

Intro to Pytorch and building neural networks with it

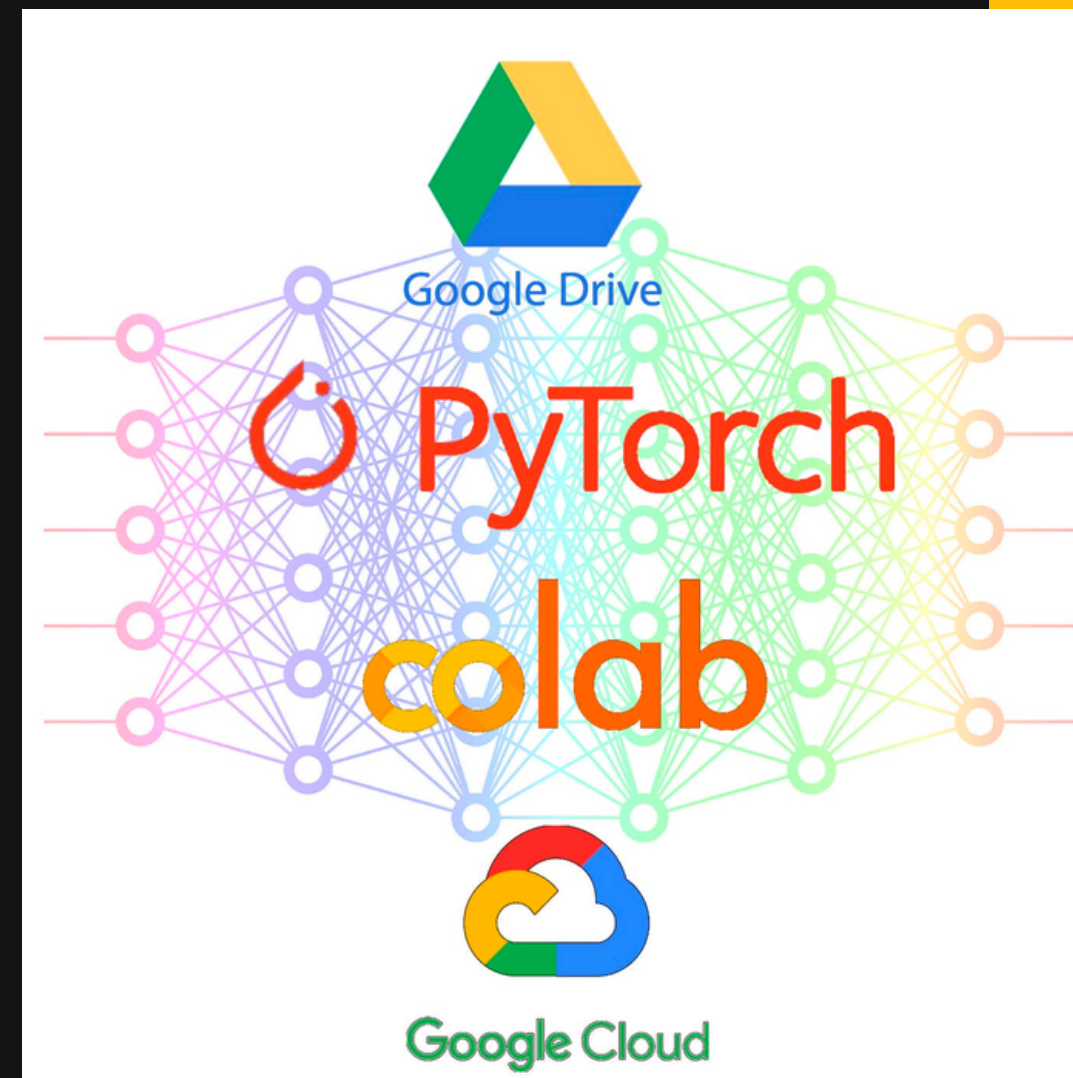


What is Pytorch?

also why?

PyTorch is a machine learning framework.


