


# Automatic Colorization of High-resolution Animation Style Line-art based on Frequency Separation and Two-Stage Generator

주파수 분리 및 이중생성자 기반의 고해상도 애니메이션 스타일 선화 자동 채색

Yeongseop Lee · Seongjin Lee

이영섭\* · 이성진†

## Abstract

In this paper, we use Generative Adversarial Networks (GAN) to address the industrial needs of auto colorization of line arts which takes enormous amount of manual labor. Auto-colorization method used in Image-to-Image conversion based on GAN has received a lot of attention due to its promising results. In this paper, we present a solution to not only colorize the line art but also transform the low resolution out image to match the resolution of the input image through two generators and frequency separation method. A high frequency components are extracted from the line, then two generators are used to colorize the image in low resolution. The high frequency component is merged with low resolution image to produce the high resolution colorized image. The resolution of final output image matches the resolution of original image while preserving the texture of the input image, whereas the other schemes reduce the output image to 512 pixels. We performed visual and qualitative evaluation using FID, PSNR, and SSIM. The FID Score of the proposed method is better than the base model by about 4 (proposed: 47.87 and base model 51.64). PSNR and SSIM of the high-resolution images are also better than the base model. PSNR and SSIM of base model is 13.01 and 0.72 whereas the proposed is 20.77 and 0.86, respectively.

## Key Words

Machine Learning, Generative Adversarial Network, Line Arts Colorization, Image Generation

## 1. 서론

선화는 스토리보드, 게임, 삽화, 애니메이션 등 다양한 미디어 산업 초기 단계에 작품 방향을 정하는데 매우 중요하다. 애니메이션과 같은 미디어 산업에서 제품화 전 콘티(또는 스토리보드)와 같이 펜 터치만을 사용한 콘티로는 색, 분위기 등을 전달하기 쉽지 않아 추가 적인 컬러 펜 터치나 컬러 콘티를 사용한다. 하지만, 선화를 채색하는 일은 Photoshop, Clip studio 등 이미지 편집 도구를 사용해야 하며 다양한 층을 아티스트가 조작하여 만들어야하기 때문에 노동집약적이며 지루한 반복 작업이다. 특히 영상으로 진행되는 애니메이션 산업에서는 90분의 상영 시간에 초당 24프레임을 사용하는 데 이 경우 약 170,000장의 동화(프레임)를 애니메이터들이 색칠해야 하므로 많은 시간과 비용이 소비된다. 그 때문에 최근 GAN(Generative Adversarial Networks)[1]을 사용해 선화 이미지를 채색하는 연구가 진행되고 있으며, Petalica Paint[2] 및 Clip Studido와 같은 상용화된 자동채색 도구들이 이러한 자동

채색 기능을 지원하려는 움직임을 보인다.

선화 채색에 사용되는 선화의 경우 그레이 스케일 이미지와 달리 질감과 음영 정보 같은 충분한 정보를 포함하고 있지 않다. 채색을 위해 사용되는 조건입력으로는 참고 이미지 또는 몇 pixel의 컬러 정보와 같은 힌트(Weak Hint)로 이루어져 있다. 때문에 선화의 자동채색은 선화의 특징을 추출하고 부족한 정보에서 질감과 음영을 생성하기 위해 이미지 분할 및 컬러화에 대한 복합적인 문제가 있기 때문에 컴퓨터 비전 영역에서 도전적인 과제이다.

선화 자동채색은 입력하는 데이터에 따라 크게 세 가지 방법이 있다. 첫째, 선화만 사용해 채색된 이미지를 생성하는 완전자동방식[3, 4], 둘째, 선화와 컬러 이미지를 사용해 선화를 입력한 컬러 이미지 스타일로 채색을 하는 스타일 변환을 통한 자동방식[5-7], 셋째, 선화와 사용자 컬러 힌트를 입력해 원하는 색으로 채색하는 방식[8-13]이다.

기존 기법 모두 GAN을 기반으로 하며, 지원하는 출력 해상도가 최대 512 pixel로 제한되어 산업에 사용되기 힘든 해상도

† Corresponding Author : Department of AI Convergence Engineering, Gyeongsang National University, Korea.

E-mail: insight@gnu.ac.kr

<https://orcid.org/0000-0003-0760-1880>

\* Department of Informatics, Gyeongsang National University, Korea.

<https://orcid.org/0000-0003-2363-7269>

Received : November 15, 2020 Revised : November 25, 2020 Accepted : November 27, 2020

를 가지고 있다. 한국의 5개의 주요 웹툰 플랫폼을 대상으로 조사해본 결과 가로 해상도가 최소 690 최대 760으로 웹에서 사용되는 이미지를 고려했을 때 기존 기법으로는 부족한 해상도를 가지고 있다. 이는 이미지 생성에 사용되는 CNN이 입력력 해상도에 따라 메모리 사용량, 연산시간이 급격하게 높아지기 때문이다. 일반적인 GPU의 메모리 사이즈가 8에서 11GB인 것을 고려하면 1,000 pixel 이상의 고해상도 이미지를 학습하고 생성하는 것은 단일 GPU에서 많은 제약이 따른다. 따라서 모델을 분할하지 않은 단일 하드웨어에서 고해상도 이미지 출력을 기존 기법에서는 지원하지 않는다. 본 연구에서는 기존 자동채색 기법들의 해상도 제한을 개선하고 높은 수준의 채색 성능을 보이기 위해 주파수 분할을 사용한 새로운 기법을 제안한다.

제안하는 기법은 3가지로 구성된다. 1. 선화 추출방식을 2가지로 진행한 선화 데이터 증식, 2. GAN을 사용한 저해상도 초안 모델과 채색을 진행하는 채색 모델을 사용한 이중생성자, 3. 고해상도 선화 이미지 채색을 위한 주파수 분할기법. 제안한 기법을 사용해 원본 이미지와의 FID, PSNR 그리고 SSIM을 통한 유사도 평가를 진행했다. 평가 결과 기존기법 [11]의 51.64 보다 낮은 47.87 FID 점수 (높은 품질) 를 기록했고 PSNR 및 SSIM은 각각 13.01, 0.72 보다 높은 20.77, 0.86 을 기록했다. 시각적인 비교 또한 색 번짐 낮고 눈, 머리카락 등과 같이 기존 선화의 질감을 보존하는 등 우수한 결과를 보였다. 본 연구의 주요기여는 다음과 같다.

