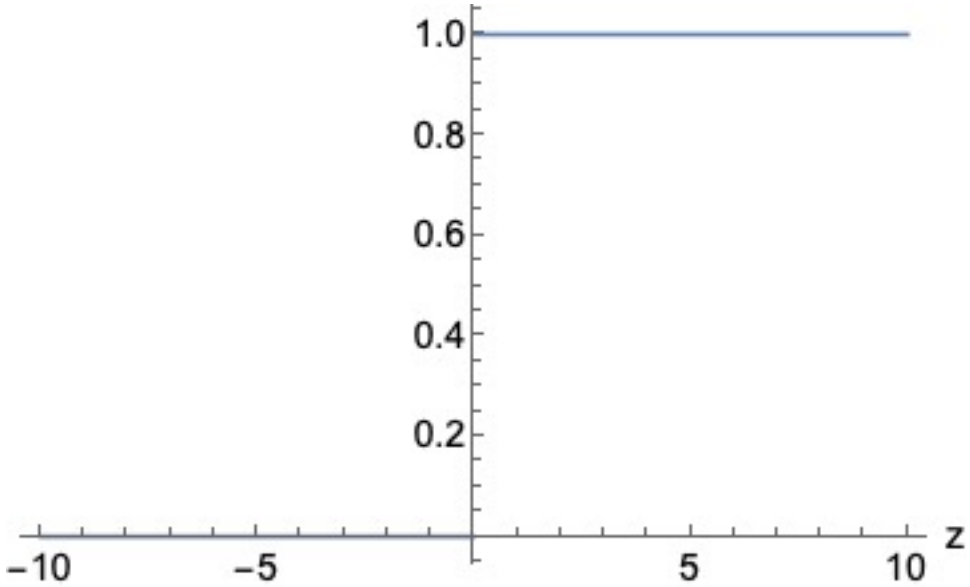
Typical activation functions: Rectified Linear Unit

Rectified Linear Unit (ReLU)



$$f(z) = \max\{0, z\}$$

“Derivative”

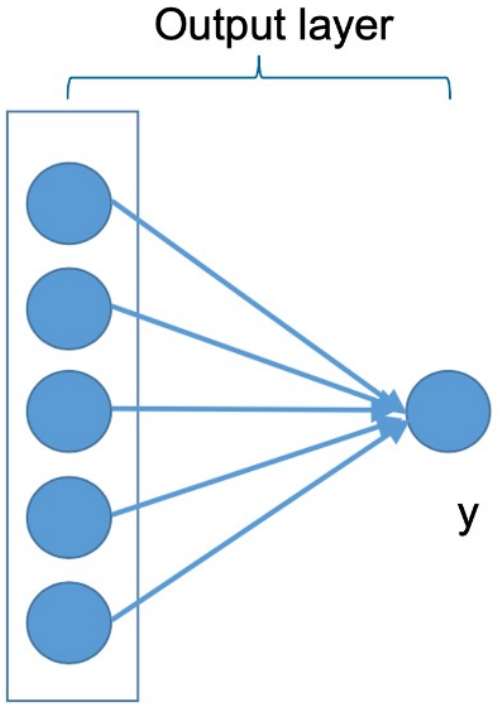


$$f'(z) = \begin{cases} 0, if\ z < 0 \\ 1, if\ z > 0 \end{cases}$$

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Binary classification

- Input: $\boldsymbol{a}^{[L-1](i)}$
- As usual, we make a linear transformation: $z^{[L](i)} = \boldsymbol{a}^{[L-1](i)}\boldsymbol{w}^{[L]} + b^{[L]}$
- We then use the logistic sigmoid function $\hat{y}^{(i)} = f\left(z^{[L](i)}\right) = \sigma(z^{[L](i)}) = \frac{1}{1+e^{-z^{[L](i)}}}$
- We can interpret the output as the probability of $y^{(i)} = 1$



