Colorization

Deep Learning Project

20182786 강민수
20182836 황태균
20192792 이예진

Contents

1. 중간발표 Review

Colorization

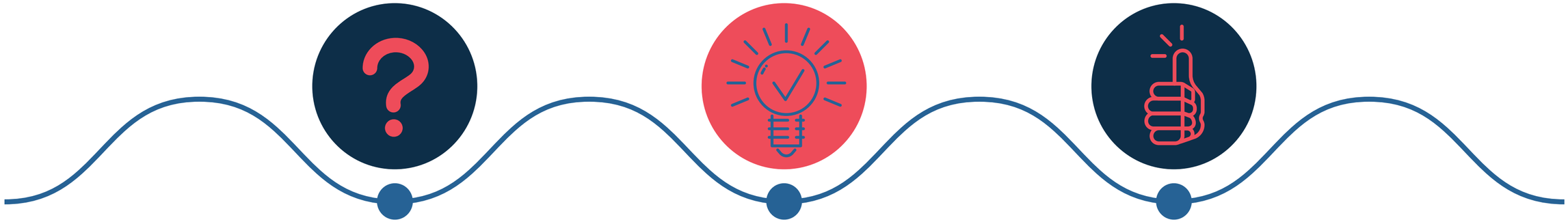
흑백 사진을 칼라 사진으로 Deep Learning을 통해 바꾸는 방법론
단색 사진이나 비디오에 컬러 정보를 추가하는 과정
고유한 솔루션이 없는 3차원 색의 값, 밝기를 이미지와 매핑해야 함

예시

출처: https://arxiv.org/abs/2108.08826

1. 중간발표 Review

주제 선택



단점

Train data와 유사한 image의 Colorization만 잘 이뤄진다는 단점이 존재

Think

특정 데이터셋에 맞게 변형을 해보면 어떨까?

Result

한국과 관련된 이미지 데이터 셋을 활용해서 Colorization을 진행

한국의 이미지 데이터셋에 Colorization 적용

2. Dataset 소개

AI Hub

- 한국 이미지(음식)

출처: https://www.aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&dataSetSn=79

구축 수량 : 150종 X 1,000장

2. Dataset 소개

Data Resize

```
for zip_name in file_list[0:]:
    print(zip_name)
    folder_name = zip_name.split('.')[0]
    f_name = zip_name.split('.')[0]

    with ZipFile(data_path+f'{zip_name}', 'r') as zipObj:
        listOfFileNames = [file for file in zipObj.namelist() if file.endswith(".jpg") or file.endswith(".JPG")]

        fol_lst = []
        for i in listOfFileNames:
            fol_lst.append(i.split('/')[1])
        # 폴더 생성
        for i in fol_lst:
            createDirectory('/content/drive/MyDrive/딥러닝/DATASET/'+f'음식/color_image')
            createDirectory('/content/drive/MyDrive/딥러닝/DATASET/'+f'음식/gray_image')

        for img_num in tqdm(range(0,len(listOfFileNames))):
            image_name = listOfFileNames[img_num].split('.')[0] # 경로 포함한 이미지 이름
            img_real_name = listOfFileNames[img_num].split('/')[-1].split('.')[0] # 이미지 자체의 이름

            name = listOfFileNames[img_num]
            readed = zipObj.open(name) # binary 형태로 읽기

            src=Image.open(readed)
            src = np.array(src)

            src_re=cv2.resize(src, (224,224))

            # 이미지 저장하기
            src_re = Image.fromarray(src_re) # NumPy array to PIL image
            src_re.save('/content/drive/MyDrive/딥러닝/DATASET/'+f"음식/color_image/"+f'{f_name}_{img_real_name}_color.png')
            # 이미지 저장하기
            src_gray = src_re.convert('L') # 흑백 변환
            src_gray.save('/content/drive/MyDrive/딥러닝/DATASET/'+f"음식/gray_image/"+f'{f_name}_{img_real_name}_gray.png')
```

- 종류
  회, 해물, 한과, 튀김, 탕, 찌개, 죽, 찜,
  전골, 조림, 적, 장아찌, 전, 음청류, 장,
  쌈, 밥, 볶음, 무침, 만두, 떡, 나물, 면,
  기타, 김치, 국, 구이 등 …

