CMSC 426

Deep Learning

Two class classification

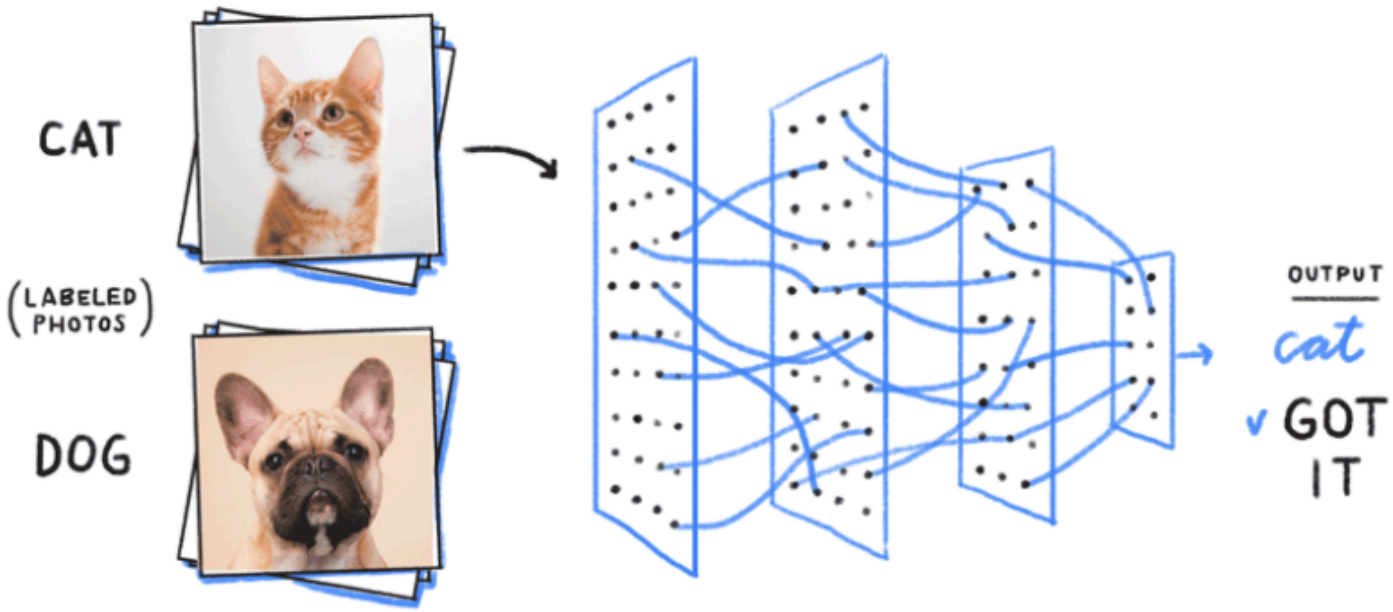
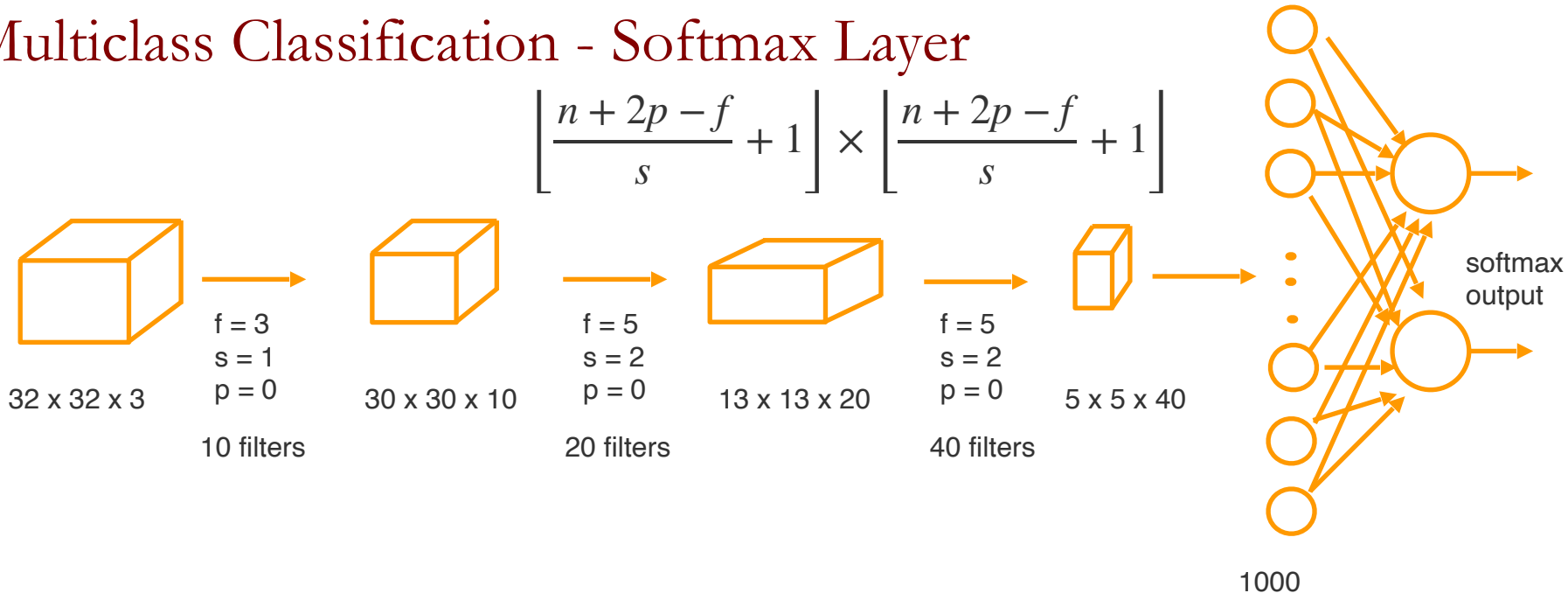


