ImageFolder

상태	CNN
담당자	

`torchvision` 사용해서 커스텀 이미지 불러오기

```
import torchvision
from torchvision import transforms

from torch.utils.data import DataLoader
```

이후 `torchvision.datasets.ImageFolder`를 사용해서 커스텀 이미지를 불러온다. `ImageFolder` 라이브러리는 이미지들이 레이블(cat, dog) 이름으로 된 폴더 안에 들어가 있는 구조라면 바로 불러와 사용할 수 있다.

> ImageFolder - Torchvision main documentation
> Join the PyTorch developer community to contribute, learn, and get your questions answered.
> https://pytorch.org/vision/main/generated/torchvision.datasets.ImageFolder.html

`transform` 옵션 사용해 이미지에 변형 주기

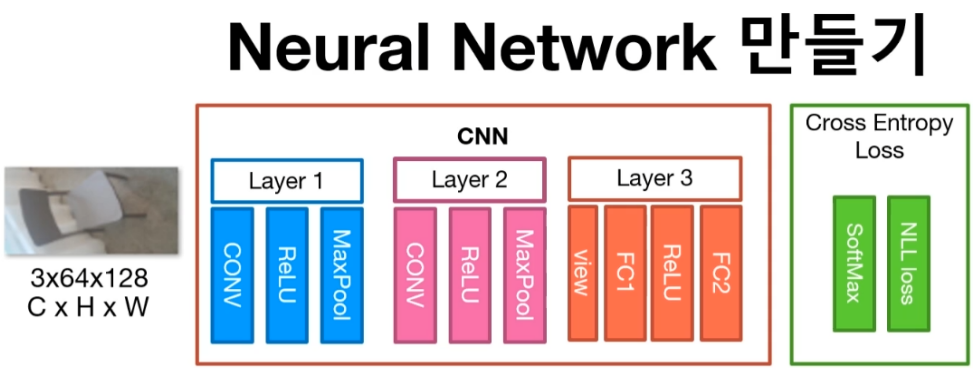
원본 이미지의 크기가 너무 크기 때문에 이미지의 크기를 적당히 줄인 다음 학습에 사용한다. `torchvision.datasets.ImageFolder`의 `transform` 옵션을 수정하면 된다.

```
trans = transforms.Compose([
    transforms.Resize((64,128))
```

```
])

train_data = torchvision.datasets.ImageFolder(root='custom_data/origin_data', transform=trans)
```

CNN 만들기



(Layer 1) Convolution layer = (in_c=3, out_c=6,kernel_size =5, stride=1)
(Layer 1) MaxPool layer = (kernel_size=2, stride =2)

(Layer 2) Convolution layer = (in_c=6, out_c=16, kernel_size =5, stride=1)
(Layer 2) MaxPool layer = (kernel_size=2, stride =2)

(Layer 3) view => (batch_size x [16,13,29] => batch_size x [6032])
(Layer 3) Fully_Connect layer => (input=6032, output = 120)
(Layer 3) Fully_Connect layer => (input=120, output = 2)

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.layer1 = nn.Sequential(
            nn.Conv2d(3,6,5),
            nn.ReLU(),
            nn.MaxPool2d(2),
        )
        self.layer2 = nn.Sequential(
            nn.Conv2d(6,16,5),
            nn.ReLU(),
            nn.MaxPool2d(2),
        )
        self.layer3 = nn.Sequential(
            nn.Linear(16*13*29, 120),
            nn.ReLU(),
            nn.Linear(120,2)
        )
```

```
    def forward(self, x):
        out = self.layer1(x)
        out = self.layer2(out)
        out = out.view(out.shape[0], -1)
        out = self.layer3(out)
        return out
```

이때, `forward` 함수의 매 `out` 사이에 `print(out.shape)`을 넣어서

```
#testing
net = CNN().to(device)
test_input = (torch.Tensor(3,3,64,128)).to(device)
test_out = net(test_input)
```

을 실행할 때마다 텐서의 사이즈가 출력되게 하면 편리하다.

학습한 모델 저장하기

```
torch.save(net.state_dict(), "./model/model.pth")
```