Img2Caricature

2014104140 임호준

최강혁

개 요

딥 러닝은 무서운 속도로 발전하고있다. 얼마 전까지만 해도 딥 러닝 기술은 대부분 데이터를 분류하거나 인식하는 데 초점이 맞춰져있었지만, 이제는 데이터를 생성하는 모델도 발전하고 있다. 이러한 생성 모델을 이용하면 기존의 인력으로 생산하던 수 많은 데이터를 컴퓨터를 통해 자동으로 생성할 수 있다. 특히 딥 러닝을 응용한 방식은 데이터를 생성하는 방법을 직접 명시하는 것이 아닌, 기존의 데이터를 이용해 데이터를 생성하는 방법을 학습하기 때문에, 적절한 네트워크 모델과 학습 방법에 대한 연구만 완료된다면 데이터를 생성하는 방법을 프로그래밍 할 수고를 덜 수 있다.

생성 모델을 응용하여 인간의 창작활동을 모방해보는 것이 이번 프로젝트의 목표이다. 구체적으로는 실사이미지에 포함된 사람의 이미지를 캐리커쳐로 변경하는 GAN 을 이용한 딥 러닝 모델을 설계하고, 모델을 학습하는 과정에 사용할 수 있는 다양한 기법들에 대해 연구를 진행한다.

1. 서론

1.1 연구 배경

이미지를 이용해 새로운 이미지를 생성하는 것은 딥러닝 응용분야 중 성공적인 분야 중 하나다. 이에 대한 일반적인 응용은 손상된 이미지의 일부를 복원하거나 기존의 이미지를 이용하여 새로운 이미지를 생성하는 것이다. 예를 들어 구름에 가려서 보이지 않는 항공사진을 복원하거나, 선글라스를 쓴 사람의 이미지에서 선글라스를 지우는 것이 있다. 이번 프로젝트에서 사용하고자 하는 것은 GAN 모델로, 가짜 이미지를 생성하는 Generator 와 생성된 이미지와 진짜 이미지를 구분하는 Discriminator 두 개의 네트워크를 경쟁시켜 학습을 진행한다.

생성모델이 발전함에 따라 필요한 데이터를 직접 얻거나 작성하는 것이 아닌,

딥러닝을 이용해서 생성하는 방식이 떠오르고 있다. 특히 일정한 도메인의 데이터를 다른 도메인으로 변경하는 방식이 정확도가 높기 때문에 각광받고 있다. 이러한 모델을 잘 학습시키면 이용하면 인간의 창작활동을 모방할 수 있다. 프로젝트에서는 실사 이미지를 캐리커쳐로 바꾸는 것을 목표로 한다.

실사 이미지와 캐리커쳐의 관계를 학습하는 GAN 모델을 응용한 네트워크를 구성한다. 이번 프로젝트에서는 Domain Transfer Network 라는 모델을 사용하는데, 특히 프로젝트에서 사용하는 모델은 Source Domain 의 Feature 를 추출하는 또 다른 네트워크 모델 포함하는 모델이다.

1.2 연구목표

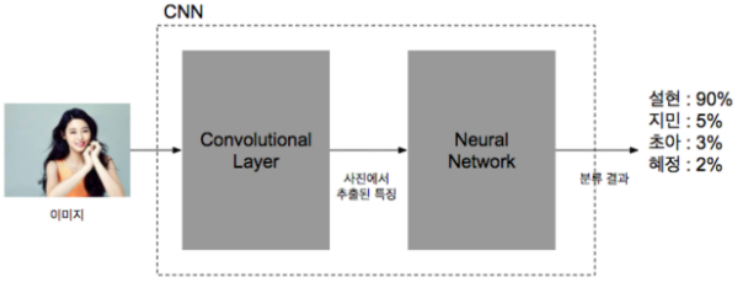
수집 가능한 학습 데이터를 모아, 사람에 대한 실사 이미지와 캐리커쳐 이미지 데이터를 모아 이 사이의 관계를 학습하는 생성모델을 만든다. 최종적으로 학습된 모델은 실사 이미지를 입력으로 받아 캐리커쳐 이미지를 생성한다.