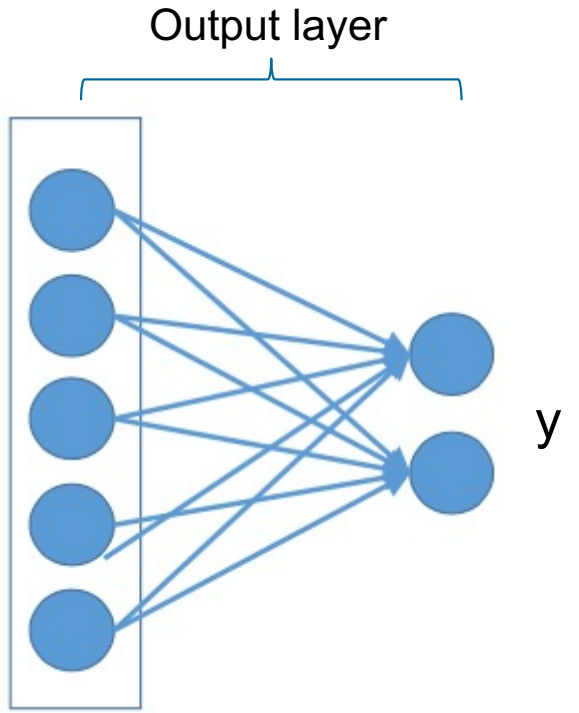
Source: Liang

Multi-class classification

- We again make a linear transformation with a matrix of weights: $\boldsymbol{z}^{[L](i)} = \boldsymbol{a}^{[L-1](i)}\boldsymbol{W}^{[L]} + \boldsymbol{b}^{[L]}$
- Note that $\boldsymbol{z}^{[L](i)} = \begin{pmatrix} z_1^{[L](i)} & z_2^{[L](i)} & \dots & z_K^{[L](i)} \end{pmatrix}$
- We then use the softmax function on each of the outputs:

$$\hat{y}_k^{(i)} = f\left(\boldsymbol{z}^{[L](i)}\right) = \frac{e^{-z_k^{[L](i)}}}{\sum_{k=1}^{K} e^{-z_k^{[L](i)}}}$$

- This implies that $\hat{y}_k^{(i)} \in (0,1)$ and $\sum_{k=1}^{K} \hat{y}_k^{(i)} = 1$
- Hence, we can interpret $\hat{y}_k^{(i)}$ as the probability that $y^{(i)} = k$ ("belongs to class k")