- 해상도 증가에 유연하게 대처하기 위해 2단 신경망을 사용한 효율적인 채색 모델 구조
- 주파수 분할을 통한 이미지 합성으로 해상도에 자유로운 채색 기법

본 논문의 구성은 다음과 같다. 2장은 GAN과 기존 선화 이미지를 사용한 자동채색 연구를 입력에 따라 분류하고 설명한다. 3장에 서는 제안하는 데이터 증식 기법, 모델 구성, 주파수 분할기법의 합성 기법, 모델학습 방식에 대해 설명한다. 4장에서는 학습에 사용한 데이터 및 시각적, 정량적 실험과 분석 결과를 설명한다. 5장에서는 논문의 결론을 제시한다.

## 2. 관련 연구

이번 장에서는 선화 채색에 사용되는 GAN 및 CNN (Convolutional Neural Network) 기반 방법을 설명한다. 입력 데이터 형식과 관련하여 선화를 채색하는 세 가지 방법 (완전 자동 채색, 스타일 전송 또는 반자동 채색, 사용자 힌트)이 있다.

### 2.1 생성적 적대적 네트워크

생성적 적대적 네트워크(Generative Adversarial Networks, GAN) Goodfellow et al. [1] 모델은 초 해상도 (super resolution),

이미지 생성, TTS (Text To Speech), 데이터 증식 (Data augmentation) 등 데이터를 생성하는데 탁월한 성능으로 최근 많은 연구[14-16] 에서 활용되고 있다.

GAN은 데이터를 생성 하는 생성자, 데이터의 진위를 구분하는 구분자 두 개의 모델로 구성되어 서로의 목적에 반하는 적대적인 학습 Framework로 구성되어 있다. GAN은 두 개의 모델의 관계를 통해 입력하는 데이터의 분포를 생성자가 따라가게 되어 사실적인 데이터를 생성한다.

GAN은 사실적인 데이터를 생성하기에 적합하지만 두 모델 간의 균형적인 학습이 어렵고, 입력 데이터의 분포를 따라가기 때문에 원하는 데이터를 생성하기 힘든 단점이 존재한다. Radford et al. [17] (DCGAN)은 많은 실험을 통해 CNN을 사용, 생성자와 분류자의 모델 구조를 변형하고 batch normalization [18]을 적용해 성공적인 학습을 위한 GAN 모델 구조를 결정했다. Mehdi et al. [19] (Conditional GAN, cGAN) 은 GAN 데이터 생성을 조절하기 위한 연구로 생성 데이터를 위한 클래스 라벨을 추가해 학습해 GAN 생성 데이터의 조절을 진행했다.

### 2.2 완전자동 채색 기법

완전자동방식의 채색 기법은 다른 형태의 입력 없이 선화만을 사용한다[3,4]. Isola et al. [3] (Pix2Pix)는 연구[19]의 조건 입력을 사용한 cGAN 구조로 이미지 대 이미지 변환의 솔루션을 제공했다. [3]은 사실적인 이미지를 생성하기 위해  $L_1$  손실 및 적대적 손실을 결합해  $L_1$  손실만 사용한 결과에 비해 선명하고 사실적인(photorealistic) 이미지를 생성했다. Kang et al. [4]에서는 채색작업을 위한 3가지 모델을 사용한다. 각각의 모델은 실질적인 채색을 담당하는 "Low-resolution Colorizer", 전경과 배경을 분류하는 "Background Detector" 그리고 채색된 저해상도 이미지와 배경 Segment를 받아 배경을 구분하여 해상도를 복원하는 "Polishing Network"를 사용 한다. 연구[4]는 말풍선과 같은 만화의 특징을 잘 활용하였고 선화를 일괄적으로 채색할 수 있는 장점이 있다. 하지만 완전 자동으로 채색을 진행하기 때문에 원하는 부위를 채색하기에는 힘들고 출력 이미지의 크기가 256x256 pixel 해상도로 한정되는 단점이 있다.

### 2.3 스타일 변환 기반 채색 기법

스타일 변환 기반의 자동채색 기법[5-7]은 선화와 참고가 되는 컬러 이미지로 구성된 두 개의 사용자 입력 데이터를 사용한다. Furusawa et al. [5]는 사용자가 선화에서 색상 선택을 제어 할 수 있도록 만화의 참조 이미지와 대화 형 색상 힌트(색상 팔레트)를 사용한다. 채색 정보를 원작 만화에서 추출한 윤곽 정보에 합성하여 만화 페이지를 생성한다. 채색과정을 통해 색 정보를 생성하고 원본 만화 이미지에서 윤곽선을 추출하여 합성하는 구조로 효율적인 채색을 진행하였다. 하지만 사용자의 색상 정보가 직관적으로 원하는 위치에 들어가는 것이 아

니며 색 정보와 텍스트와 같은 윤곽선을 혼합하는 과정에서 이미지 질감 손상이 심하다는 단점이 있다. Zhang et al. [7]에서는 VGG16/19 [20] 구조의 네트워크를 통해 스타일 이미지를 추가해 채색을 진행했다. 모델 중간의 두 개의 “Guide Decoder” 사용함으로써 학습에서의 기울기가 사라지는 문제(Vanishing Gradient)를 방지했다. 하지만, VGG16/19 네트워크를 사용하기 때문에 네트워크모델 용량이 크고 참고 이미지를 사용하여 자동으로 채색하기 때문에 원하는 부위에 원하는 색으로 채색하기 힘들며 출력 이미지의 크기가 256x256 pixel 해상도로 한정된다.