기존의 GAN 모델을 응용한 이미지 생성은 채색을 하거나 손상된 부분을 복원하는 등, 원본 이미지를 처리하여 학습데이터를 얻을 수 있는 문제에 대한 모델이 많다. 이번 프로젝트에서 다루는 캐리커쳐 생성 모델은 원본 이미지를 처리하여 학습 데이터를 얻기 어려운 문제점이 있다. 이 부분에 대해서는 한 쪽의 Domain 의 데이터가 모자란 상황에서 동작할 수 있는 네트워크를 알아보거나, 반대로 사용 가능한 학습 데이터를 늘리는 방법 중 한 가지를 채택한다.

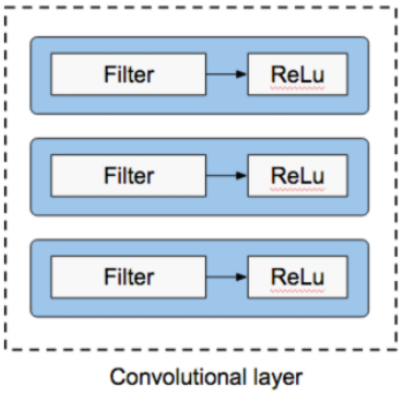
2. 기존 연구

2.1 CNN(Convolutional neural networks)

CNN 은 전통적인 뉴럴 네트워크 앞에 여러 계층의 컨볼루셔널 계층을 붙인 모양이다. 이 네트워크는 앞의 컨볼루셔널 계층을 통해서 입력받은 이미지, 텍스트에 대한 특징(Feature)를 추출하게 되고, 이렇게 추출된 특징을 기존의 뉴럴 네트워크를 이용하여 분류를 해내게 된다.

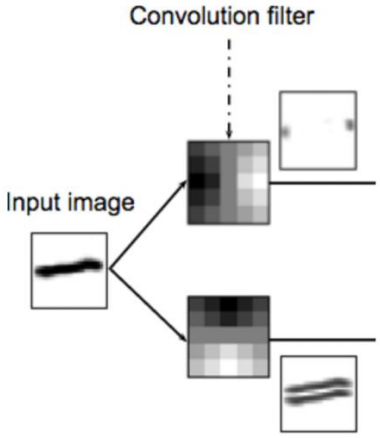


컨볼루셔널 레이어는 특징을 추출하는 역할을 수행하기 위해, 필터(Filter)와, 이 필터의 값을 비선형 값으로 바꾸어 주는 활성화 함수(Activation 함수)로 이루어 진다.

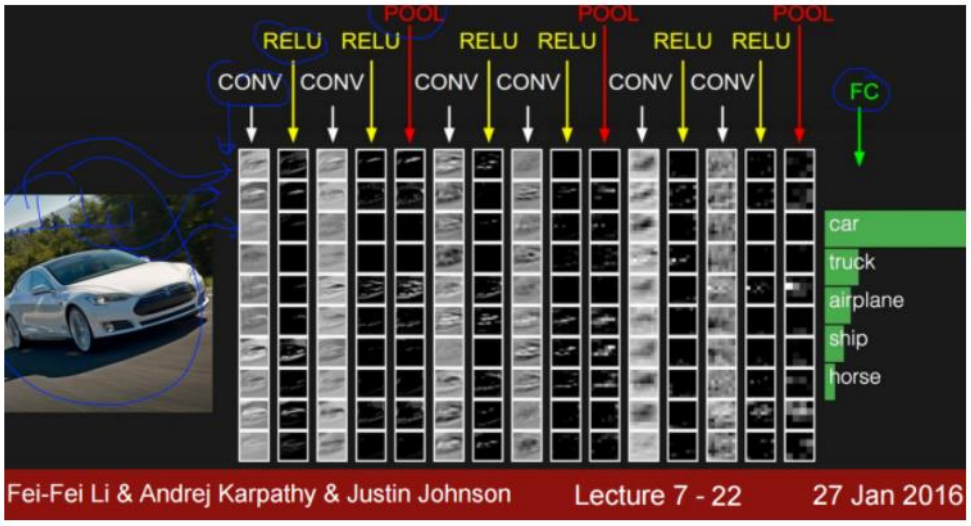


<그림 Filter와 Activation 함수로 이루어진 Convolutional 계층>

각기 다른 특징을 추출하는 필터를 조합하여 네트워크에 적용하면, 원본 데이터가 어떤 형태의 특징을 가지고 있는지 없는지를 판단해 낼 수 있다. 다음 사진은 세로선과 가로선에 대한 필터를 사용하여, Input 이미지로 부터 특징을 추출해 내는 예시이다.