Image Credit: Google

Multiclass Classification - Softmax Layer

$$\left\lfloor \frac{n+2p-f}{s}+1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s}+1 \right\rfloor$$



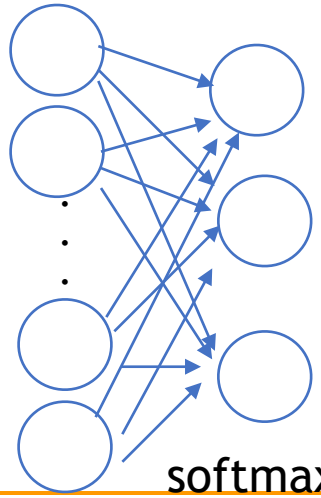
$$z_o = w_o a_{o-1} + b_o$$

z_o - Input to the activation units of the output layer

Activation function, $a = e^{z_o}$

$$a_o = \frac{e^{z_o}}{\sum_{i=1}^{n_o} a_i}$$

$$a_o = g(z_o)$$



softmax layer

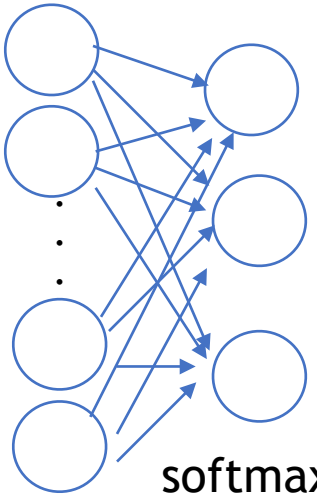
Multiclass Classification - Softmax Layer

$$z_o = w_o a_{o-1} + b_o$$

z_o - Input to the activation units of the output layer

Activation function, $a = e^{z_o}$

$$a_o = \frac{e^{z_o}}{\sum_{i=1}^{n_o} a_i}$$



softmax layer

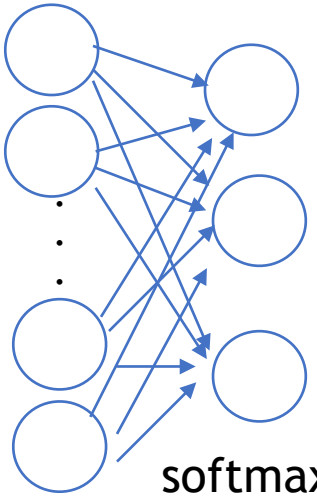
$$a_o = g(z_o)$$

$$z_o = \begin{bmatrix} 2 \\ 4 \\ 1 \\ 0 \end{bmatrix} \qquad a = \begin{bmatrix} e^2 \\ e^4 \\ e^1 \\ e^0 \end{bmatrix} \qquad a_0 = \begin{bmatrix} e^2/(e^2+e^4+e^1+e^0) \\ e^4/(e^2+e^4+e^1+e^0) \\ e^1/(e^2+e^4+e^1+e^0) \\ e^0/(e^2+e^4+e^1+e^0) \end{bmatrix}$$

