What is pytorch?

It is a python based scientific computing package used for :

1. Replacement for numpy to use the power of GPU.
2. A deep learning platform to provide speed and flexibilty.

Numpy arrays uses CPU for computation and so if there are just a few images than it works in an efficient manner. However when we have large no of images for computation than if CPU is used for computation it takes a long long time. Thus we need GPU for computation of large image set in lesser time. But numpy arrays cannot work on GPU, so pytorch is used in which tensors are the alternative of NUMPY arrays.

Tensors

They are similar to numpy arrays. It is used for GPU to accelerate computing.

Firstly import torch library which has the *tensor* function.

```
from __future__ import print_function
import torch
```

Constructing an uninitialized tensor. A two-dimenional array of 5X3 is created.

```
x = torch.Tensor(5, 3)
```

Construct a randomly initialized matrix. This creates a two-dimensional matrix of 5X3 with assigning random values to array.

```
x = torch.rand(5, 3)
```

x.size() will give the size of the array created.

Operations

There are multiple syntaxes for operations. Let's see **addition** as an example:

```
y = torch.rand(5, 3)
```

Syntax 1:

```
    print(x + y)
```

Syntax 2:

```
    print(torch.add(x, y))
```

Syntax 3:

```
    result = torch.Tensor(5, 3)
    torch.add(x, y, out=result)
    print(result)
```

Syntax 4: In-place addition

y.add_(x)
print(y)

Here the value of y changes and become y=x+y

Numpy Bridge

Converting a torch Tensor to a numpy array and vice versa is a breeze.
The torch Tensor and numpy array will share their underlying memory locations, and changing one will change the other.
Just like if we create two pointer for a variable, then changing the value at any of the two locations changes the value of variable.

Converting torch Tensor to numpy Array

a = torch.ones(5) # Creates a one dimensional array (tensor).
Here a is a one dimensional matrix of five elements with value of each element '1'.

b = a.numpy()
a.add_(1)
Array a and b point to the same matrix, hence changing one affects the other.

Converting numpy Array to torch Tensor

```
import numpy as np
a = np.ones(5)
b = torch.from_numpy(a)
np.add(a, 1, out=a)
```

CUDA Tensors

Tensors can be moved onto GPU using the .cuda function.

```
if torch.cuda.is_available():
   x = x.cuda()
   y = y.cuda()
   x + y
```