### 2.4 사용자 힌트 입력 기반 채색

세 번째 자동채색은 선화 이미지에 사용자가 제공한 힌트를 이용하여 이미지에 특정 색을 칠하는 방식이다[8-13]. 힌트를 사용하는 연구 중 대표적으로 Ci et al. [11]가 있다. 연구[11]에서는 모델의 인공 선화(원본 컬러 이미지에서 알고리즘으로 만들어낸 선화)의 과적합(overfitting)을 막기 위해 LFN(Local Feature Net)을 사용한다. LFN는 선화의 특징을 추출해, 생성자 및 분류자의 추가 입력으로 사용한다. 하지만 Loss 계산 시 VGG16 [20] 네트워크를 사용 하므로 모델 용량이 크고 LFN을 추론 과정에서 사용해야 하는 단점이 있다. Sangkloy et al. [8]은 4가지의 다른 알고리즘을 적용해 선화를 추출했다. 다양한 방식의 선화 분포로 데이터 증식을 진행하여 선화의 과적합을 방지해 얼굴 이미지를 채색했다. Frans et al. [10]은 채색 및 음영을 생성하는 생성자를 별도로 학습했다. 생성자의 역할을 나누어 이중생성자 구조를 사용해 효과적인 채색을 진행했다. 하지만 결과물의 질이 낮고 해상도가 512x512 pixel 인 단점이 있다. 컬러 점을 힌트로 사용한 연구로는 Liu et al. [9]이 있으며 생성자 학습을 위한 Loss를 나누어 각각의 Loss 계수 항을 조절하여 학습을 진행하였다. 학습 결과 색 번짐을 효과적으로 방지하면서 Pix2Pix [3] 모델보다 좋은 이미지를 보여주었다. HATI et al. [13]은 연구[11]의 모델 기반으로 생성자를 구성했다. 선화를 생성하는 모델을 사전 학습하여 실제 선화와 채색 모델에서 생성된 결과물의 선화의 손실을 고려하여 채색 모델의 성능을 높였다. Zhang et al. [12]은 이중생성자 구조에서 2단계 모델의 초안(1단계) 의존도를 줄이기 위해 생성된 초안 이미지에 색 번짐 등과 같은 인공물 시뮬레이션을 적용해 후처리 모델의 초안 의존도를 줄이며 채색 성능을 높였다.

## 3. 제안 기법의 구성

제안하는 시스템의 전체 구조는 그림 1과 같다. 제안하는 시스템은 채색을 진행하는 Model(Draft Model, Colorization Model) 및 입력된 선화와 채색 결과물의 저주파 성분을 이용한 주파수 분할 합성 모듈로 구성되어 있다.

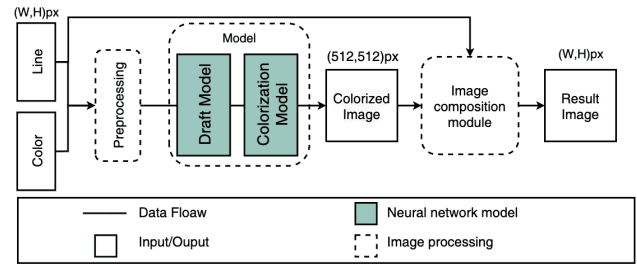


그림 1 시스템 구조 다이어그램  
Fig. 1 System Architecture Diagram

### Algorithm 1 Dilate abs sub

```

Input : color image image
Input : Kernel size K
kernel ← KxK size kernel, init one
dilated ← dilate(image, kernel) //fromOpenCV
diff ← absdiff(dilated, image)
line ← ImageToGray(255 - diff)
    
```



그림 2 전처리 입력 이미지 쌍(hint에서 회색은 alpha 값이 0)  
Fig. 2 Preprocessing Input Image Pair (In hint, gray has an alpha value of 0)

### 3.1 데이터 전처리

모델 학습은 선화, 컬러 쌍으로 구성된 이미지가 필요하다. 이를 위해 컬러 일러스트에서 Extended Difference of Gaussians [21] (XDoG) 및 Dilate abs sub (Algorithm 1) 두 알고리즘을 임의로 사용해 선화를 추출했다. 학습에 사용된 이미지 페어는 각 이미지 쌍을 512x512 잘라내어 사용했으며 초안 모델(Draft Model)의 경우 256x256 사이즈로 크기를 조절해 사용했다. 데이터 증식(data augmentation)은 선화 추출과정에서 두께에 대한 변화를 만들기 위해 XDoG의 경우  $\alpha$  값을 0.3, 0.4, 0.5로 Dilate abs sub의 커널 사이즈를 4x4, 5x5 로 임의의 조정해 두께에 대한 다양한 조건을 생성했다.

원하는 색으로의 채색을 하기 위해 생성자의 조건입력으로 컬러 힌트를 사용했다. 컬러 힌트는 이진 마스크를 생성해 컬러 이미지에서 픽셀을 유출하는 방식으로 생성했다. 그 후 이진 마스크는 힌트의 alpha 채널로 추가하여 유출된 영역은 0, 유출되지 않은 영역은 1로 사용한다. Alpha 채널을 제외한 다른 채널은 -1에서 1로 정규화해 학습을 위한 데이터를 생성한다. 그림 2에서 볼 수 있듯 선화 추출기법(XDoG, Dilate abs sub)은 사실적인 스케치 이미지를 생성한다.

### 3.2 초안 채색 단계

본 논문에서 제안하는 생성자인 채색 모델의 구조는 초안(Draft)을 만들고 초안을 사용해 채색(colorized)해 선화와 합성

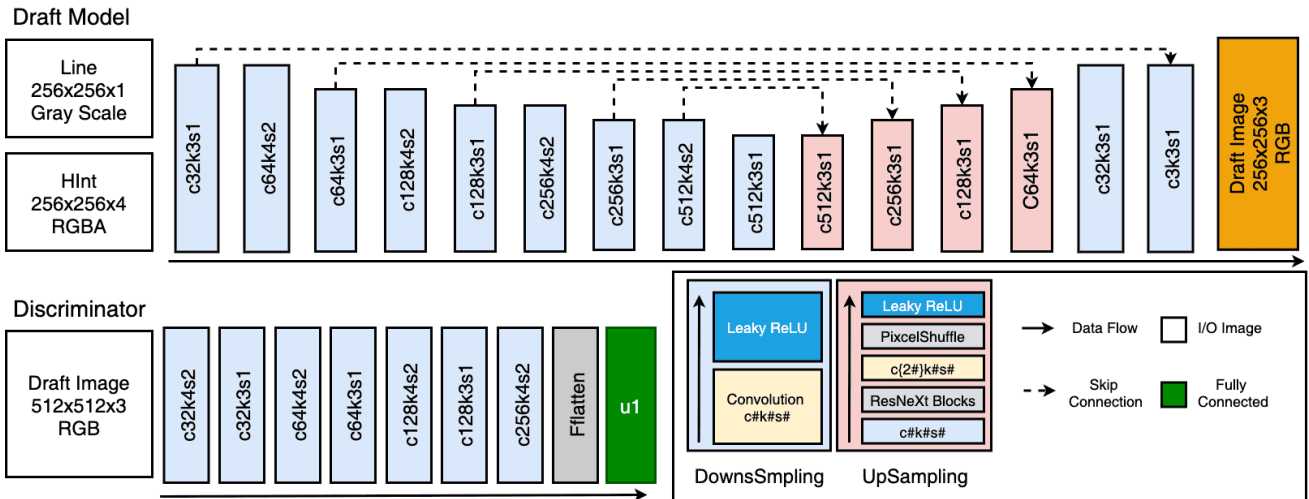


그림 3 초안 채색 모델 구조 (c: 출력 필터 개수, u: 출력 유닛 개수, k: 커널 크기, s: 보폭) 예를 들어 c32k3s1은 convolution 층의 출력 필터가 32개, 커널 사이즈는 3, Stride는 1이다.