The values of a_0 should add up to 1

Multiclass Classification - Softmax Layer

$$y = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \hat{y} = \begin{bmatrix} 0.20 \\ 0.25 \\ 0.55 \end{bmatrix}$$



softmax layer

$$\mathscr{L}(y, \hat{y}) = -\sum_{i=1}^{n_o} y_i ln \hat{y}_i = -ln 0.2$$

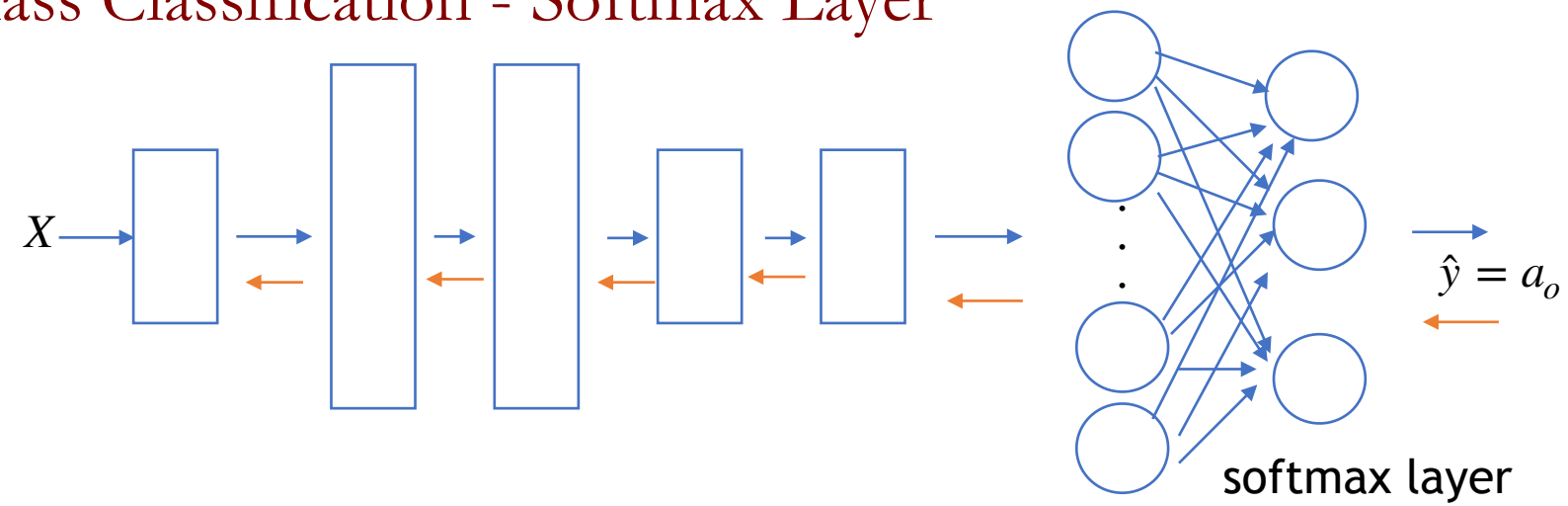
$$J = \frac{1}{m} \sum_{i=1}^{m} \mathscr{L}(y_i, \hat{y}_i)$$

In vector notation

Y is a $3 \times m$ dimensional matrix

$\hat{Y}$ is also a $3 \times m$ dimensional matrix

Multiclass Classification - Softmax Layer



Backpropagation

$$dz_o = \frac{\partial J}{\partial z_o} = \hat{y} - y$$

$$z_o = w_o a_{o-1} + b_o$$

$$\mathscr{L}(y, \hat{y})$$

Machine Learning Hardware

- *CPUs*
- *GPUs*
- *FPGAs*
- *Other accelerators*

Machine Learning hardware

- *Speed up each block of the pipeline*
  for example, matrix-matrix multiplication, convolution

