# Image Auto colorization

## 프로젝트 개요

가끔 부모님들의 예전 사진들을 보면 흑백사진이 있는 볼 수 있습니다. 물론 흑백사진 그 자체도 좋지만, 흑백사진을 컬러사진으로 본다면 사람들에게 또 다른 감정이나, 생각이 들게 할 수 있다고 생각해서, 이 프로젝트를 수행하게 되었습니다.

## DATA SET

데이터는 kaggle 에 있는 100_bird_spices 를 선택하였습니다. 새들은 색들이 다채로워서 모델이 어떻게 decoder 하는지 궁금했습니다.


우선 데이터는 224x224x3 사이즈로 컬러 이미지 입니다.

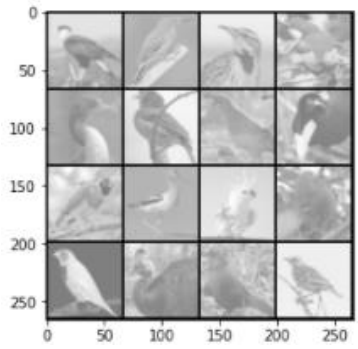
## DATA PREPROCESSING

RGB 채널의 이미지중 흑백부분만을 추출하여 train image 를 구성합니다.

```
def extractGray(batchSize, img):
    lst = []
    for data in img:
        lst.append(data[0])
    return np.asarray(lst).reshape(batchSize, 1, 224, 224)
```

```
train_root = '../input/100-bird-species/train'
valid_root = '../input/100-bird-species/valid'
train_transform = transforms.Compose([
    transforms.CenterCrop(224),
    transforms.ToTensor(),
])
train_dataset = datasets.ImageFolder(train_root,transform=train_transform)
valid_dataset = datasets.ImageFolder(valid_root,transform=train_transform)
index = list(range(len(train_dataset)))
sampler = SubsetRandomSampler(index[:10000])
valid_loader = DataLoader(valid_dataset,batch_size=128,drop_last=True)
train_loader = DataLoader(train_dataset,batch_size=128,drop_last=True,sampler=sampler)
```

```
test_root = '../input/100-bird-species/test'
test_transform = transforms.Compose([
    transforms.CenterCrop(224),
    transforms.ToTensor(),
])
test_dataset = datasets.ImageFolder(test_root,transform=test_transform)
test_loader = DataLoader(test_dataset,batch_size=100,drop_last=True)
```




