딥러닝의 기초 Final Project - 7조

Automatic image colorization with Deep Convolutional Networks and improving performance with GAN

|  |  |  |  |
| --- | --- | --- | --- |
| 공과대학  전기정보공학부  2013-11571  정만수 | 자연대학  물리천문학부  2012-10850  강성주 | 공과대학  전기정보공학부  2017-13683  김재원 |  |

Abstract

우리는 Encoder – Decoder의 구조를 가진 Deep convolution network를 이용하여 grayscale의 이미지를 Colorize 하였다. 그러나 이러한 모델은 두 개의 문제점을 갖고 있었는데, 첫째는 사물의 경계를 제대로 인식하지 못하는 Segmentation Issue이고 둘째는 전체적인 색감이 칙칙해지는 색감 Issue이다. Segmentation Issue를 해결하기 위해서, 우리는 u-net의 구조에서 영감을 받아 Encoder와 Decoder간의 Skip Connection을 추가하였으며 이는 Segmentation Issue를 크게 개선하였다. 또한 우리는 색감 Issue를 개선하기 위해서 Objective Function을 L2 Loss에서 L1 Loss으로 바꾸는 등의 여러 방법을 시도하였으며, 최종적으로 Generator를 GAN을 통해 Fine-tuning하는 방법으로 색감 Issue를 개선하였다. 또한, GAN을 통한 Fine-tuning과정에서 Generator를 학습시킬 때 Discriminator의 결과만을 이용하기 보다는, Output과 Ground Truth간의 L1 Loss를 추가하는 것이 좀 더 바람직한 결과물을 얻는 것을 관찰하였다. 마지막으로, GAN 학습에 L1 Loss를 추가하는 것이 좋은 결과물을 만들게 할 뿐만 아니라, GAN의 학습을 돕는 다는 것을 볼 수 있었다.

* 1. Introduction

과거에는 흑백 이미지를 색칠하기 위해서 사람의 손이 필요하였다. 컴퓨터가 주어진 물체가 어떤 색을 가지는지 판별할 수 없었기 때문이다. 그러나 딥 러닝이 발전하고 CNN(Convolutional Neural Network)이 등장하면서 컴퓨터가 물체를 인식하는 것이 가능 해졌으며, 이를 이용하여 Classification, Segmentation 등의 컴퓨터 비전 분야에서 놀라운 성과를 보여주었다. Grayscale된 이미지에 색을 입히는 Colorization 또한 딥 러닝의 발전에 힘입어 등장한 분야이다.



Figure 1. Colorization 결과 이미지

Colorization의 기본 골자는, 어떤 물체의 이미지를 모델에게 주고 그 물체의 색을 학습시키는 것이다. 그러나 기본적으로 Colorization은 답이 정해져 있지 않은 ill-posed problem이다. 예를 들어, 어떤 자동차가 있을 때 그 자동차가 빨간색인지 파란색인지는 알 수 없다. 왜냐하면 동일한 기종의 자동차여도 여러가지 색을 가지기 때문이다. 그렇기 때문에 단순히 데이터를 학습시키는 Supervised Learning만으론 완전한 결과를 얻기 힘들다.

그러나 Ian Goodfellow가 제시한 GAN은 그러한 문제점을 해결해주었다, **[1]** GAN은 데이터를 직접 학습하기 보다는, Model의 결과물을 Discriminator가 진짜인지 가짜인지 판별하는 방식으로 학습한다. 비록 GAN은 학습이 불안정하고 오래 걸린다는 단점이 존재하여 아직 수많은 연구가 필요하지만, DCGAN이 보여준 것처럼 GAN이 보여준 결과물들은 놀랍도록 현실적이다. **[2]**

우리는 Deep Convolutional Network를 이용하여 주어진 Grayscale 이미지를 자동으로 색칠하는 Automatic Colorization 모델을 구현하였다. 또한 모델이 보여주는 약점들을 보완하기 위해서 모델을 Modify하고, GAN을 이용하여 Fine-Tuning하였다. [Figure 1]은 우리가 구현한 모델을 통한 Colorization 결과의 예시이다.