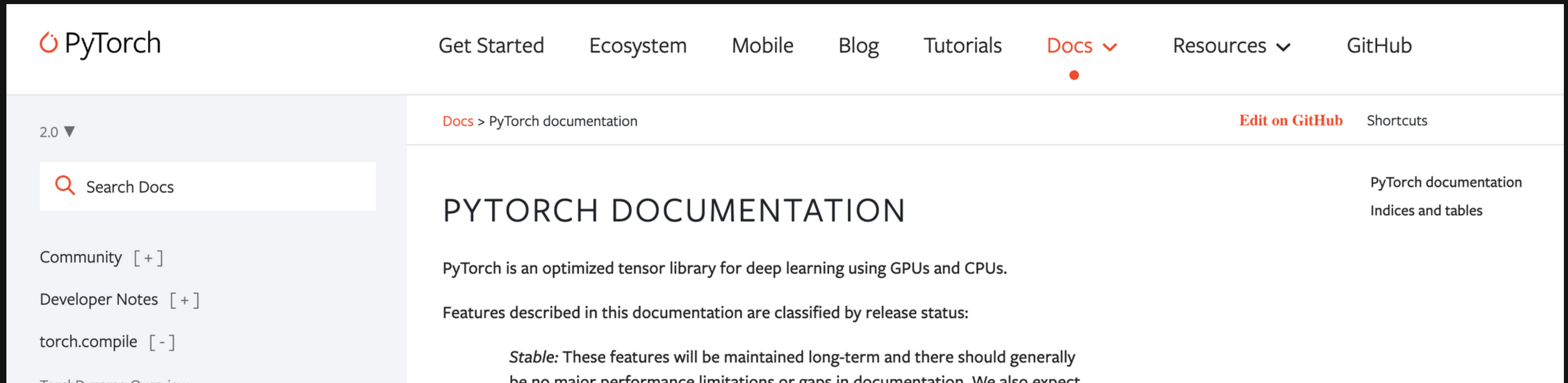
A Machine Learning Framework is an interface, library or tool which allows developers to more easily and quickly build machine learning models, without getting into the nitty-gritty of the underlying algorithms.




Quick Poll




Why Pytorch?



PyTorch is pythonic, which means the procedural coding and syntax is similar to Python. This makes it easy to use and learn.



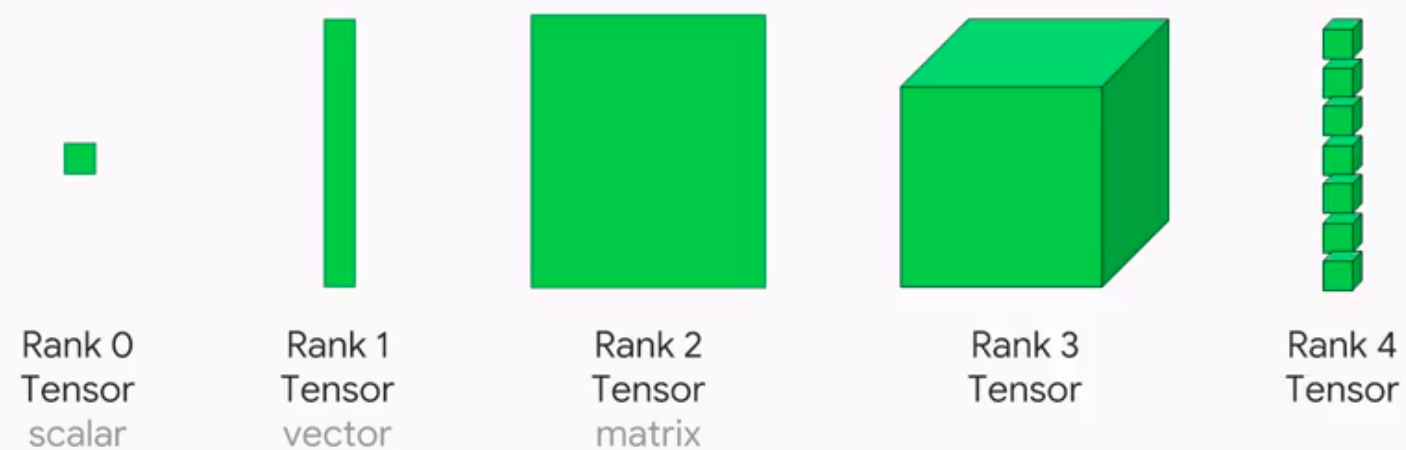
PyTorch supports dynamic computational graphs, enabling network behavior to be changed at runtime. This makes more flexible.



Pytorch has organised, open source documentation. It also boasts an active and helpful community.

The Tensor

A tensor is an N-dimensional array of data



The central component of PyTorch is the **tensor** data structure.