Fig. 3 Draft Model Architecture(c:Output filter num, u:Output unit num, k:Kernel size, s:Stride) For example, c32k3s1means inter convolution layer Output Filter Number is 32, Kernel Size is 3, and Stride is 1

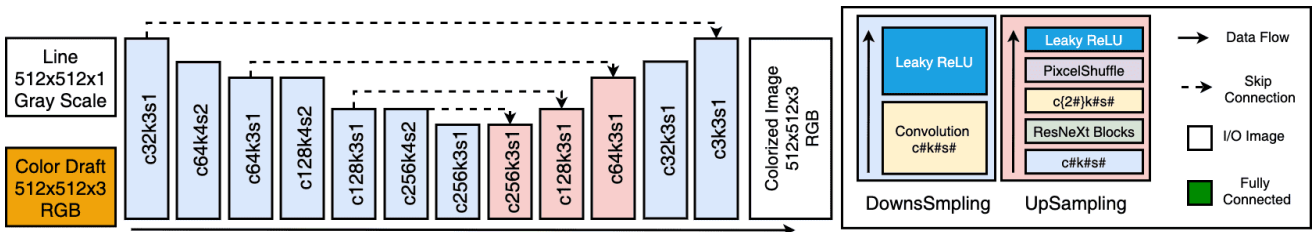


그림 4 채색 모델 구조 (c: 출력 필터 개수, u: 출력 유닛 개수, k: 커널 크기, s: 보폭)

Fig. 4 Colorization Model Architecture(c:Output filter num, u:Output unit num, k:Kernel size, s:Stride)

하는 세 과정을 가지고 있다. 초안 모델(Draft Model, 그림3)은 입력받은 선화와 사용자 힌트를 사용해 저 해상도(256x256) 컬러 초안을 만든다. 초안 모델은 고품질의 결과를 생성할 필요가 없지만, 채색 모델에서 참고할 풍부한 색을 예측한다. 초안 모델은 기존 다양한 선화 채색연구에 활용된 이미지 대 이미지 변환에 주로 사용되는 U-Net [22] 구조를 사용했다.

Up-Sampling을 구성하는 내부 모델은 transpose convolution의 체커보드 인공물(checkerboard artifacts)[23]을 해결하기 위해 Shi et al. [24]의 sub pixel convolutional (pixel shuffle 혹은 depth to space)을 사용해 해상도를 높였다. Up sampling를 구성하는 내부 모델은 ResNeXt block [25]을 사용해 계산 복잡도를 크게 늘리지 않으면서 네트워크 용량을 늘렸다. 모델에 사용된 ResNeXt block은 각 Upsampling 층당 10개를 사용했다. 생성자 모델에서[11,25,26]의 연구를 참고하여 채색의 정확도를 높이고 출력 데이터의 범위 유연성을 유지하기 위해 normalization layer [18,27]를 사용하지 않았다. 마지막으로 tanh 활성화 함수를 사용하는 마지막 층을 제외한 모든 내부 모델은 기울기가 0.2인 Leaky ReLU 활성화 함수를 사용했다.

초안 모델에서 풍부한 색을 예측하는 목표를 위해 GAN

[1] 구조로 적대적 학습 메커니즘을 사용한다. 초안 모델을 학습하는데 사용된 분류기는 연구[17]의 분류기와 유사한 구조를 가지고 있다. 연구[17]에서 제안된 strided convolution를 사용해 stride 2, kernel size 4의 CNN으로 차원을 줄이고 fully connected layer를 사용해 1개의 확률 벡터로 출력(그림 3의 Discriminator의 u1)하는 구조를 가진다. 초안 모델의 분류기는 마지막 층을 제외한 모든 내부 모델(DownSampling, UpSampling)에서 기울기가 0.2인 Leaky ReLU 활성화 함수를 사용했다. 연구에서 제안하는 초안 모델과 분류기의 구조는 그림 3과 같다.

### 3.3 채색 단계

채색 단계에서는 초안 모델에서 만들어진 초안 이미지를 사용, 선화를 채색하는 작업을 진행한다. 초안의 경우 색상의 오류와 불필요한 인공물(artifacts) 등이 포함될 수 있다. 채색단계에서 초안 이미지의 의존을 줄이기 위해 연구[12]에 따라 컬러 스프레이, 색 번짐, 왜곡 등과 같은 인공물을 합성하는 단계(Artifact Stimulation)를 추가했다. 인공물이 합성된 초안 이미지는 고해상도(512x512)로 크기변환을 거쳐 채색 모델의 입력으로 사용된다. 채색 모델은 초안 모델과 동일하게 U-Net

[22] 구조로 되어 있으며 마지막 층은  $\tanh$  활성화 함수를 사용했다. 채색 모델의 구조는 그림 4를 통해 확인할 수 있다. 채색 단계에서는 초안과 달리 풍부한 색을 만들어낼 필요가 없고 이미 초안에서 채색을 위한 충분한 정보가 생성되기 때문에 GAN은 사용하지 않았다.

### 3.4 이미지 합성 모듈

합성 모듈은 입력받은 원본 선화와 채색 결과물을 사용해 해상도를 늘리는 작업을 진행한다. 결과물의 해상도를 높이기 위해 주파수 분할이라는 기법을 사용했다. 고해상도에 필요한 고주파 성분의 경우 입력된 원본 선화를 변환해 사용할 수 있으며 저주파 성분은 채색 단계에서 생성된 512x512 해상도의 채색 이미지를 활용해 생성한다. 먼저 입력받은 원본 선화 이미지를 사용해 50% 회색(밝기 값 127) 이상의 밝기를 50% 회색으로 변환해 고주파 성분을 생성한다. 이후 채색 결과물의 해상도를 원본 선화 이미지와 동일하게 크기 조정 한다. 크기 조정된 채색 결과물에 가우시안 필터를 적용해 저주파 성분을 생성한다. 생성된 고/저주파 성분을 Linear light 혼합 모드로 혼합하여 채색 결과물을 생성한다. Linear light는 혼합 색상에 따라 밝기를 줄이거나 높여 색상을 버닝(어둡게), 닷지(밝게) 한다. 광원 이 50% 회색(밝기 값 127)보다 밝은 경우 밝기를 높여 이미지가 밝아지며 어두우면 밝기를 줄여 이미지를 어둡게 한다. 여기서는 선화 영역의 밝기가 0에 가까운 값을 가지고 있어 채색 이미지에 선화를 자연스럽게 합성할 수 있다. 제안한 주파수 분할기법의 알고리즘은 Algorithm 2와 같다. 알고리즘의 각 단계 이미지는 그림 5와 같다.