흑백이미지

## MODEL

모델 구성 방식

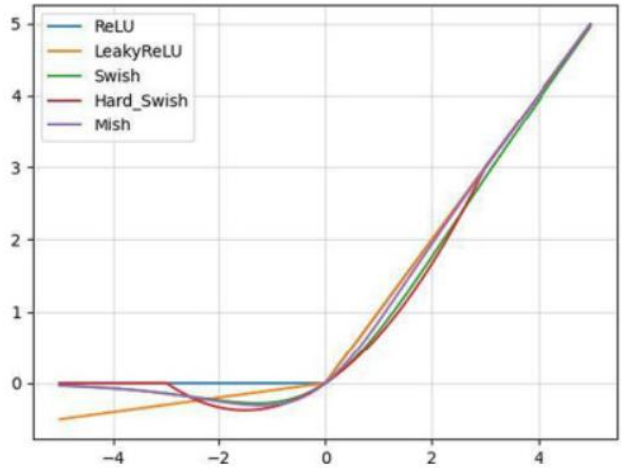
```python
class AutoEncoder(nn.Module):

    def __init__(self):
        super(AutoEncoder, self).__init__()
        # 1 x 224 x 224
        self.conv1 = nn.Conv2d(1, 32, 3, 2, 1)
        self.bn1 = nn.BatchNorm2d(32)
        # 16 x 112 x 112
        self.conv2 = nn.Conv2d(32,64 ,3, 2, 1)
        self.bn2 = nn.BatchNorm2d(64)
        # 32 x 56 x 56
        self.conv3 = nn.Conv2d(64,128, 3, 2, 1)
        self.bn3 = nn.BatchNorm2d(128)
        # 64 x 28 x 28
        self.conv4 = nn.Conv2d(128,256, 3, 2, 1)
        self.bn4 = nn.BatchNorm2d(256)
        # 128 x 14 x 14
        self.conv5 = nn.Conv2d(256,512, 3, 1, 1)
        self.bn5 = nn.BatchNorm2d(512)
        # 256 x 14 x 14
        self.conv6 = nn.Conv2d(512,256, 3, 1, 1)
        self.bn6 = nn.BatchNorm2d(256)
        # 256 x 28 x 28
        self.conv7 = nn.Conv2d(256,128, 3, 1, 1)
        self.bn7 = nn.BatchNorm2d(128)
        # 128 x 56 x 56
        self.conv8 = nn.Conv2d(128,64, 3, 1, 1)
        self.bn8 = nn.BatchNorm2d(64)
        # 64 x 112 x 112
        self.conv9 = nn.Conv2d(64, 32, 3, 1, 1)
        self.bn9 = nn.BatchNorm2d(32)
        # 32 x 224 x 224
        self.out  = nn.Conv2d(32,2, 3, 1, 1)
        # 2 x 224 x 224
        self.up = nn.Upsample(scale_factor=2)
        self.swish = nn.Hardswish()
        self.tanh = nn.Tanh()

    def forward(self, x):
        down_1 = self.swish(self.bn1(self.conv1(x)))
        down_2 = self.swish(self.bn2(self.conv2(down_1)))
        down_3 = self.swish(self.bn3(self.conv3(down_2)))
        down_4 = self.swish(self.bn4(self.conv4(down_3)))
        down_5 = self.swish(self.bn5(self.conv5(down_4)))
        up_1 = self.up(self.swish(self.bn6(self.conv6(down_5))))
        up_2 = self.up(self.swish(self.bn7(self.conv7(up_1))))
        up_3 = self.up(self.swish(self.bn8(self.conv8(up_2))))
        up_4 = self.up(self.swish(self.bn9(self.conv9(up_3))))
        out = self.tanh(self.out(up_4))
        output = torch.cat([x,out],dim=1)
        return output
```

```
AutoEncoder(
  (conv1): Conv2d(1, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv2): Conv2d(32, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv3): Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (bn3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv4): Conv2d(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (bn4): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv5): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (bn5): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv6): Conv2d(512, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (bn6): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv7): Conv2d(256, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (bn7): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv8): Conv2d(128, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (bn8): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv9): Conv2d(64, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (bn9): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (out): Conv2d(32, 2, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (up): Upsample(scale_factor=2.0, mode=nearest)
  (swish): Hardswish()
  (tanh): Tanh()
)
```

down_1 부터 down_5 까지 encoder, up_1 부터 up_4 까지 decoder 를 구성하였습니다.처음 layer 에 채널 gray 만 들어간 이미지가 들어가서 마지막 layer 에서 나머지 색이 있는 채널을 예측하는 모델을 구성했습니다. 그래서 마지막

layer 에서 input 이미지인 x 와 출력 layer 인 out 을 concat 해서 3 채널의 이미지로 변환하였습니다.

그리고 activation 에서는 mobilenetv3 에서 사용된 Hardswish 를 사용하였습니다.




## MODEL COMPILE

optimizer

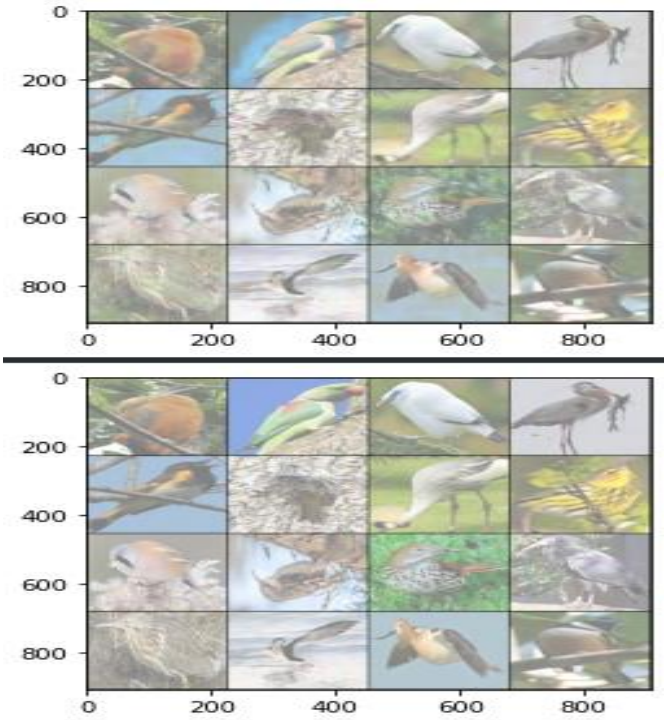
```
device = ('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)
optimizer = torch.optim.Adamax(model.parameters(),lr=0.0003)
criterion = nn.MSELoss()
```