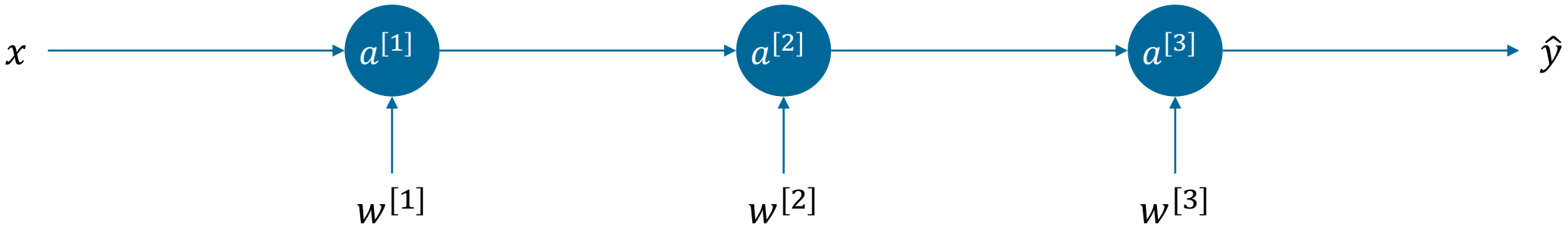


Source: Liang

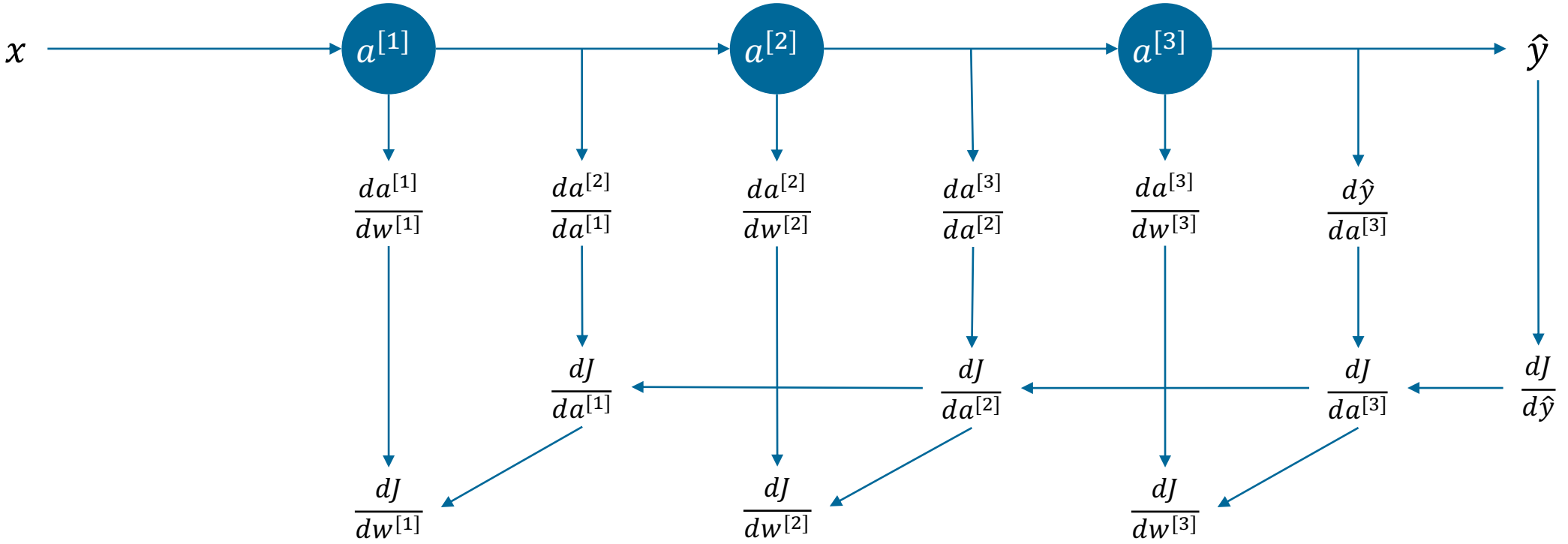
Generalizing our optimization algorithm

1. Decide a "learning rate" α
2. Start with some parameters $\boldsymbol{\theta}$ (initialization)
3. For a certain number of iterations
   - Compute $J(\boldsymbol{\theta})$ (forward propagation)
   - Let $\boldsymbol{\theta} := \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ (back-propagation)

Step 1: Forward propagation through the computational graph



Step 2: Back-propagation through the computational graph



BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Deep learning programming frameworks

Deep learning in Python: the two main rivals

TensorFlow

- Developed by Google
- Python API based on C engine

- Largest community
- Parallelization of models and data
- Visualization (TensorBoard)
- Relatively intuitive when familiar with numpy
- Runs on most systems (e.g., TensorFlow Lite)
- Broader framework (e.g., reinforcement learning)

- Can be slower than other frameworks

Pytorch

- Developed by Facebook
- Python API based on LUA engine (Torch)

- Many pre-trained models
- Highly modular
- Easy to run on GPUs
- Runs on most systems (e.g., PyTorch Mobile)