그 뒤, 추출된 특징에서 가장 큰 값을 추출해 내는 맥스 풀링 기법을 사용하여, 다른 특징들을 대표하는 특징 값을 뽑아 낼 수 있다. 이 들을 조합하여 아래의 사진과 같은 CNN 을 구성할 수 있다.



2.2 Pix2Pix, GAN

Pair 로 되어있는 Data 를 모아서 CNN 을 기반으로 학습시켜서 문제에 적용시켰다. Pix2Pix 에서의 학습이란 많은 데이터를 통해서 한 이미지의 representation 을 다른 이미지로 변환하는 방식을 배우도록 하는 것을 말한다.

아래의 사진은 위의 데이터셋을 사용하여 학습이 진행되는 과정을 나타낸다. G 는

입력 데이터(예시 데이터 셋에서는 에지-edge)를 사용하여, 거짓 이미지를 생성하는 Generator, D 는 거짓 이미지인지, 실제 이미지인지를 판별해 내는 Discriminator 를 의미한다.

G 는 자신이 생성해 낸 이미지가 D 를 속여 실제 이미지로 판별되게끔 하는 방향으로, D 는 거짓 이미지와 실제 이미지를 높은 정확도로 판별하게끔 하는 방향으로 학습이 진행된다. 이러한 경쟁 과정 속에서, G 는 좀 더 realistic 한 거짓 이미지를 생성하게 된다.

학습의 방향을 수식으로 나타낸 식은 아래와 같다.

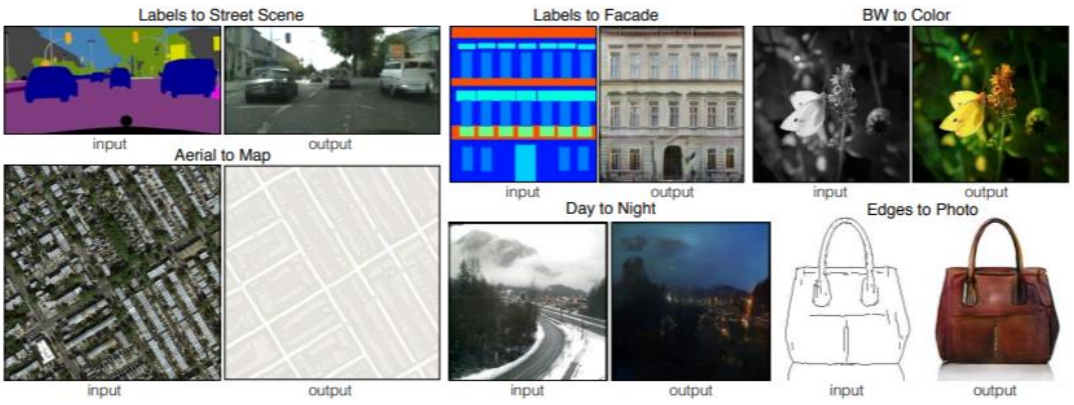
$$G^* = \arg\min_G \max_D L_{cGAN}(G,D) + \lambda L_{L1}(G) \quad (1)$$

$$L_{cGAN}(G,D) = \mathbb{E}_{y\sim p_{data}(y)}[\log D(y)] + \mathbb{E}_{x\sim p_{data}(x),z\sim p_z(z)}[\log(1 - D(G(x,z)))] \quad (2)$$

$$L_{L1}(G) = \mathbb{E}_{x,y\sim p_{data}(x,y),z\sim p_z(z)}[\|y - G(x,z)\|_1] \quad (3)$$

