Overview of the tutorial

- We will focus on a well studied task at the LHC: **jet tagging**
- All notebooks and slides are available at: https://github.com/jngadiub/MLtutorial/
- **Part 1:**
  - explore the LHC jet dataset: `Notebook1_ExploreDataset.ipynb`
  - multi-layer perceptron model: `Notebook2_JetID_DNN.ipynb`
  - convolutional 2D NN model: `Notebook3_JetID_Conv2D.ipynb`
- **Part 2:**
  - convolutional 1D NN model: `Notebook4_JetID_Conv1D.ipynb`
  - recurrent NN model: `Notebook5_JetID_RNN.ipynb`
- **Part 3:**
  - autoencoder model: `Notebook6_AE_DNN.ipynb`
  - variational autoencoder mode: `Notebook7_VAE_DNN.ipynb`
  - Surprise BONUS

Setup

- Let's look at the README for the setup: **Colab** or local installation with **Miniconda3**

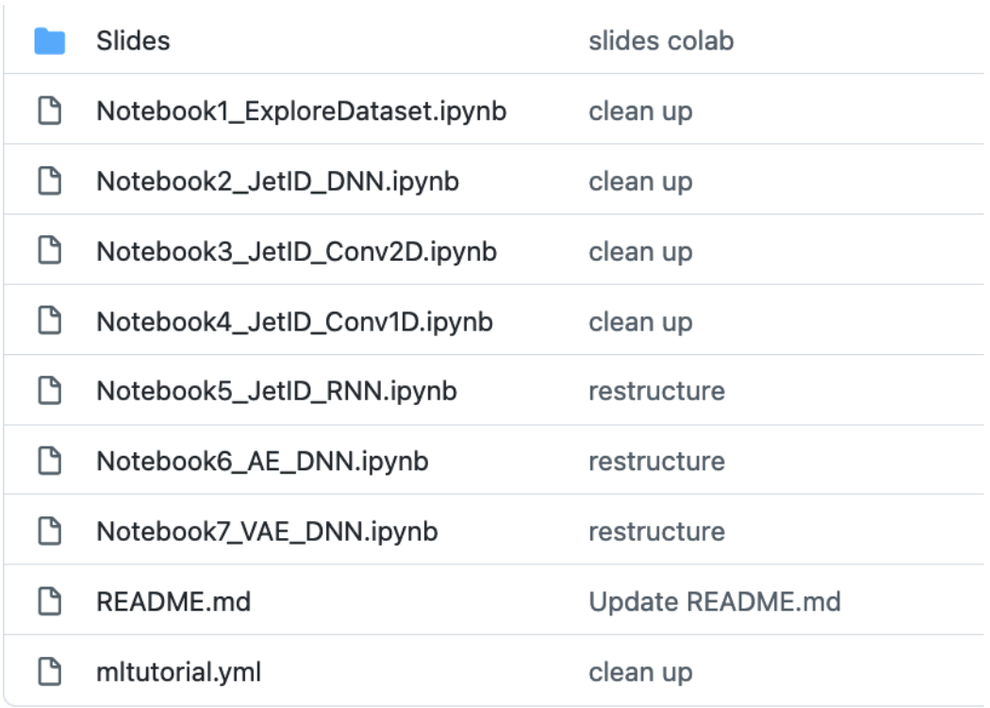
  - **Colab** is a free platform developed by Google to execute code on the cloud
    nb, you will need a google account

  - **Anaconda** is a distribution of python for scientific computing to simplify package management and deployment — the distribution includes data-science packages for all OS

  - **Miniconda** is a small, bootstrap version of Anaconda

- In both setups: the interactive part is served with Python notebooks through **jupyter**

- If you're new to jupyter notebooks, select a cell and hit "shift + enter" to execute the code



Software and tools

- Many solutions exist for training libraries — most popular softwares live in a python ecosystem