* 1. Related Works

우리가 처음 Colorization Net을 구현할 때에는 M. K. Luke의 모델을 많이 참고하였다. **[3]** 해당 모델은 간단한 Encoder–Decoder의 구조를 가지고 있다. Encoder로는 Imagenet Dataset에 대해 Pre-train된 Resnet18을 사용하였으며, Objective Function으로 L2 Loss를 사용하였다.

그러나 M. K. Luke의 모델은 두 가지 문제점을 갖고 있었다. 첫째는 물체의 테두리를 따라서 색칠하기보다는 물체가 있을 법한 위치에 원형으로 색칠하는 Segmentation Issue이다. 물체의 테두리를 제대로 인식하지 못하는 문제는 Segmentation분야에서도 중요한 문제인데, Ronneberger O. 가 제안한 U-Net은 이 문제를 크게 개선해주었다. **[4]** U-Net은 Low level의 Feature를 Decoder에 직접 concatenate 하는 특징을 갖고 있으며, 이를 통해 모델이 물체의 테두리를 좀 더 정확하게 인식할 수 있게 되었다.

M. K. Luke의 모델이 가진 두번째 문제점은 색감이 칙칙하다는 것이다. 이는 앞서 언급했듯이 Colorization이 답이 정해져 있지 않은 ill-posed problem이기 때문이다. 이는 비단 Colorization 뿐만이 아닌 대부분의 generation problem에서 나타나는 현상이다. 그러나 CNN과 GAN을 결합한 DCGAN(Deep Convolutional Generative Adversarial Networks)은 generation problem에서 놀라운 결과물을 만들어냈다. **[2]** GAN은 Convolutional layer들로 Generator와 Discriminator을 만들어 Generator는 진짜와 같은 이미지를 만들려고 하고 Discriminator는 진짜 이미지와 Generator가 만든 가짜 이미지를 구분하려고 한다. 이 둘이 서로 경쟁적으로 학습하면서, 결과적으로 Generator는 사실적인 이미지를 만들어내게 된다. DCGAN은 이러한 GAN의 구조를 Convolutional layer들로 구성한 것이다.

* 1. Our Approach
     1. Dataset Selection

우리는 시간 및 자원적 제약으로 인해 Imagenet과 같은 대용량의 Dataset에 대해 학습시키는 것은 현실적으로 불가능이라 판단하여, 비교적 양이 적은 Dataset인 Flower102와 Pokemon Generation 1을 선택하였다. **[5]**, **[6]** Flower102는 102종의 꽃의 사진을 가지고 있는 7,370장의 Image set이며 우리는 그중 6,552장을 Training Data, 818장을 Validation Data로 사용하였다. Pokemon Generation 1은 150종의 포켓몬 사진을 가지고 있는 10,657장의 Image set이며 그중 8,501장을 Training Data, 2,606장을 Validation Data로 사용하였다. 추가적으로, 우리는 모델을 확정한 후 다른 데이터셋에 대해서도 정상적으로 작동하는지 확인하기 위해 Intel Image Classification에 대해서도 모델을 학습시켰다. **[7]** Intel Image Classification은 숲이나 산 같은 여러가지 풍경에 대한 각종 이미지를 가지고 있는 24,335장의 Image set이며 그중 14,304장을 Training Data, 3,000장을 Validation Data, 7,301장을 Test Data로 사용하였다.

* + 1. Color Space

우리는 모델을 효율적으로 훈련시키기 위해 이미지의 Color space를 RGB에서 Lab으로 바꾼 후 훈련시켰다. RGB 이미지를 Colorization을 하는 경우 R, G, B 3개의 채널의 값을 학습해야 한다. 그러나 Lab으로 변환한 이미지는 L 채널만 분리하면 그것이 곧 Grayscale 이미지가 되므로, Colorization을 위해서 나머지 2개의 a, b 채널만 학습하면 된다. 따라서 Colorization을 학습할 때, RGB보다는 Lab가 유리하다.

* + 1. Data Augmentation

학습에 사용된 데이터의 수가 매우 적기 때문에 Overfitting의 가능성이 크다. 이를 보완하기 위해 Data augmentation을 적용하였다. 우리가 적용한 Augmentation은 256x256 Randomsized Crop과 Random-HoriontalFlip이다.