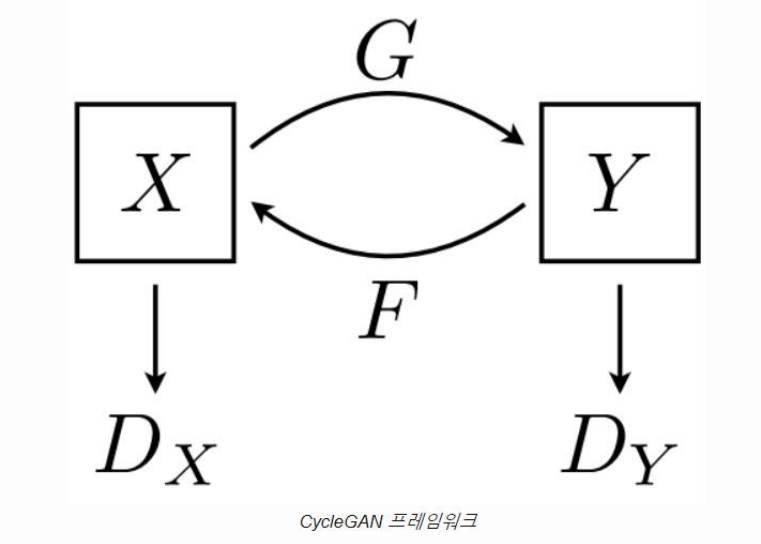
이와 같이 학습된 모델을 사용하여, 흑백 사진에 색을 입히는 Colorization 이나 스케치로 부터 전체 이미지(초상화 등)을 생성해 내는 것과 같은 작업을 할 수 있다.

아래 사진은 그 예시이다.



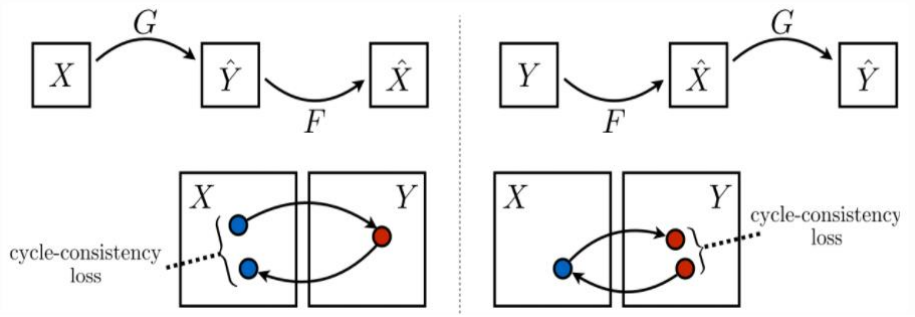
2.3 Cycle GAN

'판별자를 속이도록 학습해 GAN 손실 함수로 더 진짜 같이 만들자'는 기본철학은 Pix2Pix 모델과 CycleGAN 모두 동일하다. 그러나 CycleGAN 에서는 Pix2Pix 와 같은 모델과는 달리 한 개의 생성자가 아닌 두 개의 생성자 G, F 를 상정한다.
CycleGAN 의 구조도는 아래와 같다.



CycleGAN 프레임워크

CycleGAN 이 두 가지 생성자 G,F 를 가지고 있는 이유는 순환 일관성(Cycle Consistency)때문이다. 이를 직관적으로 설명하면, G 를 통해 Y 로 보내진 X 데이터 G(Y)가 있을 때, 이를 다시 F 를 통해 보내면 X 로 되돌아와야 한다는 것이다. 아래 사진은 이 순환 일관성이 유지되지 못한 예시를 나타낸다.



이러한 순환 일관성을 이용한 순환 손실 함수(Cycle Loss)는 아래와 같다.

$$L_{cyc}(G,F) = \mathbb{E}_{x\sim p_{data}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y\sim p_y(x)}[\|G(F(y)) - y\|_1] \qquad (4)$$

2.4 기존 연구의 문제점 <-임호준

2.4.1 정확도

다음 사진은 Photo-to-Caricature Translation on Faces in the Wild 라는 논문에서 사람의 얼굴을 입력 이미지로 받아, GAN 을 이용해 캐리커처화 한 사진이다. 주제는 우리가 캡스톤 디자인 2 에서 진행하고자 하는 주제와 같으나 Output 이미지를 보면, 대체로 사람의 형상을 알아볼 수 없는 이미지들이 많다.

