교 육 세 미 나

ToBig's 9기 이잉걸

PyTorch Tutorials

Contents

Unit 01 | PyTorch vs TensorFlow

"PyTorch vs TensorFlow — spotting the difference"
https://medium.com/towards-data-science/pytorch-vs-tensorflow-spotting-the-difference-25c75777377b

딥러닝을 써보고 싶은데… 뭘로 시작해야 하지?

Unit 01 | PyTorch vs TensorFlow

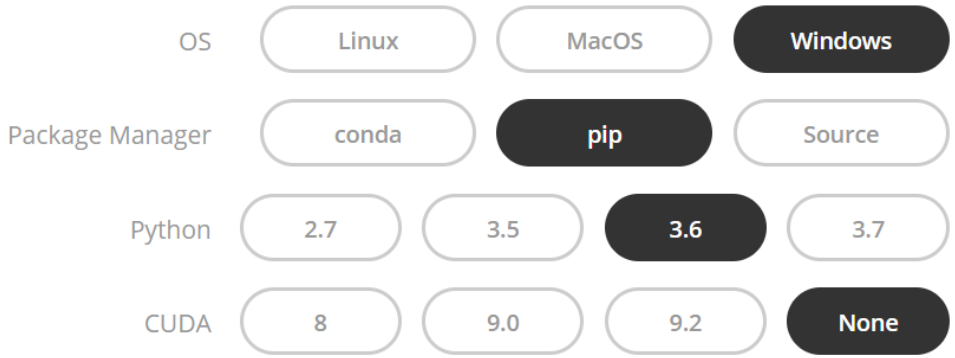
static (tf.session / tf fold)	dynamic (variable length inputs)
difficult to debug (tfdbg)	runtime debugging
Define-and-Run	Define-by-Run
Tensorboard	(Visdom)
rich community	growing community
low-level (but have high-level wrappers)	from low to high-level (Keras <-> torch.nn.sequential)

Unit 01 | PyTorch vs TensorFlow

- 딥러닝에 대한 기초 지식이 있다.
- 딥러닝 코드를 보면 대충 이해는 되지만 스스로 작성하긴 어렵다.
- 쉽게 배워서 빠르게 쓰고 싶다.

✓ Dependencies

- Linux, MacOS, Windows
  - https://pytorch.org/
- Help
  - https://pytorch.org/docs/stable/index.html
- Community
  - https://www.facebook.com/groups/PyTorchKR/?ref=br_rs



✓ Packages

패키지	기능
torch	강력한 GPU 지원 기능을 갖춘 Numpy와 같은 Tensor 라이브러리
torch.autograd	Torch에서 모든 차별화된 Tensor 작업을 지원하는 테이프 기반 자동 차별화 라이브러리
torch.nn	최고의 유연성을 위해 설계된 자동 그래프와 깊이 통합된 신경 네트워크 라이브러리
torch.optim	SGD, RMSProp, LBFGS, Adam 등과 같은 표준 최적화 방법으로 torch.nn과 함께 사용되는 최적화 패키지
torch.utils	편의를 위해 DataLoader, Trainer 및 기타 유틸리티 기능
torch.multiprocessing	파이썬 멀티 프로세싱을 지원하지만, 프로세스 전반에 걸쳐 Torch Tensors의 마법같은 메모리 공유 기능을 제공. 데이터 로딩 및 호그 워트 훈련에 유용.
torch.legacy(.nn/optim)	이전 버전과의 호환성을 위해 Torch에서 이식된 레거시 코드

✓ Tensors

A `torch.Tensor` is a multi-dimensional matrix containing elements of a single data type.

Torch defines eight CPU tensor types and eight GPU tensor types:

Data type	dtype	CPU tensor
32-bit floating point	`torch.float32` or `torch.float`	`torch.FloatTensor`
64-bit floating point	`torch.float64` or `torch.double`	`torch.DoubleTensor`
16-bit floating point	`torch.float16` or `torch.half`	`torch.HalfTensor`
8-bit integer (unsigned)	`torch.uint8`	`torch.ByteTensor`
8-bit integer (signed)	`torch.int8`	`torch.CharTensor`
16-bit integer (signed)	`torch.int16` or `torch.short`	`torch.ShortTensor`
32-bit integer (signed)	`torch.int32` or `torch.int`	`torch.IntTensor`
64-bit integer (signed)	`torch.int64` or `torch.long`	`torch.LongTensor`

Unit 02 | About PyTorch

✓ Tensors

일반적인 준비물

```
import torch
import torch.nn as nn
import torch.nn.functional as F
# import torchvision.datasets as dsets
# import torchvision.transforms as transforms
from torch.autograd import Variable
```

tensor의 선언

```
a = torch.rand(3,3)
print(a)
```

```
tensor([[ 0.6257,  0.0126,  0.6023],
        [ 0.1826,  0.9548,  0.3897],
        [ 0.8464,  0.3374,  0.4689]])
```

Unit 02 | About PyTorch

✓ Making networks

```
'''
class my_network(nn.Module):
    def __init__(self):
    super(my_network, self).__init__()
    (사용할 함수들을 정의할 장소)

    def forward(self, x):
        (함수들을 사용하여 Network의 forward를 정의하는 장소)
        return ~~
'''
```

```
class model(nn.Module):
    def __init__(self):
        super(model, self).__init__()
        self.Max_pool = nn.MaxPool2d(2, stride=1)
        self.Avg_pool = nn.AvgPool2d(2, stride=1)
    def forward(self, x):
        x = F.relu(x)
        x = self.Max_pool(x)
        x = F.tanh(x)
        x = self.Avg_pool(x)
        return x
```

과 제

Coming Soon!

감사합니다!