- 224X224로 resize

- color와 gray 한쌍으로 구성

- train : test = 12,922:2,465 으로 구성

3. 실험 방식

1. 선택한 3가지 모델에 한국 전통 이미지에 적용했을 때 잘 작동되는지 확인해보기

2. 데이터셋(한국 관련 이미지)으로 3가지 모델들을 fine-tuning

3. 그 중 어느 모델이 더 나은지 실험, 비교해보기

선택한 모델 : ChromaGAN, DeOldify, InstColor

4. 모델링

GitHub

Google Colab

4. 모델링

ChromaGAN

Paper : ChromaGAN: Adversarial Picture Colorization with Semantic Class Distribution

```python
class Colorization(nn.Module):
    """
    The Colorization model that takes an input image and outputs a colorized image.
    The model is based on the VGG16 network.
    :param: input_image: input image
    :param: output_image: output image
    """

    def __init__(self, input_size=224):
        super().__init__()
        # raise error if input size is not 224
        if input_size != 224:
            raise ValueError("Input size must be 224")

        # Load VGG16 model pretrained on ImageNet dataset
        vgg = models.vgg16(pretrained=True)

        #  Freeze all the parameters of the VGG16 model
        self.vgg = nn.Sequential(*list(vgg.features.children())[:-8])

        # Global Features
        self.global_features_1 = ConvBlock(512, 512, 3, 2, 1)
        self.global_features_2 = ConvBlock(512, 512, 3, 1, 1)
        self.global_features_3 = ConvBlock(512, 512, 3, 2, 1)
        self.global_features_4 = ConvBlock(512, 512, 3, 1, 1)

        # Dense layers
        self.flatten = nn.Flatten()

        self.fully_connected_1 = nn.Sequential(
            nn.Linear(512 * 7 * 7, 1024),
            nn.Linear(1024, 512),
            nn.Linear(512, 256),
        )
```

```python
        self.fully_connected_2 = nn.Sequential(
            nn.Linear(512 * 7 * 7, 4096),
            nn.Linear(4096, 4096),
            nn.Linear(4096, 1000),
            nn.Softmax()
        )

        # Mid-level Features
        self.mid_level_features_1 = ConvBlock(512, 512, 3, 1, 1)
        self.mid_level_features_2 = ConvBlock(512, 256, 3, 1, 1)

        # Output layers
        self.output_1 = SimpleConvBlock(512, 256, 1, 1, 0)
        self.output_2 = SimpleConvBlock(256, 128, 3, 1, 1)
        self.output_3 = SimpleConvBlock(128, 64, 3, 1, 1)
        self.output_4 = SimpleConvBlock(64, 64, 3, 1, 1)
        self.output_5 = SimpleConvBlock(64, 32, 3, 1, 1)
        self.output_6 = nn.Sequential(
            nn.Conv2d(32, 2, 3, 1, 1),
            nn.Sigmoid()
        )

        self.up_sample = nn.Upsample(scale_factor=2)
```

224 × 224 pixels를 VGG-16에서 학습

4. 모델링

ChromaGAN

Paper : ChromaGAN: Adversarial Picture Colorization with Semantic Class Distribution

config.py × 리소스

```python
import torch

GRADIENT_PENALTY_WEIGHT = 10

EPOCHS = 10
MODEL_DIR = 'models'
MODEL_C = MODEL_DIR + '/colorization_10.pt'
MODEL_D = MODEL_DIR + '/discriminator_10.pt'

TRAIN_PATH = '../DATASET/train/color_image'
TEST_PATH = '../DATASET/test/color_image'

TR_PATH = 'train/'
TE_PATH = 'test/'

OUTPUT_PATH = 'Output_10/'+TR_PATH

BATCH_SIZE = 1
TEST_BATCH_SIZE = 1

# -1 for no log
CHECK_PER = 100

LR = 2e-5
DEVICE = "cuda" if torch.cuda.is_available() else "cpu"

