[0]

Image generator조 미니 프로젝트 발표 시작하도록 하겠습니다. 저희는 총 2가지의 프로젝트를 진행해보았는데요, 첫번째는 GAN을 이용해 새로운 한글 인쇄체 이미지를 만들어내는 프로젝트이고, 두번째는 CycleGAN을 이용해 실제 사진을 다른 스타일의 그림으로 바꾸는 프로젝트입니다.

[1]

혹시 드라마 스타트업을 보셨다면, AI가 손글씨를 폰트로 만들어주는 프로젝트가 나오는 것을 볼 수 있는데요, 이와 비슷한 원리로 저희는 AI를 이용해 기존 인쇄체 이미지를 학습시켜 새로운 인쇄체를 만들어 내는 프로젝트를 진행했습니다. 이 프로젝트의 목표는 글자의 특징을 잘 학습해서 비슷한 스타일의 다른 글자들을 생성해낼 수 있도록 모델을 학습시키는 것입니다.

[2]

저희는 이러한 프로젝트를 진행하기 위해서 AI Hub의 한국어 글자체 이미지 AI데이터를 이용하였습니다. 이 데이터는 현대 한글 11,172자를 가장 많이 활용하는 폰트 50종을 선정하여 얻어 낸 글자체의 이미지입니다.

[3]

학습을 위해 이용한 모델은 논문 스터디 시간에 공부했던 GAN 모델입니다. GAN은 이미지를 생성해내는 Generator 부분과, 진짜 이미지인지 가짜 이미지인지 검출해내는 Discrimitor 부분으로 구성됩니다. Discriminator는 이미지를 입력받으면 진짜인지 가짜인지에 대한 확률값을 출력하고, 출력된 확률값과 정답을 비교해 얼마나 잘 맞췄는지 피드백을 받으면서 점점 똑똑한 검사기가 됩니다. 반면, Generator는 Discriminator 를 속이는 것이 목표입니다. 처음에는 제대로 된 이미지를 생성하지 못하던 Generator는 Discriminator를 속일 수 있을만큼 점점 더 진짜같아 보이는 이미지 생성해내도록 학습합니다. 이렇게 Discriminator와 Generator가 서로 겨루며 학습하는 것이 GAN 학습의 기본 원리입니다. 즉, GAN은 아무 의미가 없는 noise 벡터를 입력으로 받으면, 의미가 있는 이미지를 생성해내는 것이 최종 목표입니다.

[4]

이제 구체적인 진행과정에 대해서 말씀드리도록 하겠습니다. 데이터는 총 50만장의 이미지 데이터들 중 학습시간 등을 고려하여 약 6만장을 랜덤하게 선택해주었습니다. 이렇게 선택한 이미지들을 opencv를 이용해 불러와준 뒤 32x32로 이미지 사이즈를 통일시켜주고, 각 이미지의 픽셀 값을 -1과 1사이의 범위로 정규화 시켜주었습니다. 오른쪽에 보이는 그림이 전처리한 이미지 데이터 100개를 시각화한 이미지입니다.

[5]

이제 gan모델의 두가지 모델을 생성해주었습니다. 먼저 생성자 모델을 구성을 해주었는데, 생성자 모델은 input값으로 노이즈 100개를 받아들이게 되고, 최종 사이즈는 학습 이미지 데이터와 동일하게 32x32x3으로 만들어 줌으로써 그 결과가 판별자의 input으로 들어갈 수 있도록 해줍니다. 오른쪽에 보이는 이미지는 아직 학습이 전혀 되지 않은 생성자 모델에 노이즈를 제공한 것인 것인데, 결과를 보면 한글 이미지와는 전혀 상관없는 그냥 무작위 이미지인 것을 볼 수 있습니다.

따라서 이러한 generator가 한글과 유사한 이미지를 생성하기 위해서는 판별자 모델이 필요합니다. 판별자 모델은 우리가 지금까지 봐왔던 CNN구조와 유사한 구조를 가지고 있는데, 결과값으로는 이미지가 진짜인 확률 값을 내보내게 됩니다.

아래 보이시는 코드가 전에 학습되지 않은 생성자 모델에서 생성한 무작위 이미지를 마찬가지로학습되지 않은 판별자 모델에 넣어준 결과로 -0.0029라는 값이 나와있는 걸 볼 수 있는데요 이 값이 나중에 학습을 진행해주면, 가짜는 0에 가깝게, 진짜는 1에 가깝게 예측을 잘하게 됩니다.

[6]

이제 네트워크를 구성했기 때문에 Optimizer와 loss function을 지정해주었는데요 앞에서 모델을 두개만들었기 때문에 각각의 optimizer를 설정해주었습니다. Gan 모델의 loss function을 수식으로 표현하면 다음과 같은 것을 살펴보았는데, 먼저 생성자 모델의 목표는 판별자 모델이 가짜이미지를 판별했을 때 판별값이 1에 가까워지도록 하는 것이기 때문에 이와 같이 fake output 값과 1의 값을 비교해 그 error가 작아지도록 학습하고, 판별자 모델의 경우에는 진짜이미지를 1에 가까워지도록, 가짜이미지를 0에 가까워지도록 하는 것이 loss function의 목표이기 때문에 다음과 같이 real\_output값과 1을, fake output값과 0을 비교해 그 error가 작아지도록 학습하였습니다.

[7]

최종적으로 학습을 모두 거친 결과입니다. 먼저 에폭이 0일 때 생성자가 만들어낸 이미지를 보면, 한글과는 거리가 먼 랜덤한 값들이 뿌려진 이미지의 형태이지만, 300에폭까지 학습을 진행한 결과, 한글의 이미지와 꽤 유사해진 모습을 확인할 수 있었습니다.

[cyclegan 설명]

Cyclegan의 loss function은 다음과 같습니다.

그림에서 실제 사진처럼 만들 때의 Generation loss와 그리고 이 생성된 이미지랑 실제 사진과의 L1 loss 를 더한 값입니다. 또한 이 반대의 경우도 동일하게 사진을 그림으로 만들었을 때의 GAN loss와 실제 그림과 사진과의 L1 loss를 구해서 더한 값이 cyclegan의 loss가 됩니다.