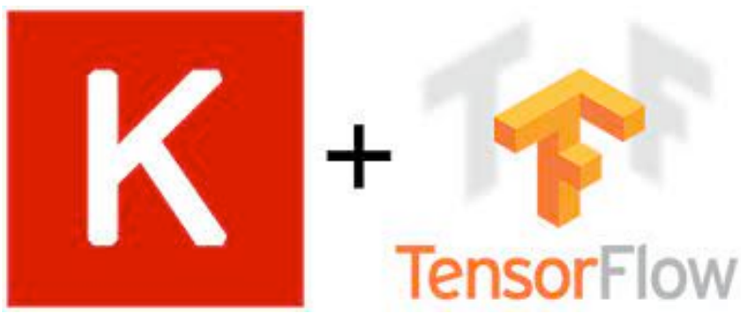
  - Tensorflow (Google)
  - Pytorch (Facebook AI)
  - MXnet (Apache)

- All of them integrated in a data science ecosystem allowing data storage, manipulation, statistical analysis, and plotting

  - h5py, pandas, numpy, scikit, matplotlib, etc …

- Convenient libraries built on top, with pre-coded ingredients:

  - **Keras API for TF → what we will be using in this tutorial**

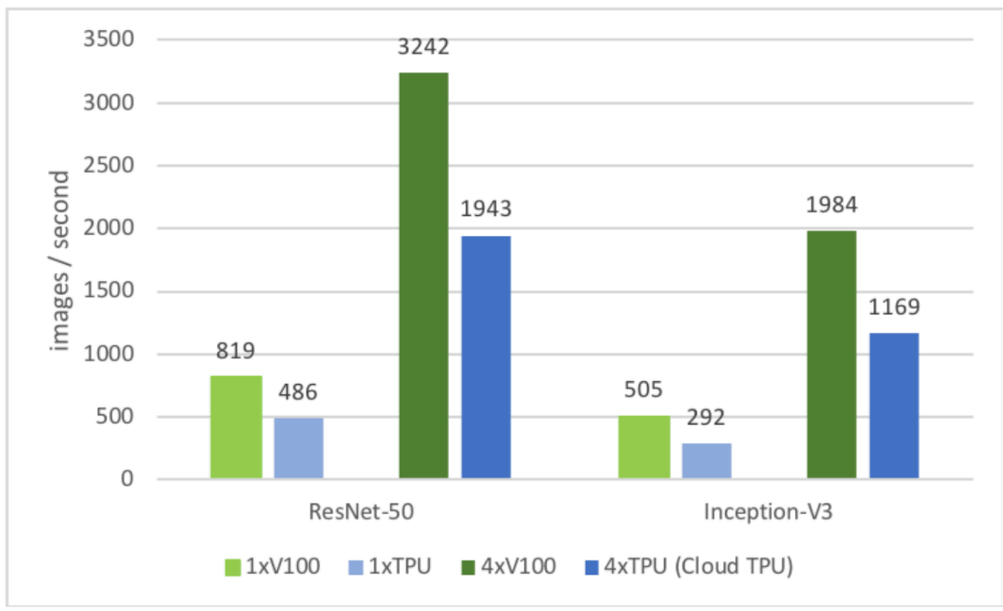


Graphics Processing Units

- All software and tools come with GPU support, through CUDA to run on Nvidia GPUs

- GPUs are very suitable to train neural networks
  - dedicated VRAM provides large memory to load datasets
  - architecture ideal to run vectorised operations on tensors
  - can also paralyse training tasks (e.g., processing in parallel multiple batches)

- Dedicated architectures as field-programmable gate arrays (FPGAs) and application-specific integrated circuits (ASICs) like Google TPU now emerging
  - hopefully I'll introduce you on this in the last part tomorrow through a short dedicated demo



- We will use GPUs on Colab — if you run on your laptop CPU will be slower — if you have access to a remote GPU you can install following same instructions in the README