```

GitHub 코드 이용

orignal code :
https://github.com/pvitoria/ChromaGAN

pytorch version remake :
https://github.com/superhighlevel/ChromaGan_Pytorch_Remake

4. 모델링

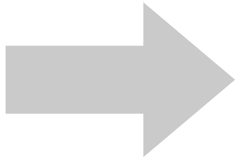
ChromaGAN

Paper : ChromaGAN: Adversarial Picture Colorization with Semantic Class Distribution

```
LOSS_G = MODEL_DIR + '/Loss_G.pkl'
LOSS_D = MODEL_DIR + '/Loss_D.pkl'
```

```
#Loss_G[epoch] = l_g
#Loss_D[epoch] = l_d

#with open(config.LOSS_G,'wb') as f:
#    pickle.dump(Loss_G,f)
#with open(config.LOSS_D,'wb') as f:
#    pickle.dump(Loss_D,f)
```



```
EPOCH 0 / 10
------------------------------
  0% 0/12922 [00:00<?, ?it/s]

Epoch 0 - Batch 0 - Loss G: -0,046164851635694504 - Loss D: 6,5293869972229
  8% 1000/12922 [06:20<23:35,  8,42it/s]

Epoch 0 - Batch 1000 - Loss G: -0,04567399621009827 - Loss D: 7,085413455963135
 15% 2000/12922 [08:20<21:43,  8,38it/s]

Epoch 0 - Batch 2000 - Loss G: -0,04809000343084335 - Loss D: 3,0626063346862793
 23% 3000/12922 [10:19<19:26,  8,51it/s]

Epoch 0 - Batch 3000 - Loss G: -0,049879949539899826 - Loss D: 9,374524116516113
 31% 4000/12922 [12:19<17:43,  8,39it/s]

Epoch 0 - Batch 4000 - Loss G: -0,04450934752821922 - Loss D: 0,6630321741104126
 39% 5000/12922 [14:18<15:37,  8,45it/s]

Epoch 0 - Batch 5000 - Loss G: -0,046820785850286484 - Loss D: 1376,5238037109375
 46% 6000/12922 [16:17<13:45,  8,39it/s]

Epoch 0 - Batch 6000 - Loss G: -0,046261802315711975 - Loss D: 16,247177124023438
 54% 7000/12922 [18:18<11:42,  8,43it/s]

Epoch 0 - Batch 7000 - Loss G: -0,05131150782108307 - Loss D: 69,78662872314453
 62% 8000/12922 [20:17<09:45,  8,41it/s]
```

Loss 값 저장해서 확인해보려고 코드를 추가했지만
out of memory로 제대로 진행이 안됨
=> print되는 것으로 loss 확인

4. 모델링

ChromaGAN

Paper : ChromaGAN: Adversarial Picture Colorization with Semantic Class Distribution

out of memory로 인해
epoch 수를 줄이는 등 시도를 해본 뒤, 데이터 양을 줄이기로 하고
train : test = 12,922 : 2,465로 확정