PSLR

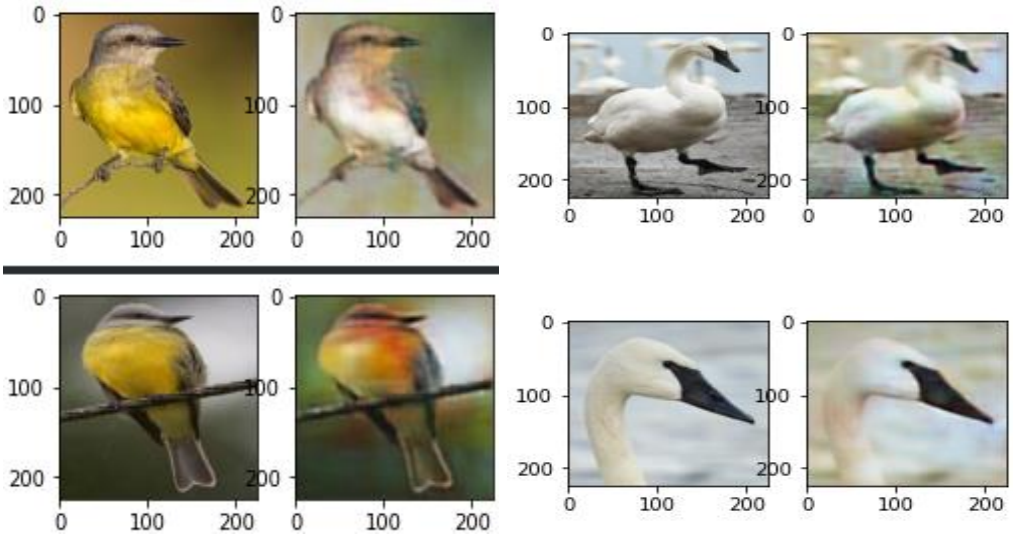
```
def PSNR_loss(pred,real):
    mse_loss = torch.nn.L1Loss()
    mse = mse_loss(real,pred)
    max_pixel = 255.0
    psnr = 20 * torch.log10(max_pixel / torch.sqrt(mse))
    return psnr
```

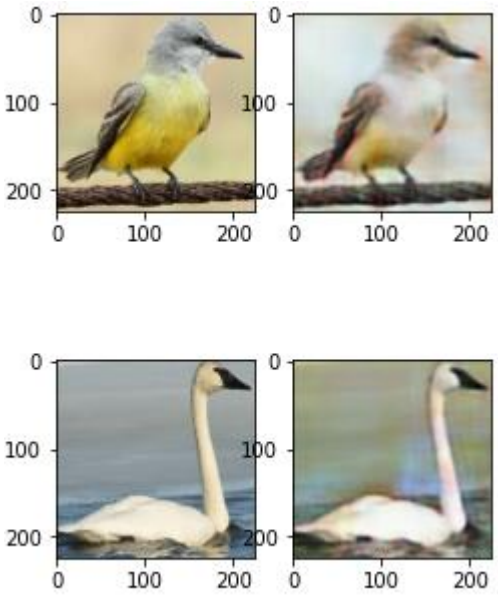
## RESULT

Train_set result




Test_set result





왼쪽이 진짜이미지이고, 오른쪽이 Model 을 통해 생성된 이미지입니다. Epoch 100 으로 훈련을 시켰습니다. 그리고 validation 부분에서 PSNR 이 가장 높은 모델을 저장해서 test_set 의 흑백이미지에 예측을 해보았을때, 표본적인 결과로 보았을 때 색을 어느정도는 예측했지만, 노이즈가 있어서 잔상같은 것이 생기는 부분도 보였다.

그래도 모델이 나름 어울릴 만한 색상을 칠한 것들을 볼 수 있었다.

Reference

Activation_Image : https://www.sciencedirect.com/science/article/pii/S0262885621002225?dgcid=rss_sd_all

function: https://github.com/YBIGTA/Deep_learning/blob/master/GAN/2017-10-27-gan-colorization-revise.md

https://becominghuman.ai/auto-colorization-of-black-and-white-images-using-machine-learning-auto-encoders-technique-a213b47f7339