* + 1. Loss Function

Generator

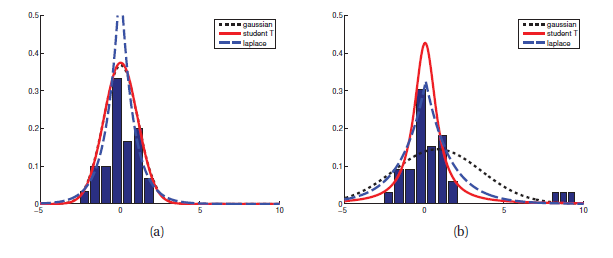


Figure 2. Outlier가 Gaussian, Student T, Laplace distribution에 대해 Model Fitting에 미치는 영향. (a) Outlier가 없을 때. (b) Outlier가 있을 때. Based on Figure 2.8 of (Murphy 2012)

Generator를 학습할 때, 처음에는 단순한 L2 Loss를 Loss Function으로 사용하였으나, L2 Loss는 이미지의 색감을 다소 칙칙하게 만드는 문제가 있었다. 이는 L2 Loss가 픽셀의 a, b값을 Gaussian Model로 Fitting하기 때문에 나타나는 현상이다. [Figure 2]는 그 현상을 잘 보여준다. 어떤 값의 히스토그램이 주로 0에 몰려 있으나 약간의 Outlier가 있을 때, Gaussian distribution의 fitting 결과가 크게 달라짐을 볼 수 있다. 한편, Laplace나 Student T Distribution은 Gaussian distribution에 비해 outlier에 대해 robust함을 볼 수 있는데, 이 점에서 착안하여 우리는 Loss Function으로써 L2 Loss 대신, Laplace Distribution에 대응하는 L1 Loss를 선택하였다. 다만 L1 Loss는 Non-Convex하여 학습시키기 어려우므로, 0 근방에서는 L1 Loss 대신에 L2 Loss를 사용하는 Smooth L1 Loss를 사용하였다. 다음식은 Smooth L1 Loss의 식이다.

where is given by

GAN

기존의 GAN 모델에서 사용하는 Objective Function은 다음 식으로 주어진다.

여기서 Discriminator는 위 Objective Function을 최대화하도록, Generator는 최소화하도록 학습한다. 그러나 GAN 모델의 경우, 결과물의 색감자체는 좀 더 화사해지나 올바른 색을 선택하지 못하는 문제가 있었다. 이를 해결하기 위해서, GAN 모델에서 Generator를 학습할 때 기존의 Loss에 L1에 일정 비율()을 곱한 후 더하였다. 그 결과, 최종적으로 우리가 GAN 모델에서 사용한 Loss는 다음과 같다.

여기서 는 0과 1사이의 Hyper Parameter이다. 만약 가 1에 가까우면 GAN을 사용하지 않은 기존의 Generator와 유사한 결과를 얻게 되고, 가 0에 가까우면 GAN만 사용한 것과 유사한 결과를 얻게 된다. 우리는 여러 번의 실험 끝에 의 값으로 0.1을 선택하였다.

* + 1. Models

Generator

[Figure 3]는 Generator의 그림이다. Generator는 Encoder와 Decoder로 구성되어 있다. Encoder는 이미지로부터 Feature를 추출하는 역할을 하며, Resnet50에서 마지막 FC Layer를 제거한 형태를 가진다. Decoder는 Encoder로부터 얻은 Feature Map을 다시 2 x 256 x 256의 ab map으로 바꾸는 역할을 하며, Convolutional Layer와 Up-sampling Layer의 연속으로 이루어져 있다. 또한 Segmentation Issue를 해결하기 위해 U-net의 구조에서 영감을 받아 Encoder와 Decoder의 사이사이에 Skip Connection을 놓았다. 이 Skip Connection들은 Low level의 Feature를 Decoder에 직접 전달하여 Encoder의 중간 결과물을 Decoder의 중간 결과물에 Concatenate한다.

GAN

[Figure 4]는 GAN의 그림이다. Generator는 [Figure 3]의 Generator와 동일한 구조를 가지고 있다. 한편 Discriminator는 Generator의 Output과 Ground Truth로부터 Real Image와 Fake Image를 구별한다. Discriminator는 원활한 학습을 위해, DCGAN 논문 [2]을 참고하여 만들어졌으며 Conv layer – BN layer – LeakyReLU의 연속이다. 마지막 레이어에서는 확률 값을 얻기 위해 Sigmoid를 통과한다.