Colab Pro

python version = 3.7

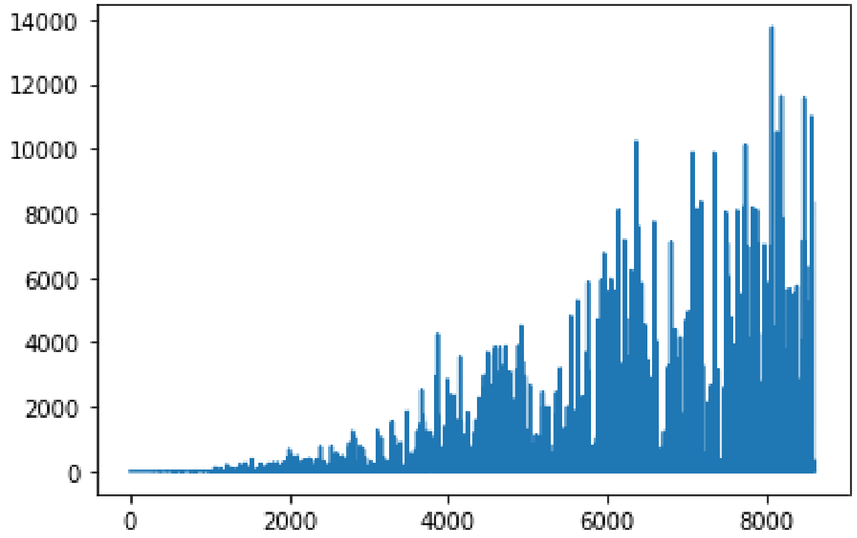
epochs = 10
learning_rate = 2e-5

4. 모델링

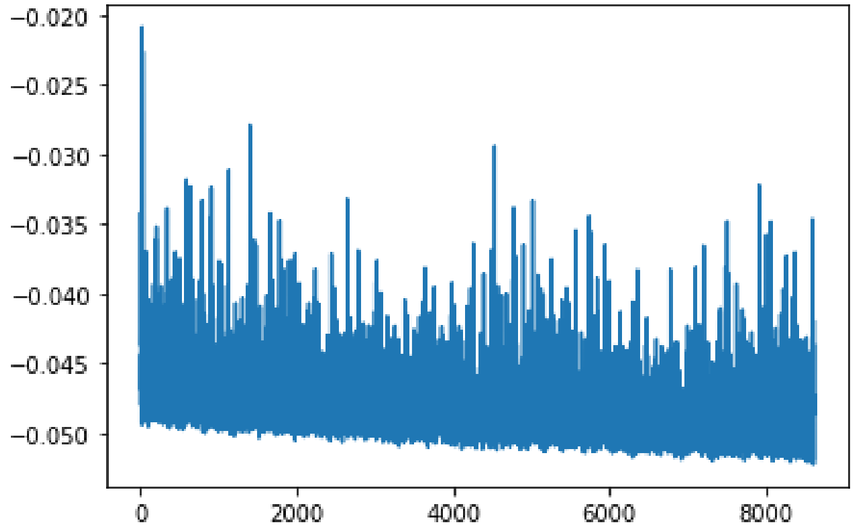
ChromaGAN

Paper : ChromaGAN: Adversarial Picture Colorization with Semantic Class Distribution

Discriminator Loss 일부



Generator Loss 일부



4. 모델링

DeOldify

GitHub 코드 이용

code :
https://github.com/jantic/DeOldify

학습 진행 실패

pre-train된 모델에 넣은 결과물 일부







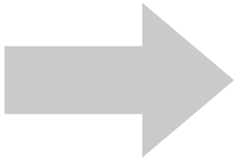
4. 모델링

InstColor

Paper : Instance-aware Image Colorization, CVPR 2020

GitHub 코드 이용

code :
https://github.com/ericsujw/InstColorization



학습 진행 실패

추가 참조 :
https://github.com/dayoungMM/Deepcolorization

4. 모델링

InstColor

Paper : Instance-aware Image Colorization, CVPR 2020

Fine-training 실패이유 : 원작자의 Pre-train-weight가 있는 URL이 삭제됨(원작자 URL업데이트 X)
해결 방안 : 삭제된 checkpoints 파일을 구글링하여 찾은 후 URL을 수정 후 다운받아서 사용
해결 방안 실패 이유 : checkpoints의 Pre-training된 Weight들이 일부밖에 없었음 완전한 파일 존재 X