Algorithm 2 Frequency separation

```

Input : colored image
Input : line image
size ← line image size
colored ← resize(colored, size)
colored ← gaussian blur to colored
for each pixel in line do
    if pixel > 127 then
        pixel ← 127
    end if
end for
image ← linearLight(colored, line)
return image
    
```

### 3.5 손실 함수

모델 학습에 사용되는 각 손실함수와 구성요소에 관해 설명한다. 먼저 초안 단계의 생성자 손실함수 식 (1)은 적대적 ( $\mathcal{L}_{GAN}$ ), 재구성( $\mathcal{L}_{recon}$ ), 콘텐츠( $\mathcal{L}_{cont}$ )의 세 가지 항의 조합으로 정의한다. 각 항의 영향력은 계수( $w_a, w_r, w_c$ )를 사용해 조정한다. 식 (1),(2),(3),(4),(5)에서 1은 추출된 선화(256x256), h는 컬러 힌트(256x256), c은 원본 컬러이미지 (256x256), D, G는 각

각 분류자 및 생성자(초안 모델)이다.

$$\mathcal{L}_{draft} = w_a \min_G \max_D \mathcal{L}_{GAN}(G, D) + w_r \mathcal{L}_{recon}(G) + w_c \mathcal{L}_{recon}(G, F) \quad (1)$$

$$\mathcal{L}_{GAN}(G, D) = E_c[\log(D(c))] + E_{l,h}[\log(1 - D(G(l, h)))] \quad (2)$$

적대적 손실( $\mathcal{L}_{GAN}$ )은 식 (2)[1,16]의 작업을 따른다. 판별자  $D()$ 는 실제 컬러 이미지 c의 확률을 추정한다.  $D()$ 의 결과는 마지막 층에 사용된 sigmoid 활성화 함수에 의해 0과 1 사이의 값을 갖는다.  $E_c$  및  $E_{l,h}$ 는 각각  $\log(D(c))$  및  $\log(1 - D(G(l, h)))$ 의 예상 값을 측정한다.  $G(l,h)$ 는 사용자가 제공 한 힌트(h)와 함께 추출 된 선화(l)를 활용하여 컬러 이미지를 생성한다.  $G(l,h)$ 는 원본 이미지와 유사한 분포를 가진 이미지를 생성하려 한다.  $D(G(l, h))$ 는 생성 된 데이터( $G(l,h)$ )의 확률을 추정한다.

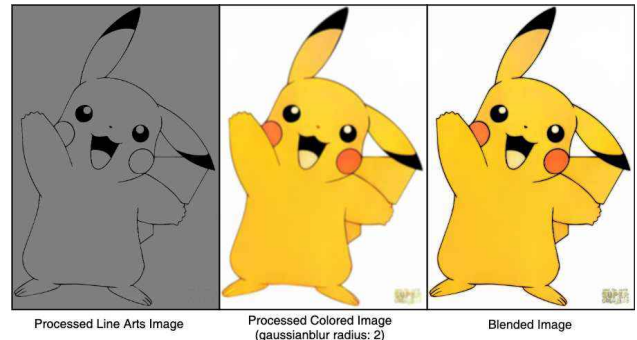


그림 5 합성된 결과물  
Fig. 5 Blending Result

이론적으로는  $D()$ 와  $G()$ 는 품질을 향상하기 위해 상호 작용하기에 충분하다. 그러나 실제로는 GAN 구조에서 두 모델 간의 균형을 이루기가 쉽지 않으며, 이러한 문제를 해결하기 위해 다른 손실 함수를 추가로 사용한다. 연구에서는 학습 과정을 안정화하기 위해 식 (3),(4)를 추가한다.

$$\mathcal{L}_{recon}(G) = E_{l,h,c}[\|c - G(l, h)\|_1] \quad (3)$$

재구성 손실( $\mathcal{L}_{recon}$ )은 식 (3)에 정의되어 있다. 재구성 손실은 실제 컬러 이미지 c와 생성 이미지  $G(l,h)$ 의  $L_1$  손실을 측정한다.  $G()$ 는 원래 색상 이미지 c의 색상 분포와 일치하도록 주어진 이미지의 색상 공간을 조정할 수 있다. 결과적으로  $G()$ 는  $D()$ 를 속일 수 있는 고품질의 이미지를 생성할 수 있다.

$$\mathcal{L}_{cont}(G, F) = MSE(F(c) - F(G(l, h))) \quad (4)$$

콘텐츠 손실( $\mathcal{L}_{cont}$ )은 식 (4)에 정의되어 있다. 콘텐츠 손실은 생성된 이미지  $G(l,h)$ 와 실제 컬러 이미지 c를 사용한  $F$  특징 맵의  $L_2$  손실(평균제곱오차, MSE)를 사용한다.  $F$ 는 ImageNet

[28] 데이터로 훈련된 VGG16 [20] 모델의 네 번째 convolution layer에서 만들어지는 특징 맵을 나타낸다. 콘텐츠 손실은 픽셀 공간에서 표현할 수 없는 이미지의 특징을 피쳐 맵을 통한 시각적 유사성을 포착하여 손실을 측정한다.

$$\mathcal{L}_{color}(G', G) = E_{l, l', h, c'} [\|c' - G'(l', \text{resize}(G(l, h)))\|_1] \quad (5)$$

채색 모델 학습에 사용되는  $L_{color}$  는 G의 학습이 끝난 뒤  $G(l, h)$  (초안 이미지)를 512x512 해상도로 크기 조정하여 실제 컬러 이미지 간의 L1 손실을 사용한다. 이 단계에서는 적대적 손실을 사용하지 않으며,  $G(l, h)$ 으로 생성된 풍부한 색을 정교하게 재생성하는 작업을 진행한다. 식 (5)에서  $G'$ 는 채색 모델,  $l$ 은 선화(256x256),  $l'$ 은 선화(512x512),  $h$ 는 힌트,  $c'$ 는 컬러 이미지(512x512)를 표현한다.

