To know about-----------

1 What is Pytorch

2 Basics of Pytorch Tensors

3 know about Autograd

4 To train Nerual Network in pytorch

5 To train an image classifier in pytorch using CIFAR10

Pytorch :

\*An open-source machine-learning library developed by (FAIR) – Facebook’s AI Research lab.

\*Widely used in Deep learning and AI research & Development

\*It is a Python-based scientific computing package serving two broad purposes:

* A replacement for NumPy to use the power of GPUs and other accelerators.
* An automatic differentiation library that is useful to implement neural networks.

Basics of Pytorch Tensors :

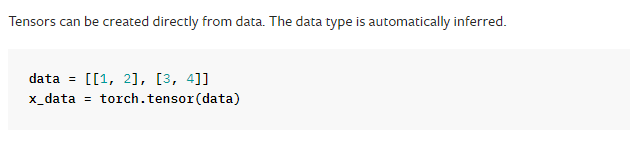
Tensors :

\*Tensors are a specialized data structure that are very similar to arrays and matrices.

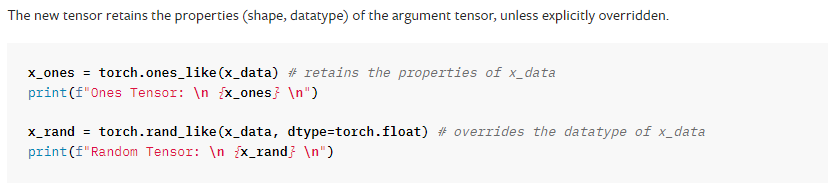
\*In PyTorch, we use tensors to encode the inputs and outputs of a model, as well as the model’s parameters.

>>>Initialization of Tensor

🡪Directly from data

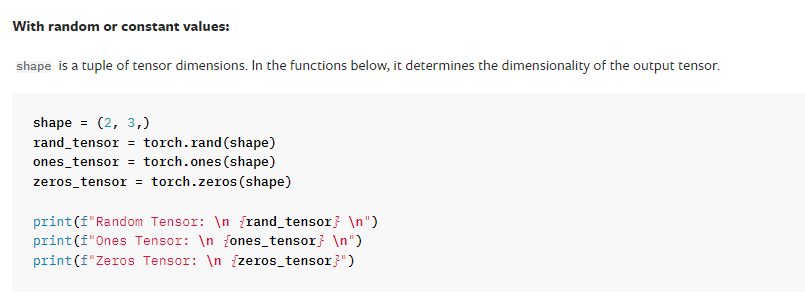


🡪From existing Tensor



O/p: Ones Tensor: tensor([[1, 1],[1, 1]])

Random Tensor: tensor([[0.8823, 0.9150], [0.3829, 0.9593]])



O/p: Random Tensor: tensor([[0.3904, 0.6009, 0.2566],

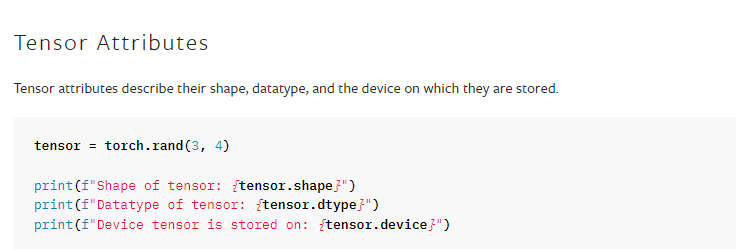
[0.7936, 0.9408, 0.1332]]) //Randomly generated valuex

Ones Tensor: tensor([[1., 1., 1.],

[1., 1., 1.]]) // constant value 1

Zeros Tensor: tensor([[0., 0., 0.],

[0., 0., 0.]]) //constant value 0



O/p: Shape of tensor: torch.Size([3, 4])

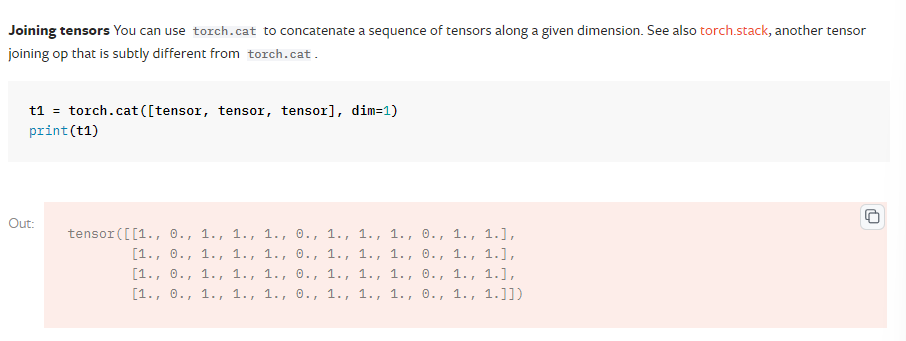
Datatype of tensor: torch.float32

Device tensor is stored on: cpu

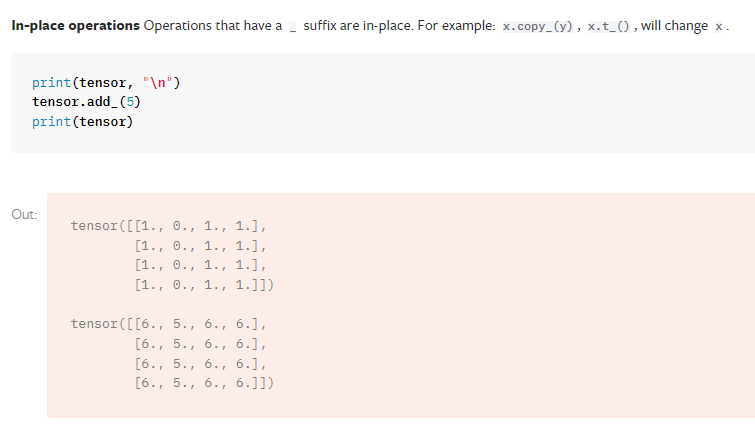
Tensor Operations :

<https://pytorch.org/docs/stable/torch.html>

Some operations are---







Link for basics of TENSORS Execution in colab

<https://colab.research.google.com/drive/18i2USBDFHSpMdJYisrtGqT55SglewuSx?usp=sharing>

--------------------------------------------------------------------------------------------------------------------------------

Torch.autograd…..

**A Gentle Introduction to torch.autograd**

torch.autograd is PyTorch’s automatic differentiation engine that powers neural network training. In this section, you will get a conceptual understanding of how autograd helps a neural network train.

Background

Neural networks (NNs) are a collection of nested functions that are executed on some input data. These functions are defined by *parameters* (consisting of weights and biases), which in PyTorch are stored in tensors.

Training a NN happens in two steps:

**Forward Propagation**: In forward prop, the NN makes its best guess about the correct output. It runs the input data through each of its functions to make this guess.

**Backward Propagation**: In backprop, the NN adjusts its parameters proportionate to the error in its guess. It does this by traversing backwards from the output, collecting the derivatives of the error with respect to the parameters of the functions (*gradients*), and optimizing the parameters using gradient descent.

For better understanding of torch.autograd follow the link for the description

<https://colab.research.google.com/drive/1GMt2GfVyoCU4i3zzm841PwWsPO4f3PnX?usp=sharing>

for the description of Neural Networks….

<https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html>

For to train an image classifier using pytorch with CIFAR 10Data set

<https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html>

aa link refer cheyandi.. for description for labelled data, classifation and image detection…ki

---------------------------------------------------------------------------------------------------------------------------------