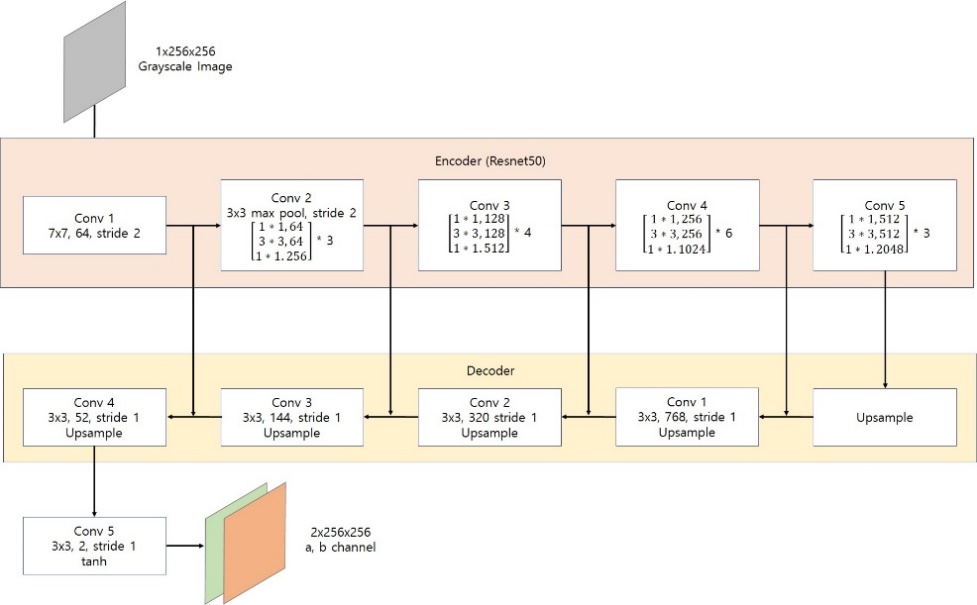


Figure 3. Generator의 그림

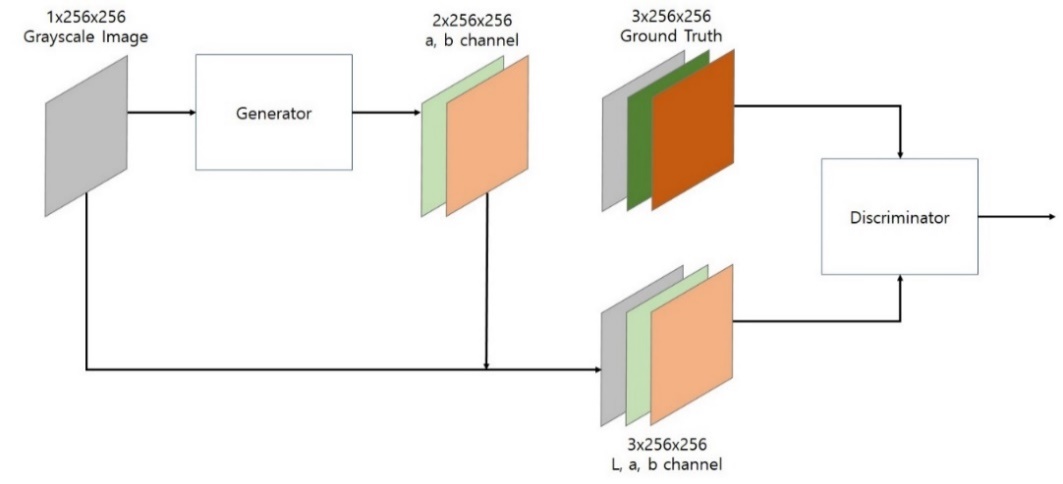


Figure 4. GAN의 그림

* + 1. Step by step training

우리는 모델 전체를 한꺼번에 학습시키는 대신 모델의 일부를 하나씩 학습시키는 방식을 취했다. 총 3번의 학습을 거치는데, 첫번째로 Encoder를 grayscale된 dataset에 대해 분류기로써 pre-train시켰다. 이는 Encoder가 grayscale된 이미지의 Feature를 좀 더 명확하게 추출하도록 만든다. 그 다음, pre-train된 Encoder를 generator에 옮긴 후 Ground Truth와 Generator의 결과물 간의 L1 Loss를 이용하여 Generator를 학습시켰다. 마지막으로, 이렇게 학습된 generator를 GAN 모델에 옮겨 Fine-tuning하였다. 이렇게 우리는 총 세단계의 학습을 거치는 방식을 취했다. 이렇게 Step by step으로 Training함으로서 모델 전체를 한꺼번에 학습시키는 것보다 학습이 훨씬 빠르고, 좀 더 바람직한 결과물을 얻을 수 있었다.