- *Data or memory paths for machine learning work*
  example: caching

- *Application-specific functional units*

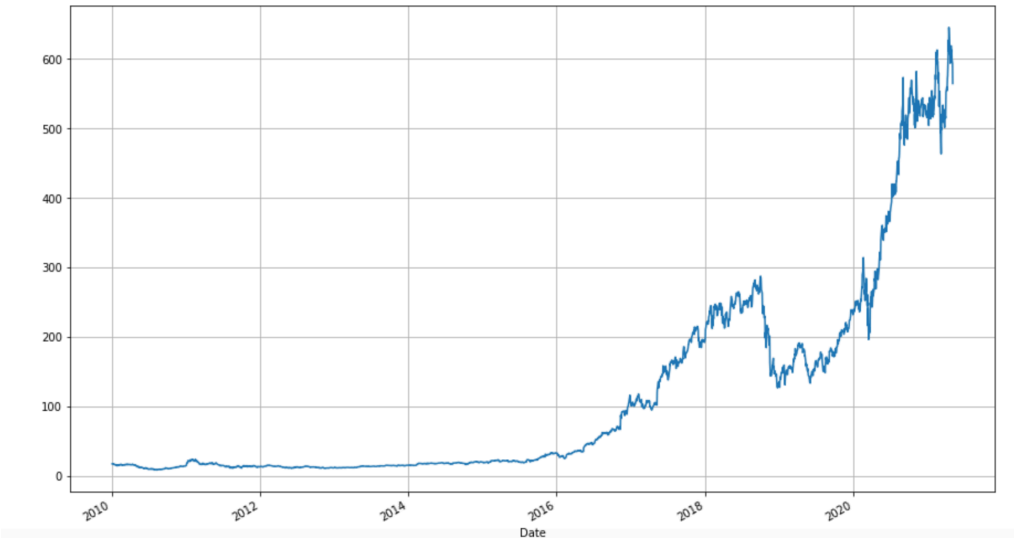
Processing

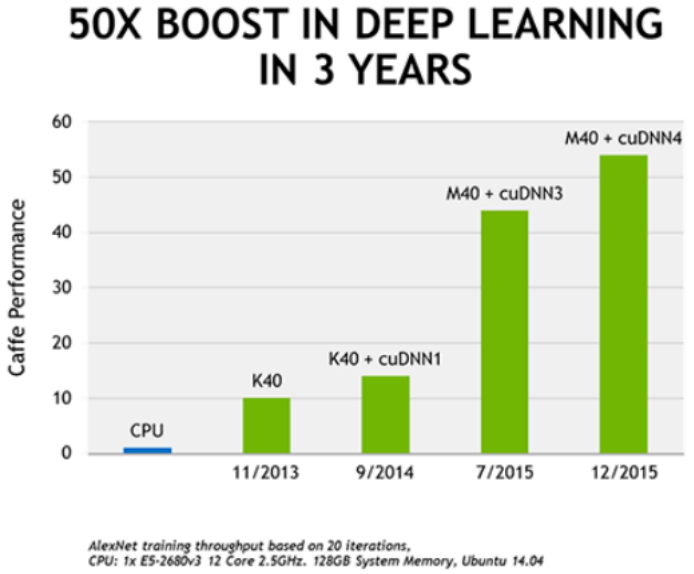
- *CPU is good at executing few complex operations.*
- *In ML most of the processing involves matrix multiplication.*
- *Lots of small calculations.*
- *GPU is well suited for those kind of computations.*