## 4. 실험과 분석

### 4.1 데이터 셋

애니메이션 스타일 일러스트 데이터 셋으로 알려진 Danbooru [29] 데이터 셋은 비율을 유지하기 위한 패딩, 작은 해상도 등 작업에 방해되는 요소가 많이 있다. 본 연구에서는 학습을 위해 대규모 일러스트 데이터 셋을 직접 수집했다. 데이터는 shuushuu-imageboard [30]를 통해 수집하였으며 수집 후 학습에 악영향을 끼칠 수 있는 부적절한 데이터를 필터링하여 약 70만 장의 컬러 애니메이션 일러스트와 500개의 실제 선화-컬러 일러스트 데이터 쌍을 수집했다. 필터링 된 부적절한 데이터는 흑백, 하이/로우 키 이미지, 그림의 작은 변이 512 pixel 이하의 작은 이미지, 전반적인 톤 혹은 색이 한쪽으로 편향된 이미지, 일러스트가 아닌 낙서 그리고 현실의 사물이 혼합된 이미지이다.

### 4.2 실험 환경

우리는 제안한 신경망 모델 구현을 위해 PyTorch framework [31]를 사용했다. 모델 학습은 한 장의 NVIDIA RTX 2080Ti를 사용했으며 학습을 위해 사용한 Hyperparameter는 초안 모델의 경우 최적화 함수로 Adam [32]을 사용했고  $\beta_1$ : 0.5,  $\beta_2$ : 0.9, learning rate:0.0001을 사용했다. Learning rate의 경우 112k 지점에서 0.1배 감소시켜 학습하였고 학습에 사용된 batch size는 64로 총 280K step을 진행했다. 초안 모델 손실의 각 가중치는 각각  $w_a$ : 0.05,  $w_r$ : 1.0,  $w_c$ : 0.1을 사용했다. 채색 모델 또한 최적화 함수로 Adam을 사용했으며 Hyperparameter는 초안 모델과 같다.

실험을 위해 제안한 신경망을 그림 6과 같이 Open Source 이미지 편집 도구인 GIMP의 Plugin으로 개발했다. GIMP는 Photoshop과 유사한 레이어 시스템과 이미지 혼합을 지원하기 때문에 제안한 이미지 처리 기법을 손쉽게 사용할 수 있으며

훌륭한 Front-End로 고수준의 사용자 경험을 제공할 수 있다. GIMP Plugin은 PyTorch로 사전 학습된 제안 모델을 ONNX (Open Neural Network Exchange) [33]로 변환하여 사용했다. 실험에 사용된 자세한 사양 및 프레임워크 버전은 표 1과 같다.

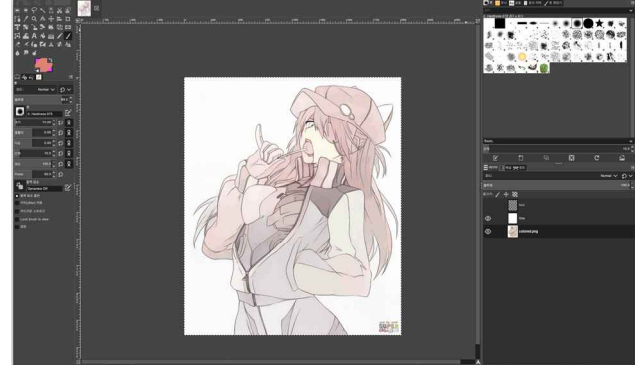


그림 6 GIMP 플러그인의 프론트 엔드  
Fig. 6 GIMP Plugin Front End

표 1 실험 환경

Table 1 Test Environment

HW	Specification	SW	Version
CPU	Intel i7-7800X	Python	3.8.5
RAM	64GB	Pytorch	1.6
OS	Arch Linux	ONNX	1.5

### 4.3 채색 성능의 시각적 분석

제안한 기법은 채색의 해상도 제약을 없애기 위해 채색 단계 및 선화 합성 단계를 나누어 진행했다. 채색 단계에서 생성된 결과물은 그림 7을 통해 확인 할 수 있다. 제안한 기법을 사용하지 않은 그림 7(a)은 낮은 채도의 결과물이 생성되며 색이 번지는 경향을 확인할 수 있다. 제안한 기법을 사용하지 않은 그림 7(b)은 transpose convolution의 checkerboard artifacts 현상을 확인할 수 있고, 일부 선화 분포에 대해 채색이 불안정한 것을 보여준다. 제안하는 기법 그림 7(c)은 높은 채도를 가지며 채색 영역에서 색이 번지는 현상 또한 방지하는 것을 확인할 수 있다.

선화 합성 단계의 결과물은 그림 8을 통해 확인 할 수 있다. 해상도의 차이를 보기 위해 짧은 변을 기준으로 1,500 pixel 이상의 선화로 채색한 후 주파수 분할을 통한 선화 합성을 진행했다. 제안한 기법을 사용하지 않은 그림 8 (a)와 (b)보다 입력한 선화를 고주파 성분으로 합성한 (c)가 머리카락 및 눈동자 선과 같은 질감을 잘 복원하였다.

### 4.4 채색 성능의 정량적 분석

사용한 기법과 기존 제안된 기법을 사용한 각 이미지의 정량적 분석을 위해 Fréchet Inception Distance (FID) [34]를 사용했다. FID는 두 이미지 데이터 셋의 유사성을 측정한다. 시각