* 1. Result and Discussion
     1. Encoder - Decoder Skip Connection

우리는 U-net의 구조를 참고하여 Encoder와 Decoder 사이의 Skip Connection을 추가하였다. [Figure 5]는 Skip Connection이 있을 때와 없을 때 각각의 결과물을 보여준다. [Figure 5]를 보았을 때, Skip Connection이 없을 때는 잎사귀와 잎사귀 사이사이에도 꽃과 비슷한 색이 칠해지는 Segmentation Issue를 볼 수 있다. 그러나 Skip Connection이 추가됨으로써 그러한 문제점이 상당부분 해결되었음을 볼 수 있었다.



Figure 5. Encoder-Decoder Skip Connection이 없을 때(위)와 있을 때(아래)의 결과물

* + 1. Improving Performance with GAN

[Figure 6]과 [Figure 7]은 L1 Loss를 이용해 Generator를 학습시켰을 때(2열)와 Generator를 GAN을 통해 Fine Tuning 했을 때의 결과물을 보여준다. 또한 GAN을 통해 추가적으로 학습시킬 때는 두 가지 경우로 나누어, L1 Loss를 Generator Loss에 추가하지 않았을 때(3열)와 추가했을 때(4열)로 나누었다.

Ground Truth(1열)과 비교했을 때, GAN을 통해 추가학습한 모델은 그렇지 않은 모델보다 좀 더 진한 색감을 가지고 있음을 볼 수 있었다. 이는 Flower 102 dataset보다는 Pokemon Generation 1 dataset에서 더욱 두드러지는 현상이었다.

한편 GAN을 통해 학습할 때, L1 Loss가 더해지지 않은 경우 색감 자체는 개선되나 제대로 된 색을 선택하지 않는 것을 볼 수 있었다. 예를 들어, [Figure 6]의 5행(민들레 씨앗)을 보면 GAN에서 L1 Loss를 추가하지 않았을 때 불그스름한 결과물이 나타남을 확인할 수 있다. 이 현상은 Pokemon Generation 1 dataset에 대해서도 동일하게 나타나, [Figure 7]의 2행(이상해씨)을 보면 눈과 입 주변에 불그스름한 색깔이 있음을 볼 수 있다. 하지만 이 문제점은 GAN 학습 도중 L1 Loss를 더했을 때 상당부분 개선되었다.

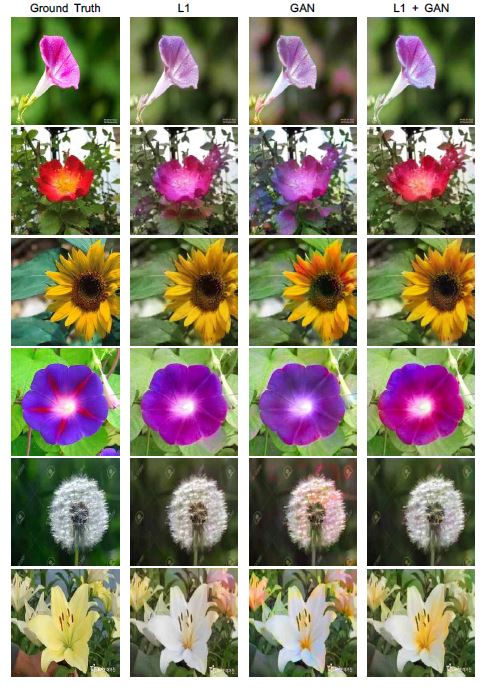


Figure 6. Flower 102 dataset에 대해 학습한 모델의 결과. 1행: Ground Truth, 2행: GAN으로 Fine-Tune 되기 전의 모델, 3행: GAN학습에 L1 Loss를 더하지 않은 모델, 4행: GAN 학습에 L1 Loss를 더한 모델