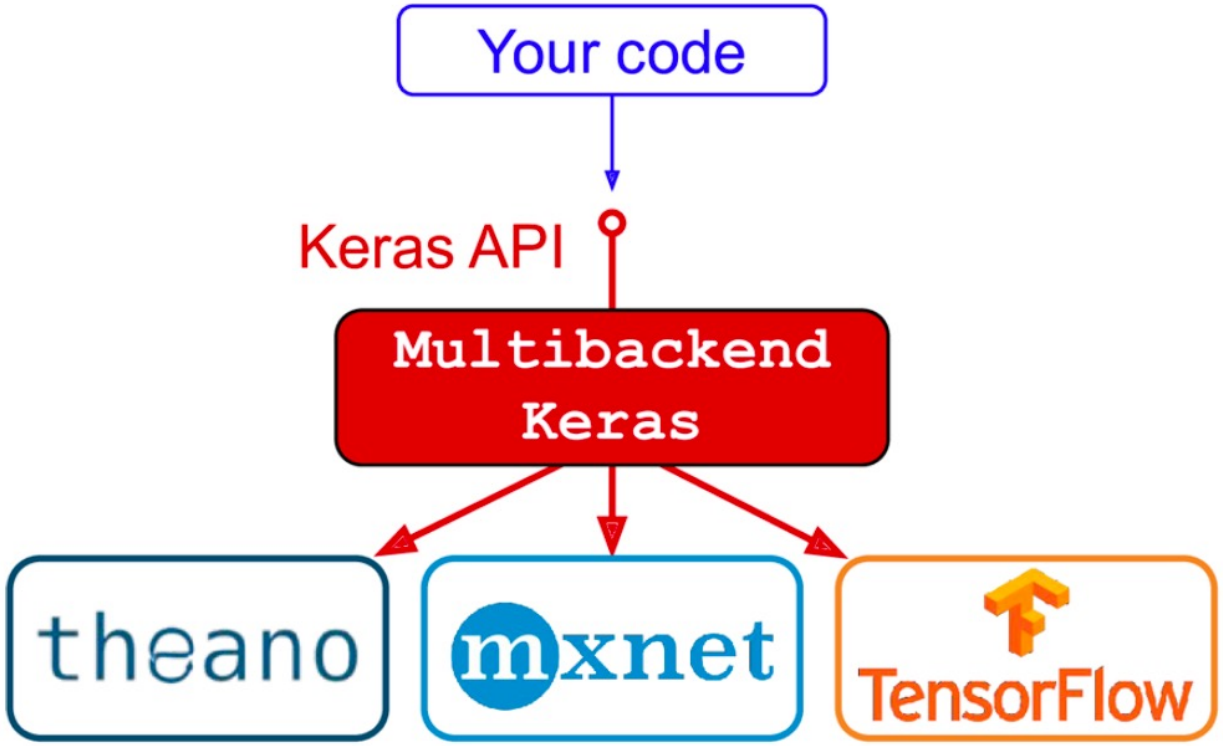
- Lacking documentation

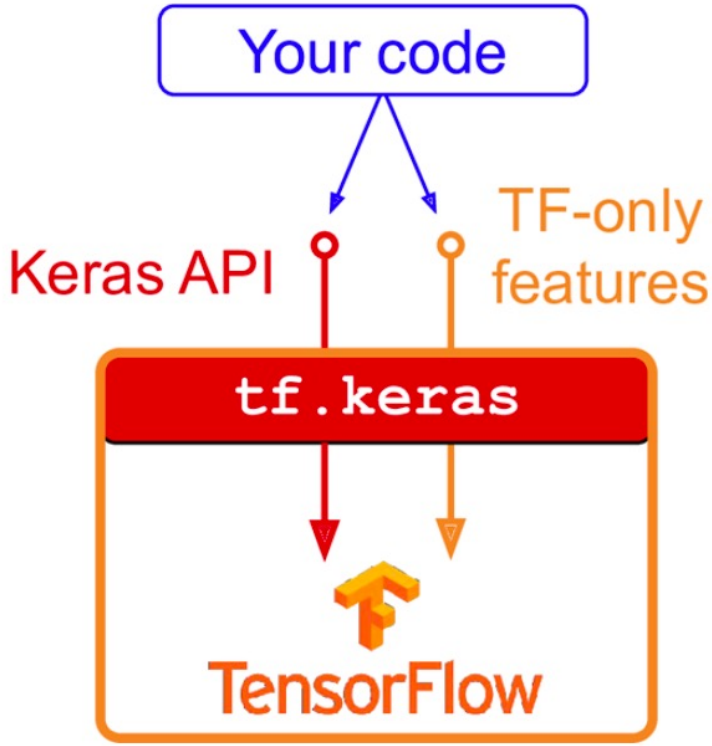
BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Wasn't there also this thing called Keras?

- High-level deep learning API to build, train, evaluate, execute neural networks
  - Documentation: https://keras.io
- Requires computation backend:
  - TensorFlow
  - Microsoft Cognitive Toolkit
  - Theano
  - Apache MXNEt
  - Core ML (Apple)
  - JavaScript & TypeScrit (for web browsers)
  - PlaidML
  - …
- TensorFlow now comes with its own Keras implementation tf.keras (with some added features)
  - E.g., use Data API from TF

Keras in general and Keras using TensorFlow





Source: Geron

TensorFlow in practice


See you next week!

Sources

- DeepLearning.AI, n.d.: deeplearning.ai
- Geron, 2019, Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow
- Goodfellow, Bengio, Courville, 2016, The Deep Learning Book: http://www.deeplearningbook.org
- Liang, 2016, Introduction to Deep Learning: https://www.cs.princeton.edu/courses/archive/spring16/cos495/