Processing

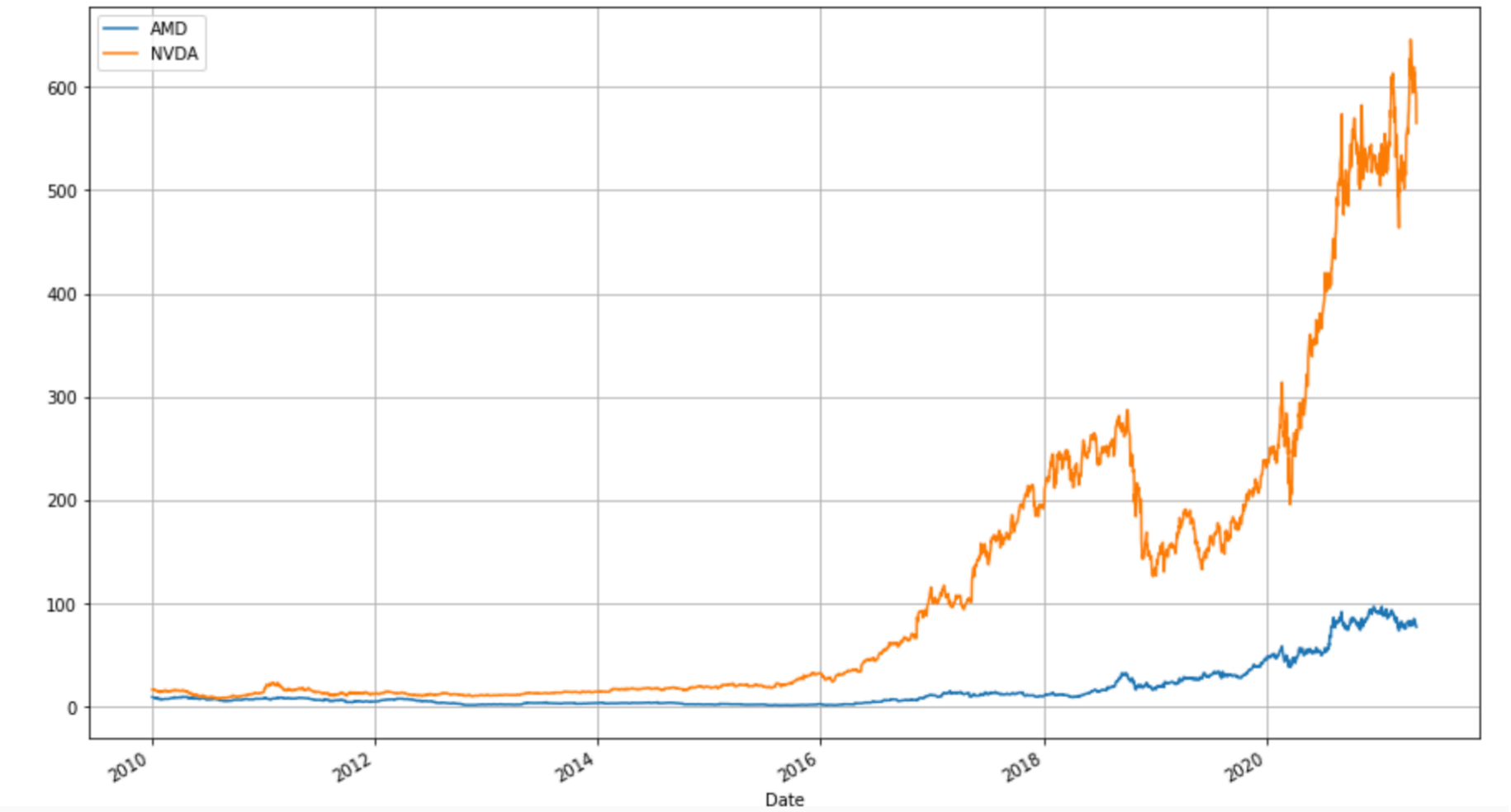
- *GPU utilizes parallel architecture.*

- *It is very good at handling many sets of very simple instructions.*





GPU's


AMD
NVDA
600
500
400
300
200
100
0
2010
2012
2014
2016
2018
2020
Date

CPU's vs. GPU's

18 cores

2560 cores

CPU's vs. GPU's

	CPU i9 Xseries	GeForce GTX 1080
Cores	18 (36 threads)	2560
Clock Speed (GHz)	4.4	1.6G
Memory	Shared	8GB
Price ($)	1799	549

GPU programming

- CUDA
  - C-like code that runs on GPU
  - Other APIs: cuBLAS, cuFFT, cuDNN, etc
- OpenCL
  - Similar to CUDA, but runs on CPU's as well
  - usually slower

Frameworks

- Caffe (Berkeley)
- Torch (NYU / Facebook)
- Theano (University of Montreal)