They are similar to numpy n-dimensional arrays. with the key difference being that they are CUDA-capable, and built to run on hardware accelerators, like GPUs. Another important feature that tensors possess is that they are optimized for automatic differentiation, which is the basis of back propagation.

These two optimizations are crucial for deep learning:

- The vast amounts of data, features, and training iterations that deep learning usually encompasses requires the massively-parallel architecture of GPU's to train in reasonable amounts of time.
- Training through back propagation requires the calculation of precise and efficient derivatives.

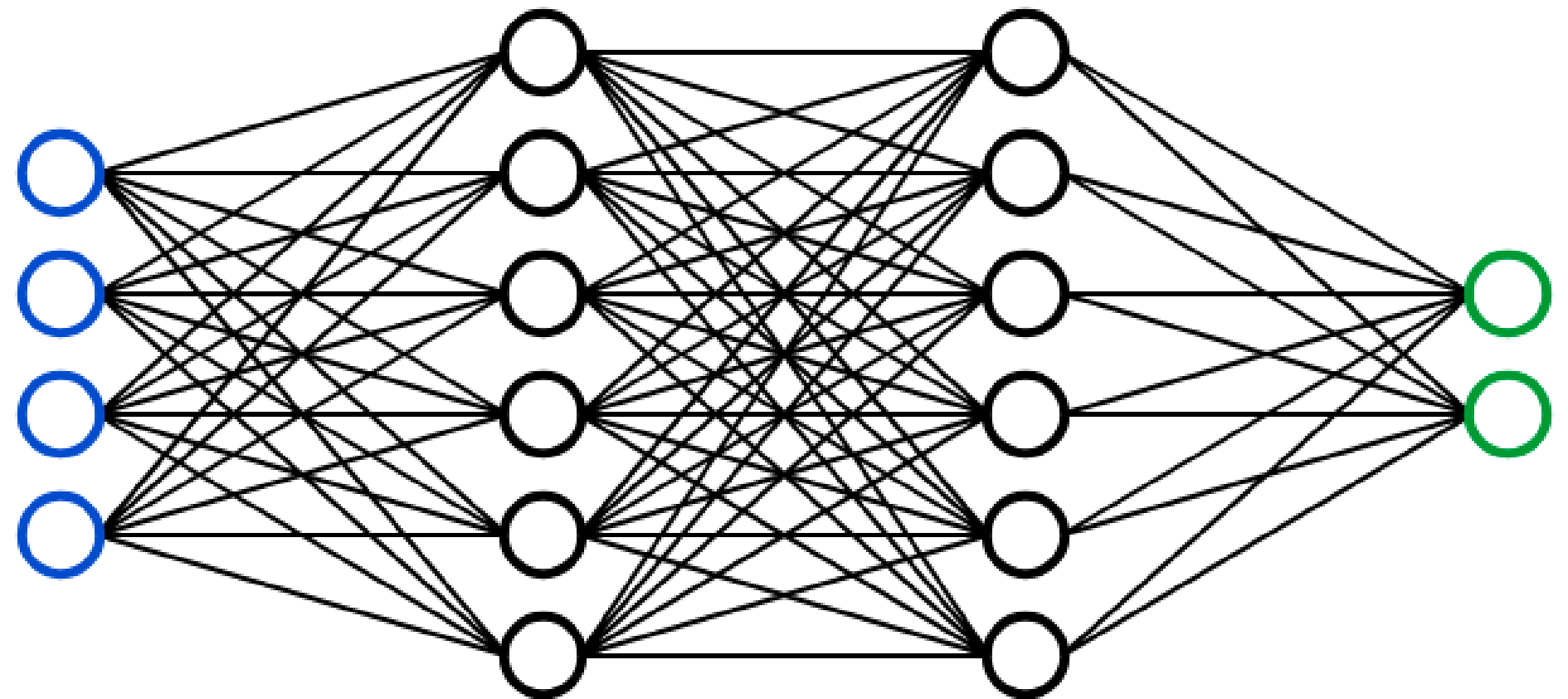
This is the heart of Pytorch.

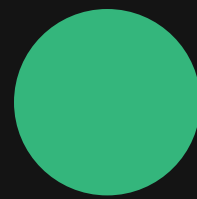
It is a base class used to develop all neural network models using pytorch.

It allows you to easily make linear neural networks, convolutionary neural networks(CNNs), etc. and even combine them to form more complex machine learning architectures.