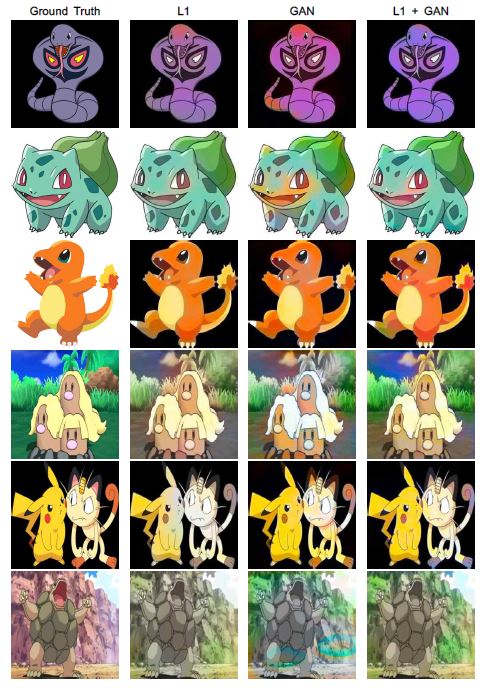


Figure 7. Pokemon Generation 1 dataset에 대해 학습한 모델의 결과. 1행: Ground Truth, 2행: GAN으로 Fine-Tune 되기 전의 모델, 3행: GAN학습에 L1 Loss를 더하지 않은 모델, 4행: GAN 학습에 L1 Loss를 더한 모델

* + 1. Test on different dataset

우리는 위와 같은 과정을 통해 모델을 확정한 후, 다른 dataset에 대해서도 정상적으로 작동하는지 확인하기 위해 Intel Image Classification dataset에 대해서도 모델을 학습시켰다. [Figure 8]은 그 결과를 보여준다.

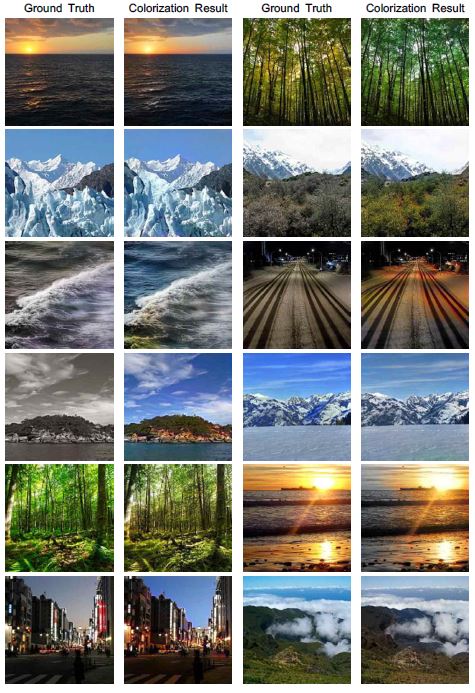


Figure 8. Intel Image Classification에 대해 학습한 결과. 1, 3행: Ground Truth. 2, 4행: Colorization Result

* + 1. Stable training of GAN

또한 흥미로운 점으로, GAN 학습 도중 L1 Loss를 더했을 때 학습이 좀 더 안정적으로 이루어졌음을 확인하였다. [Figure 9]는 Hyper Parameter를 포함한 동일한 조건 하에서, GAN 학습 도중 Generator Loss에 L1 Loss를 더하지 않았을 때(위)와 더했을 때(아래)의 Discriminator와 Generator의 Loss 곡선을 보여준다.