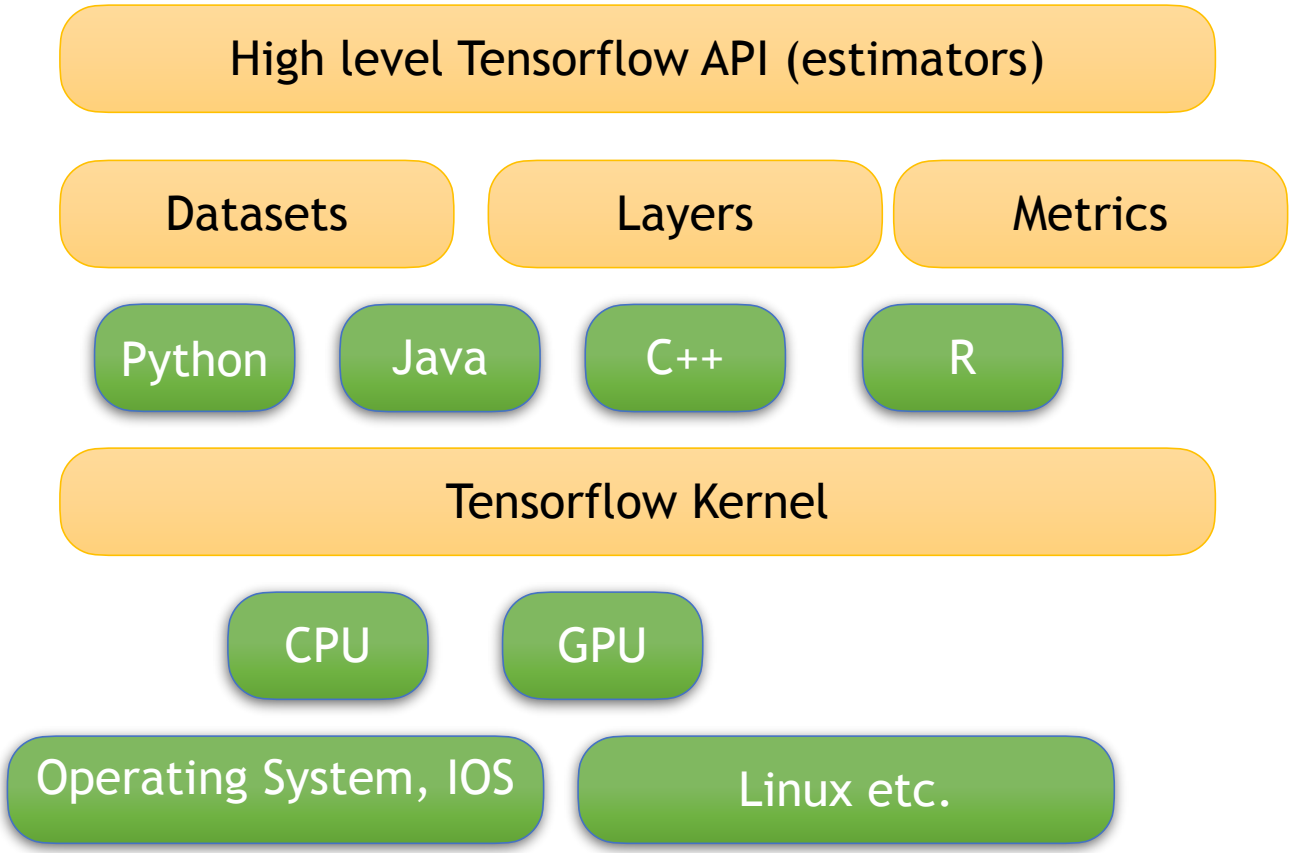
Frameworks

- Caffe (Berkeley)
- Caffe2 (Facebook)
- Torch (NYU / Facebook)
- PyTorch (Facebook)
- Theano (University of Montreal)
- TensorFlow (Google)
- Paddle (Baidu)
- CNTK (Microsoft)
- MXNet (Amazon)

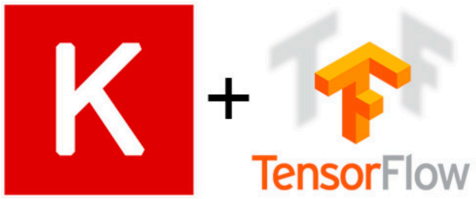
DeepLearning Frameworks

- Gradient computation
- Run on GPU seamlessly

Tensorflow


High level Tensorflow API (estimators)
Datasets
Layers
Metrics
Python
Java
C++
R
Tensorflow Kernel
CPU
GPU
Operating System, IOS
Linux etc.

Keras & Tensorflow



High level Tensorflow API (estimators)

Keras

Datasets

Layers

Metrics

Python

Java

C++

R

Tensorflow Kernel

CPU

GPU

Operating System, IOS

Linux etc.

Backends for Keras

Keras

- *Computational backends perform the heavy lifting*
- *Keras sits on top of it as an abstraction layer.*
- *Tensorflow is now the default backend of Keras.*
- *Keras and tf.keras*