We will be looking at developing a fully-connected linear neural network today.

nn.module





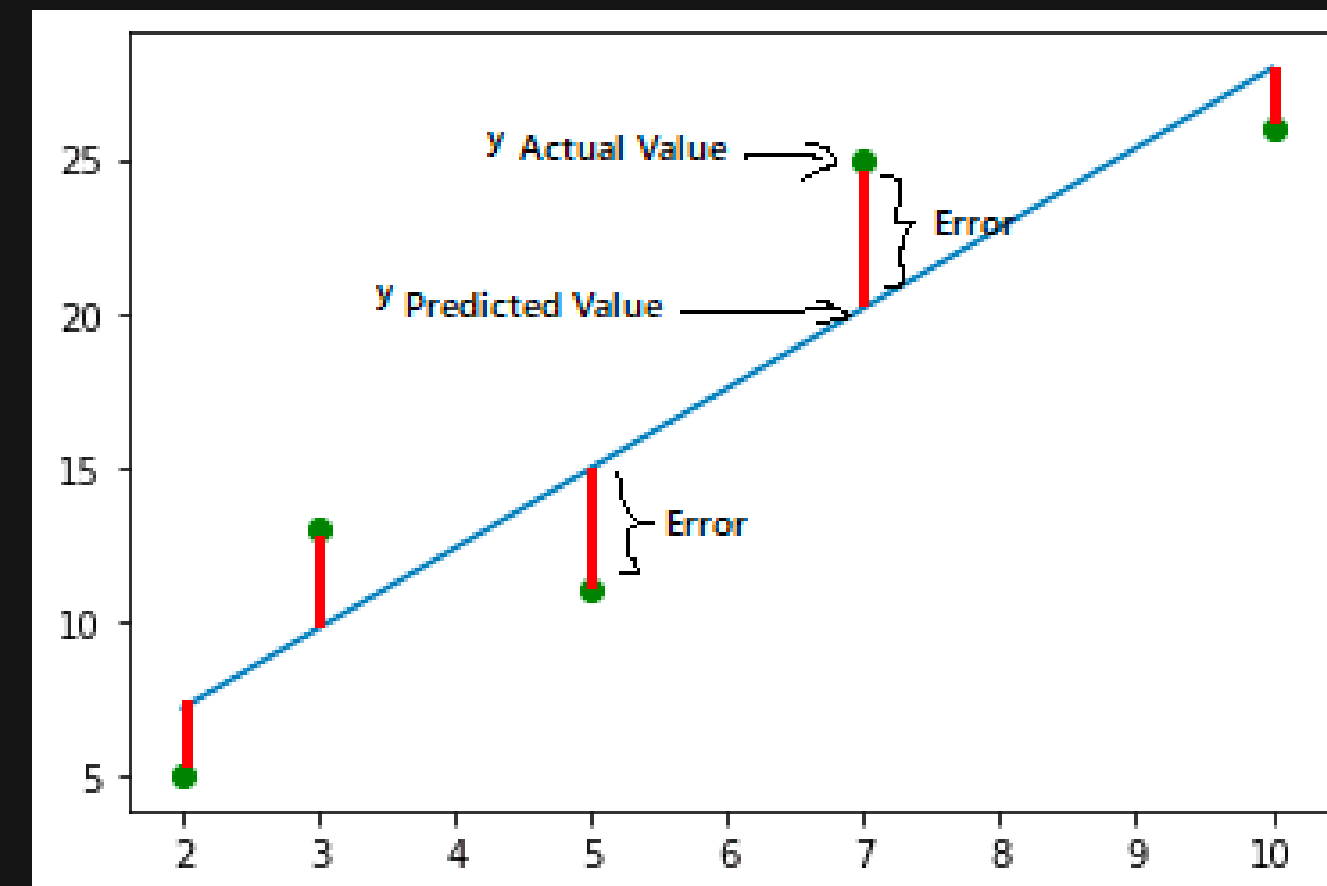
`torch.device()`

Used to set the device on which that particular function or tensor is computed.