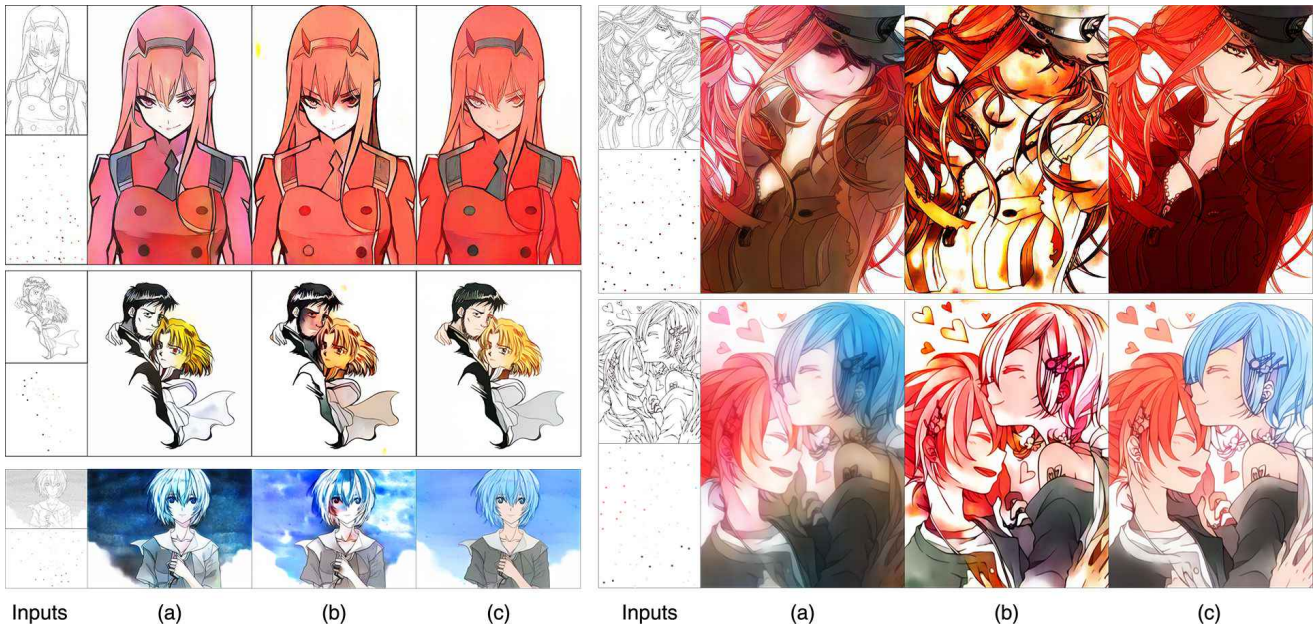


그림 7 시각적 비교 (a):Base [11], (b):Pix2Pix [3], (c):Ours  
 Fig. 7 Visual comparison. (a):Base [11], (b):Pix2Pix [3], (c):Ours

적 품질에 대한 인간의 판단과 잘 연관되는 것으로 나타났으며 주로 GAN으로 생성된 샘플의 품질을 평가하는 데 자주 사용된다. FID는 ImageNet [28] 데이터로 사전 훈련된 Inception 모델을 통해 생성된 실제 이미지와 생성된 이미지 간의 두 정규분포 특징 맵 사이의 Fréchet 거리를 사용해 계산된다. FID 점수는 낮을수록 높은 품질을 가진다. 평가에는 14 만 장의 XDoG 인공 선화와 530장의 실제 선화-일러스트 쌍을 사용해 각각 비교했다. FID를 사용한 평가는 공정성을 위해 연구 [11,13]와 동일하게 힌트를 사용하지 않은 결과물을 비교하는 방식을 사용했다.

는 PSNR (Peak Signal to Noise Ratio) 및 SSID (Structural SIMilarity)를 사용해 이미지의 손실 정보를 평가했다. PSNR은 최대 신호 전력 및 그 품질에 영향을 주는 노이즈 간의 비율을 계산하는 데 사용된다. log 항으로 계산되기 때문에 dB 형식으로 나타낸다. PSNR은 손실이 적을수록 높은 수치를 나타낸다. SSID는 휘도와 명암비를 고려한 이미지의 구조적인 차이를 포함하여 계산한다. SSID는 1에 근접할수록 높은 품질(원본 이미지와 유사한 이미지)을 가진다. 평가에 사용한 이미지는 짧은 변을 기준으로 1,500 pixel 이상의 고해상도 이미지를 채색하여 원본 사이즈로 크기 조정해 원본 이미지와 비교하였다.



그림 8 시각적 비교 (주파수 분할). (a):Base [11], (b):Pix2Pix [3], (c):Ours  
 Fig. 8 Visual Comparison(Frequency Separation). (a):Base [11], (b):Pix2Pix [3], (c):Ours

주파수 분할 기법의 평가를 위해 이미지 품질평가에 사용되

표 2 FID, PSNR, SSIM를 사용한 정량적 비교

Table 2 Quantitative comparison of FID (lower is better), PSNR (higher is better) and SSIM (higher is better)

Model	FID		PSNR		SSIM	
	mean	std	mean	std	mean	std
Base [11]	51.64	1.36	13.01	5.14	0.73	0.21
Pix2Pix [3]	57.47	3.93	13.55	2.71	0.79	0.11
Ours	<b>47.87</b>	<b>2.71</b>	<b>20.77</b>	<b>3.62</b>	<b>0.86</b>	<b>0.09</b>

FID 평가 결과표 2에서 제안한 기법(Ours)을 사용한 경우 가장 낮은 FID 을 가지고 있으며, checkerboard artifacts [23]의 현상이 나타난 Pix2Pix 기반의 모델은 Base [11]에 비해 높은 FID(낮은 품질)를 기록하였다. 주파수 분할기법을 사용한 제안 기법의 경우 PSNR 및 SSIM에서 다른 두 기법(Base, Pix2pix)에 비해 20.77(PSNR), 0.86(SSIM) 높은 점수로 우수한 결과를 보여주었다. 높은 점수를 받은 이유는 기존 모델과 비

교해 채색의 정확도 및 색 번짐 등의 인공물이 적게 나타나 PSNR에서 우수한 점수를 받았다. 또한 선화의 질감을 유지하는 시각적인 결과와 비교해 생각했을 때 SSIM 평가 점수에서도 다른 기법과 비교해 높은 점수를 얻었다.

## 5. 결론

본 연구는 기존 선화 자동 채색 기법들이 최대 512x512 pixel로 산업 수준보다 낮은 해상도를 가지는 문제를 해결하기 위해 이중생성자 및 주파수 분할 기법을 제안했다. 이중생성자는 채색의 단계를 초안 및 채색으로 역할을 나누어 각각의 생성자를 학습한다. 초안 모델 및 채색 모델은 각각 풍부한 채색을 진행하고, 채색 과정에서 생성되는 다양한 인공물을 제거하여 깔끔한 채색을 진행한다. 원본 선화의 질감을 보존하며 사용된 선화 해상도로 채색하기 위해 사용한 주파수 분할은 고주파(입력한 선화)와 저주파(채색 모델에서 생성된 채색 이미지)를 사용해 해상도를 늘려 1500 pixel 이상의 고해상도 선화 이미지 채색을 가능하게 한다.