download_model.sh download.py

```python
#taken from this StackOverflow answer: https://stackoverflow.com/a/39225039
import requests
from os.path import join, isdir
import os
from argparse import ArgumentParser

def download_file_from_google_drive(id, destination):
    URL = "https://docs.google.com/uc?export=download"

    session = requests.Session()

    response = session.get(URL, params = { 'id' : id }, stream = True)
    token = get_confirm_token(response)

    if token:
        params = { 'id' : id, 'confirm' : token }
        response = session.get(URL, params = params, stream = True)

    save_response_content(response, destination)

def get_confirm_token(response):
    for key, value in response.cookies.items():
        if key.startswith('download_warning'):
            return value
```

```
!sh scripts/train.sh

cp: cannot stat './checkpoints/siggraph_retrained/latest_net_G.pth': No such file or directory
#training images = 15
initialize network with normal
model [TrainModel] was created
Setting up a new session...
Exception in user code:
------------------------------------------------------------
Traceback (most recent call last):
  File "/usr/local/lib/python3.8/dist-packages/urllib3/connection.py", line 158, in _new_conn
    conn = connection.create_connection(
  File "/usr/local/lib/python3.8/dist-packages/urllib3/util/connection.py", line 80, in create_connection
    raise err
  File "/usr/local/lib/python3.8/dist-packages/urllib3/util/connection.py", line 70, in create_connection
    sock.connect(sa)
ConnectionRefusedError: [Errno 111] Connection refused
```

```
!sh scripts/train.sh

WARNING: timestamping does nothing in combination with -O. See the manual
for details.

--2022-12-11 11:28:31--  http://colorization.eecs.berkeley.edu/siggraph/models/pytorch.pth
Resolving colorization.eecs.berkeley.edu (colorization.eecs.berkeley.edu)... 128.32.244.190
Connecting to colorization.eecs.berkeley.edu (colorization.eecs.berkeley.edu)|128.32.244.190|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 136785828 (130M)
Saving to:  './checkpoints/siggraph_retrained/latest_net_G.pth'

./checkpoints/siggr 100%[===================>] 130.45M  11.3MB/s    in 36s

2022-12-11 11:29:07 (3.67 MB/s) -  './checkpoints/siggraph_retrained/latest_net_G.pth'  saved [136785828/136785828]

mkdir: cannot create directory  './checkpoints/coco_full' : File exists
#training images = 15
initialize network with normal
model [TrainModel] was created
loading the model from ./checkpoints/coco_full/latest_net_G.pth
Setting up a new session...
Exception in user code:
------------------------------------------------------------
Traceback (most recent call last):
  File "/usr/local/lib/python3.8/dist-packages/urllib3/connection.py", line 158, in _new_conn
    conn = connection.create_connection(
  File "/usr/local/lib/python3.8/dist-packages/urllib3/util/connection.py", line 80, in create_connection
    raise err
  File "/usr/local/lib/python3.8/dist-packages/urllib3/util/connection.py", line 70, in create_connection
    sock.connect(sa)
ConnectionRefusedError: [Errno 111] Connection refused

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/usr/local/lib/python3.8/dist-packages/urllib3/connectionpool.py", line 597, in urlopen
    httplib_response = self._make_request(conn, method, url,
```

5. 평가 지표

PSNR(Peak Signal-to-Nosie Ratio)

: 실제 이미지와 생성된 이미지 간의 손실 정도를 판단하기 위한 평가 지표

```python
from math import log10, sqrt
import cv2
import numpy as np

def PSNR(original, compressed):
    mse = np.mean((original - compressed) ** 2)
    if(mse == 0):  # MSE is zero means no noise is present in the signal .
                   # Therefore PSNR have no importance.
        return 100
    max_pixel = 255.0
    psnr = 20 * log10(max_pixel / sqrt(mse))
    return psnr

def main():
     original = cv2.imread("original_image.png")
     compressed = cv2.imread("compressed_image.png", 1)
     value = PSNR(original, compressed)
     print(f"PSNR value is {value} dB")

if __name__ == "__main__":
    main()
```

SSIM(Structural Similarity Index Measure)

: 두 이미지 간의 유사성을 측정하는데 사용되는 metric

```python
from IQA_pytorch import SSIM, utils
from PIL import Image
import torch

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

ref_path  = 'r0.png'
dist_path = 'r1.png'

ref = utils.prepare_image(Image.open(ref_path).convert("RGB")).to(device)
dist = utils.prepare_image(Image.open(dist_path).convert("RGB")).to(device)

model = SSIM(channels=3)

score = model(dist, ref, as_loss=False)
print('score: %.4f' % score.item())
```

5. 평가 지표

ChromaGAN

trian data 약 1만3천 장10epoch 학습 결과

기존 weight

10epoch 학습

원본 컬러 이미지

5. 평가 지표

ChromaGAN

기존 약 2만5천장의 사진으로 3epcoh 학습 진행

흑백 이미지

기존 weight

한국 음식 이미지로 fine-tuning

원본 컬러 이미지

감사합니다!