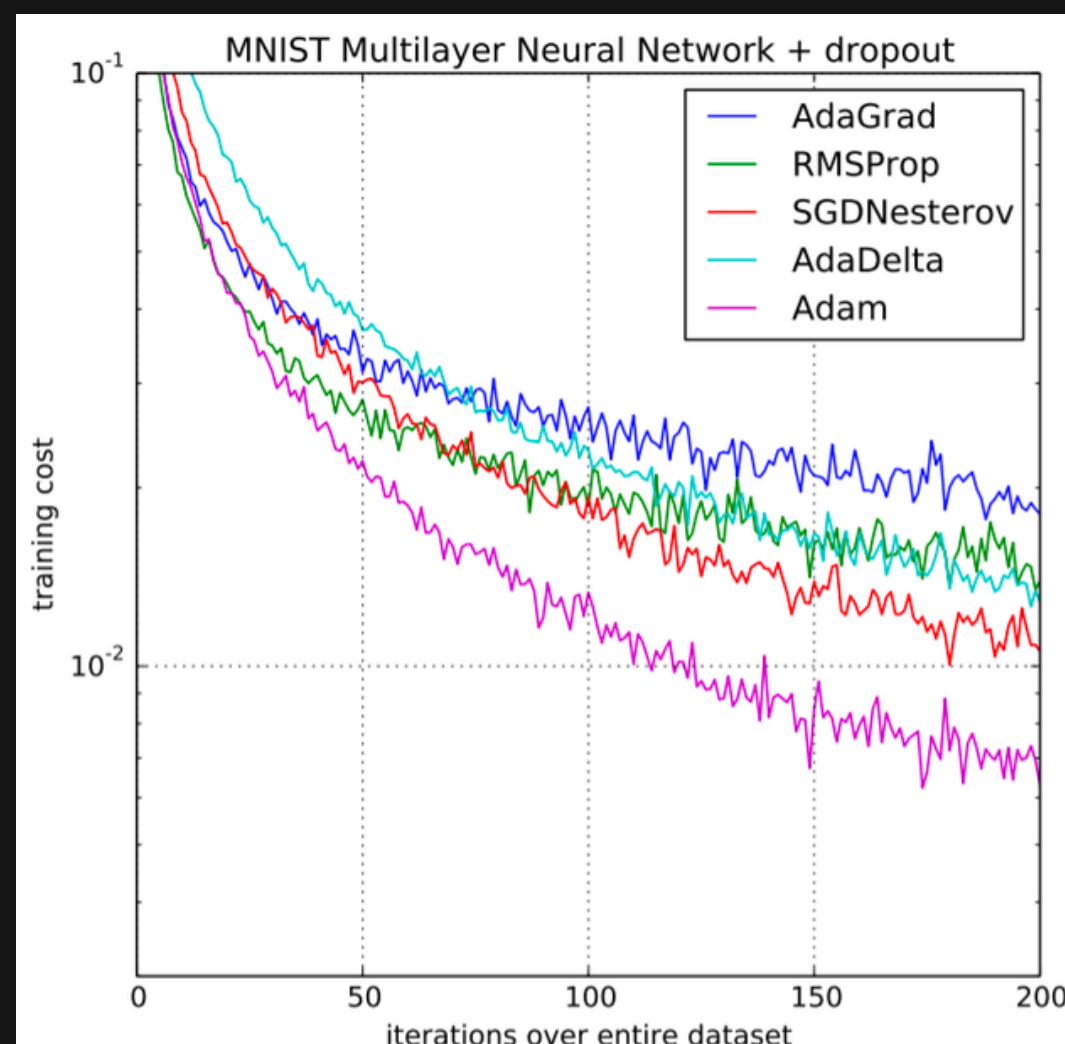
Loss Functions:

Some loss functions are built into pytorch like `nn.MSELoss()` and `nn.CrossEntropyLoss()`