2.4.3 GAN 모델 학습의 어려움

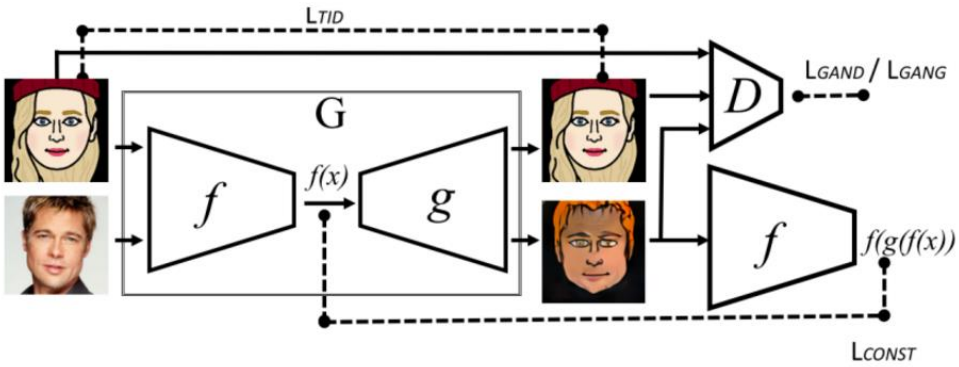
배치 사이즈, learning rate 와 같은 하이퍼 파라미터를 조정하거나, 모델에 적절한 이미지의 사이즈를 찾아내는 등의 어려움이 있으나, 더 큰 문제는 적절한 질의 데이터와 충분한 양의 데이터를 구해야 학습이 잘 진행된 GAN 모델을 생성할 수 있다. 그러나 공공 데이터로 publish 된 충분한 양의 캐리커처 이미지를 구하는데 어려움이 있었다. 또한, 눈 코 입의 위치로 레이블링 된 사람의 이미지 데이터셋은 많았으나, 프로젝트를 진행함에 있어 필요했던 사람 이름으로 레이블링 된 사람의 이미지 데이터셋이 많지 않아 이 역시 데이터 수집에서 어려움이 있었다.

마지막으로 네트워크를 구성함에 있어, CNN 레이어의 필터의 숫자, 정규화, 활성화 함수의 선정등이 학습에 큰 영향을 미쳐 이를 선정하는 것 역시 어려움이 있었다.

3. 프로젝트

3.1 기존 연구와 차이점 및 해결방안

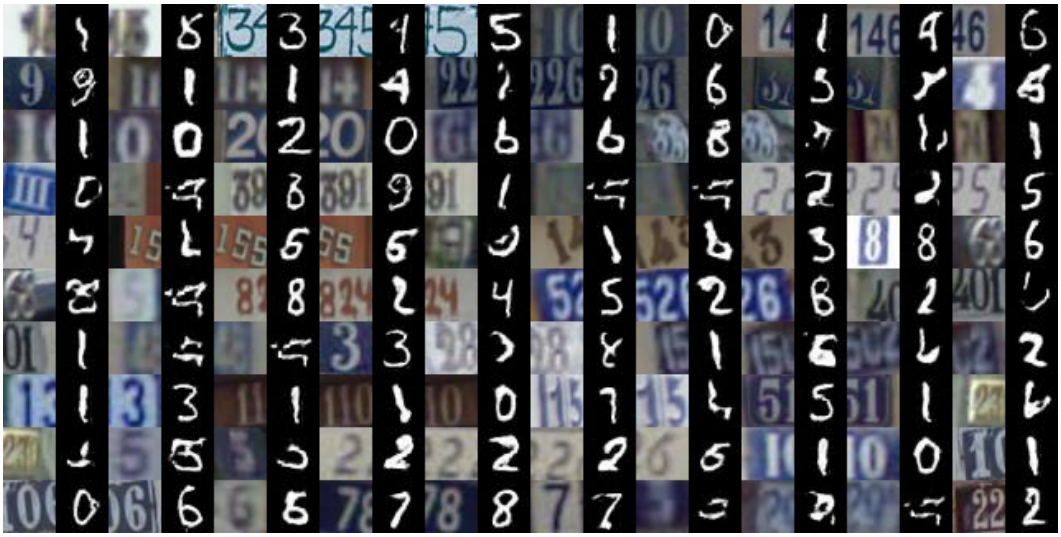
기존 방식의 문제점은 크게 성능과 데이터 수집 두 가지고, 이 두 가지 문제를 해결하는것이 이번 프로젝트의 진행의 핵심이다. 먼저 성능 즉, 정확도를 높이는 방법이다.



