Deep Learning Software

CPU : Fewer cores, each core is much faster·capable, sequential
GPU : More cores, each core is much slower·dumb, parellel
↳ Matrix Multiplication에 매우 적합

Deep Learning frameworks
- Easily build computational graphs
- Easily compute gradients in computational graphs
- Run it all efficiently on GPU

NumPy
- can't run on GPU

Tensorflow, PyTorch : forward pass looks just like NumPy
computes gradients (GPU)
PyTorch : .cuda()

Keras : Tensorflow wrapper

PyTorch

3 levels of abstraction
- Tensor : Imperative ndarray, but runs on GPU
- Variable : Node in a computational graph ; stores data & gradient
- Module : A neural network layer ; may store state or learnable weights

Can define own autograd functions by writing forward and backward for Tensors (but 대부분 만들어져 있음)

example code:

```
model = torch.nn.Sequential(
            torch.nn.Linear(   ),
            torch.nn.ReLU(),
            torch.nn.Linear(   ))
loss_fn = torch.nn.MSELoss(    )
            ⋮
optimizer = torch.optim.Adam(model.parameters(),
            ⋮                    lr = learning_rate)
for loop → gradient step
```

Can define own Modules using autograd

```
class model(torch.nn.module):
    def: ...
```

DataLoader wraps a Dataset and provides minibatching,
shuffling,
multithreading ...

Dynamic Graph Applications
- Recurrent networks
- Recursive networks
- Modular networks