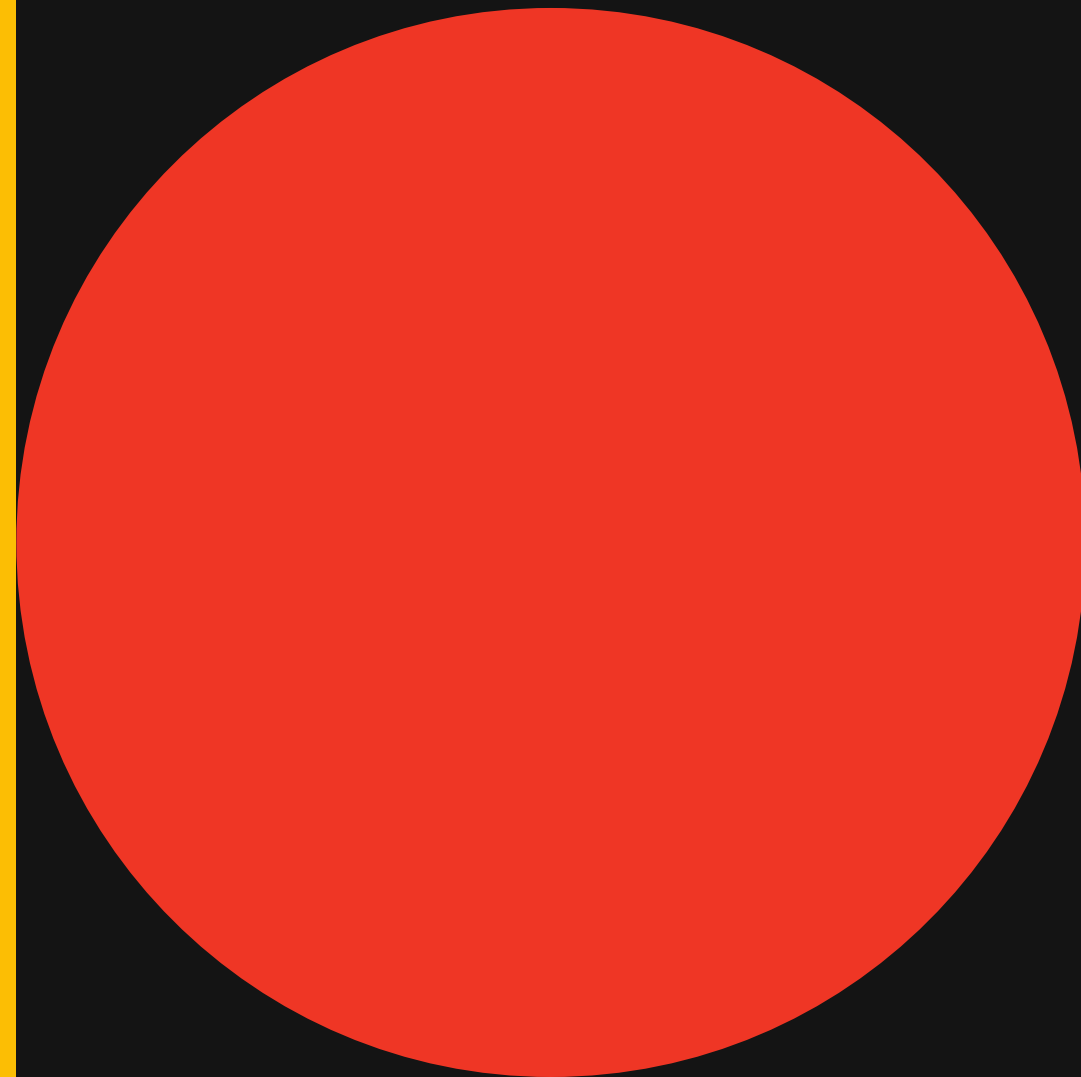


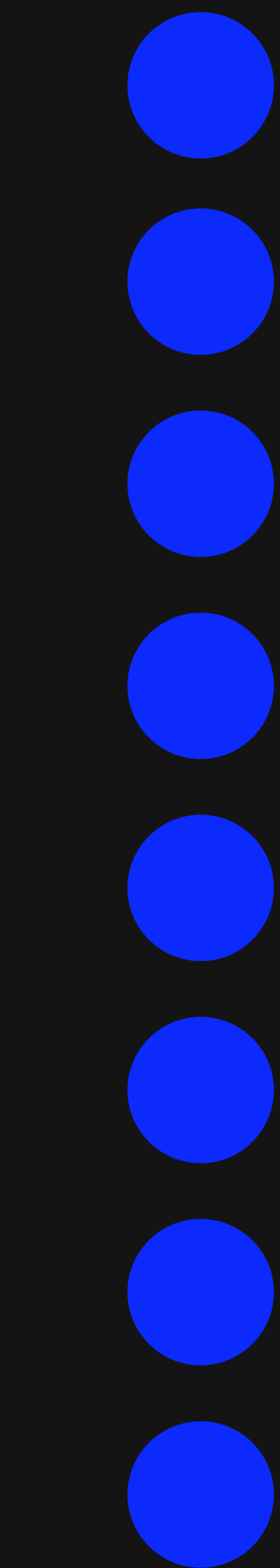
`torch.optim()`

This is a package that includes the implementation of optimisation algorithms like adam and SGD.



Attendance QR Code
for today:





..784 neurons



..50 neurons



0
1
2
3
4
5
6
7
8
9

Model Architecture.