제안한 기법의 시각적인 비교 결과는 그림 7, 8을 통해 확인할 수 있다. 비교 결과 제안한 이중생성자 방식을 사용하지 않을 경우 색 번짐, 체커보드 인공물, 잘못된 색상 등 비정상적인 채색 결과를 얻었다. 반면 제안한 기법을 사용한 결과는 색 번짐과 같은 현상이 적게 보이고 정확한 채색이 가능했다. 낮은 해상도 문제를 해결하기 위해 사용한 주파수 분할 방식의 선화 합성을 사용하지 않은 기존 이미지에서는 눈, 머리카락 등 이미지의 품질을 볼 수 있는 질감 정보가 사라지는 현상을 보였다. 해상도를 늘리는 과정에서 CNN을 사용하지 않기 때문에 기존 기법에서 지원하지 않은 2,000 pixel 이상의 초고 해상도 이미지 또한 채색할 수 있었다. 정량적 평가를 위해 FID, PSNR 그리고 SSIM을 사용해 채색된 결과물을 비교했다. 비교 결과 생성된 이미지의 품질을 나타내는 FID 평가에서 기존 기법[11]의 51.64보다 낮은(높은 품질) 점수인 47.87을 보여주었으며 PSNR 및 구조적 유사도를 나타내는 SSIM 평가 역시 13.01, 0.72보다 높은(높은 품질) 20.77, 0.86을 보여주었다. 시각적, 정량적 평가 결과로부터 이중생성자 및 주파수 분할을 사용한 채색 기법은 고해상도 이미지 채색 품질을 향상한다는 결론을 도출할 수 있다.

### Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1G1A1100455).

### References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D.

Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, pp. 2672-2680, 2014.

[2] pixiv inc., "Petalica paint." [https://petalica-paint.pixiv.dev/index\\_en.html](https://petalica-paint.pixiv.dev/index_en.html), 2019. [Online; accessed 2020.11.23].

[3] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125-1134, 2017.

[4] S. Kang, J. Choo, and J. Chang, "Consistent comic colorization with pixel-wise background classification," *NIPS'17 Workshop on Machine Learning for Creativity and Design*, 2017.

[5] C. Furusawa, K. Hiroshiba, K. Ogaki, and Y. Odagiri, "Comicolorization: semi-automatic manga colorization," *SIGGRAPH Asia 2017 Technical Briefs*, pp. 1-4, 2017.

[6] P. Hensman and K. Aizawa, "cgan-based manga colorization using a single training image," *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 3, pp. 72-77, IEEE, 2017.

[7] L. Zhang, Y. Ji, X. Lin, and C. Liu, "Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan," *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, pp. 506-511, IEEE, 2017.

[8] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5400-5409, 2017.

[9] Y. Liu, Z. Qin, Z. Luo, and H. Wang, "Auto-painter: Cartoon image generation from sketch by using conditional generative adversarial networks," *arXiv preprint arXiv:1705.01908*, 2017.

[10] K. Frans, "Outline colorization through tandem adversarial networks," *arXiv preprint arXiv:1704.08834*, 2017.

[11] Y. Ci, X. Ma, Z. Wang, H. Li, and Z. Luo, "User-guided deep anime line art colorization with conditional adversarial networks," *Proceedings of the 26th ACM international conference on Multimedia*, pp. 1536-1544, 2018.

[12] L. Zhang, C. Li, T.-T. Wong, Y. Ji, and C. Liu, "Two-stage sketch colorization," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1-14, 2018.

[13] Y. Hati, G. Jouet, F. Rousseaux, and C. Duhart, "Paintstorch: a user-guided anime line art colorization tool with double generator conditional adversarial network," *European Conference on Visual Media Production*, pp. 1-10, 2019.

[14] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., "Photorealistic single image super-resolution using a generative adversarial network," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681-4690, 2017.

[15] M. Bińkowski, J. Donahue, S. Dieleman, A. Clark, E.



- Elsen, N. Casagrande, L. C. Cobo, and K. Simonyan, "High fidelity speech synthesis with adversarial networks," arXiv preprint arXiv:1909.11646, 2019.
- [16] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation using gan for improved liver lesion classification," 2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018), pp. 289-293, IEEE, 2018.
- [17] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," arXiv preprint arXiv:1511.06434, 2015.
- [18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.03167, 2015.
- [19] B. Dai, S. Fidler, R. Urtasun, and D. Lin, "Towards diverse and natural image descriptions via a conditional gan," Proceedings of the IEEE International Conference on Computer Vision (ICCV), Oct 2017.
- [20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.
- [21] H. Winnemöller, J. E. Kyprianidis, and S. C. Olsen, "Xdog: an extended difference-of-gaussians compendium including advanced image stylization," Computers & Graphics, vol. 36, no. 6, pp. 740-753, 2012.
- [22] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," International Conference on Medical image computing and computer-assisted intervention, pp. 234-241, Springer, 2015.
- [23] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," Distill, vol. 1, no. 10, p. e3, 2016.
- [24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1874-1883, 2016.
- [25] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.
- [26] S. Nah, T. Hyun Kim, and K. Mu Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3883-3891, 2017.
- [27] Y. Wu and K. He, "Group normalization," Proceedings of the European conference on computer vision (ECCV), pp. 3-19, 2018.
- [28] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248-255, June 2009.
- [29] G. B. Danbooru community and A. Gokaslan, "Danbooru 2017: A large-scale crowdsourced and tagged anime illustration dataset." <https://www.gwern.net/Danbooru2017>, 2019. [Online; accessed 2020.11.23].
- [30] some, "E-Shuushuu-Kawaii Image Board." <https://e-shuushuu.net/>, 2018. [Online; accessed 19-July-2018].
- [31] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.
- [32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.
- [33] J. Bai, F. Lu, K. Zhang, et al., "Onnx: Open neural network exchange." <https://github.com/onnx/onnx>, 2019.
- [34] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," Advances in neural information processing systems, pp. 6626-6637, 2017.

---

#### 저자소개

---

##### Yeongseop Lee



Yeongseop Lee graduated from Gyeongsang National University in 2020. He is pursuing master degree at the Dept of Information Science, Gyeongsang National University. His research interests includes Machine Learning, Neural Network, Image Generation, and Image Processing.

---

##### Seongjin Lee



Seongjin Lee graduated from Hanyang University in 2006. He received Master and Ph.D. degree in the same university in 2008 and 2015, respectively. He worked as postdoc in Storage Center Hanyang University till 2017 and became an assistant research professor there. He joined Gyeongsang National University in 2017 as an assistant professor. His research interest includes Operating System, Storage System, System Optimization, Avionics, and Machine Learning.