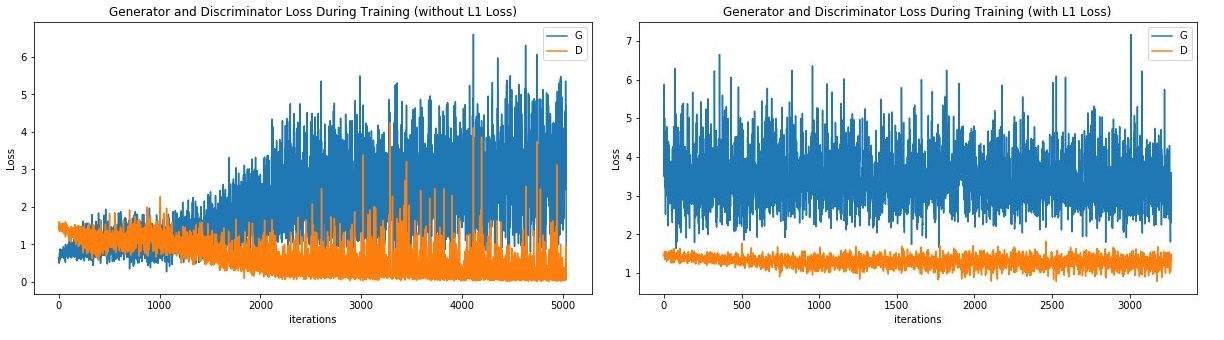


Figure 9. GAN 학습 도중 L1 Loss를 더하지 않았을 때(좌)와 더했을 때(우)의 Loss Graph

위 그래프에서, GAN 학습에 L1 Loss를 도입하지 않은 경우(좌) 학습이 진행되면서 Discriminator의 Loss는 0에 수렴하고 Generator의 Loss가 증가하는 것을 볼 수 있다. 이는 Discriminator와 Generator간의 학습균형이 깨진 것으로 볼 수 있으며, 바람직한 결과를 기대하기 어려웠다. 반면 L1 Loss를 도입했을 때(우)에는 Discriminator와 Generator의 Loss가 거의 일정하게 유지되고 있음을 볼 수 있다. GAN의 학습이 어려운 이유 중 하나가 Discriminator와 Generator가 서로 학습의 균형을 유지하는 것에 있기 때문에, L1 Loss를 도입했을 때 Loss가 일정하게 유지되는 것은 상당히 흥미로운 현상이었다. 이 현상이 나타난 정확한 이유는 아직 찾지 못하였으나, 생각되는 이유는 다음과 같다. GAN의 Discriminator Loss는 realistic 결과물을 만들게 한다. 그러나 realistic하다는 것은 상당히 많은 결과를 함축하므로, local minima가 많이 존재한다. 그러나 L1 Loss는 오직 원본이라는 일관된 결과를 만들게 하므로, GAN 학습 도중 local minima에 빠지는 것을 상당부분 해소해주는 것이다.

* 1. Conclusion and Future work

지금까지 우리는 Grayscale 이미지를 Colorize하는 Automatic Colorization Network를 구현하고, 그 모델이 가지고 있는 Segmentation Issue나 색감 Issue와 같은 몇몇 단점을 보완하였다. 또한, 우리는 이 과정에서 GAN을 좀 더 안정적으로 학습시킬 수 있는 방법을 찾아냈다.

그럼에도 불구하고 [Figure 6], [Figure 7]에서 볼 수 있듯이 Colorization Result는 Ground Truth와 비교했을 때 다소 부족한 면이 있다. 이러한 한계점을 극복하기 위해서, Model을 여러 개의 Peak를 가진 Gaussian Mixture Model로 학습하는 등의 방법이 적용될 수 있을 것이다.

마지막으로, GAN 학습에 L1 Loss를 도입했을 때 학습이 안정적으로 되는 것은 매우 흥미로운 현상으로, 학습이 불안정하다는 GAN의 단점을 상당부분 개선해주었다. 이 현상에 대한 나름의 원인을 고찰하였으나, 정확한 원인은 아직 파악하지 못하였다. 따라서, 이에 대한 추가적인 연구 또한 진행 되어야할 것이다.

Reference

1. Goodfellow, Ian, et al. (2014) "Generative adversarial nets." *Advances in neural information processing systems*.
2. R. Alec, M. Luke, C. Soumith, (2015) Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, arXiv:1511.06434.
3. M. K. Luke, <https://github.com/lukemelas/Automatic-Image-Colorization/>, 2015
4. O. Ronneberger, P. Fischer, T. Brox (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. 234-241
5. <https://www.robots.ox.ac.uk/~vgg/data/flowers/102/>, 2018
6. H. Dwivedi, <https://www.kaggle.com/thedagger/pokemon-generation-one>, 2018
7. Puneet Bansal, <https://www.kaggle.com/puneet6060/intel-image-classification>, 2018