[그림] Domain Transfer Network

이 프로젝트에서는 Domain Transfer Network 모델을 사용해서 실사 이미지를 이모지로 변경한다. 기존의 방식은 두 도메인과 간단한 GAN 을 이용해서 도메인 사이의 관계를 학습하지만, 이 프로젝트에서 사용하는 방식은 Source Domain 에서 Feature 를 추출하는 딥러닝 모델을 학습한 뒤, 이 모델을 이용해서 추출한 Feature 를 이용해 Domain Transfer 가 일어나는 방식이다.

위 그림에서 나타나는 인코더 f 가 Feature 를 추출하는 네트워크 모델이 되고, g 와 d 를 이용해서 Feature 로부터 이모지를 생성하는 네트워크를 구성한다. 이 방식은 인코더와 디코더의 응용으로, Target Domain 이모지에 f 와 g 를 적용하면 원본에 가까운 이모지를, Source Domain 실사 이미지에 f 와 g 를 적용하면 실사 이미지의 Feature 를 갖는 이모지를 생성하는 방식이다.



이에 관한 논문으로 (참고자료)가 있지만, svhn(Street View House Numbers)과 mnist 데이터 셋을 사용하는 단순한 예제만을 공개하고 있고, 실제 이모지를 생성하는 모델은 이론적인 부분만을 언급하고 구체적인 결과와 데이터 셋을 공개하고 있지 않다. 학습에 사용할 데이터 셋을 직접 준비하고, 실제 이모지를 생성하는 Domain Transfer Model 을 작성한다.

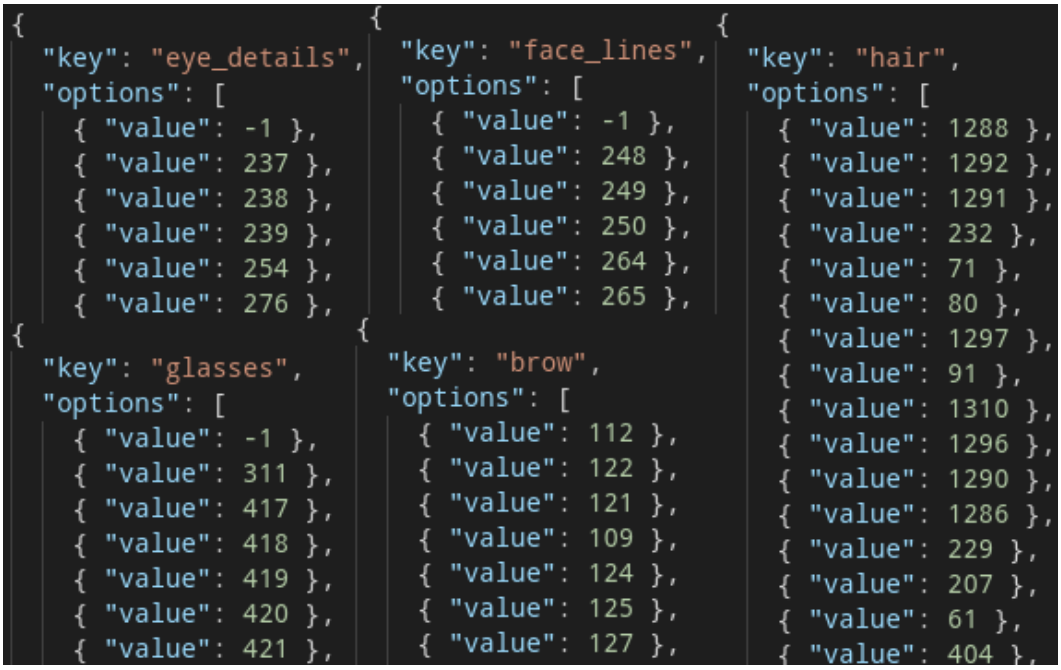
3.2 **프로젝트 내용**

이 프로젝트는 크게 데이터 처리와 딥 러닝 모델 개발 두 가지 파트로 나눠진다. 데이터 파트의 구현은 다시 이미지 수집과 전처리 과정으로 나눠진다. 상대적으로 실사 이미지에 대한 데이터는 수집하기 쉽지만, 캐리커쳐의 경우 학습에 사용할 적절한 데이터를 구하기 어렵다. 이러한 문제점을 해결하기 위해 Bitmoji 라는 캐리커쳐 생성 서비스를 이용한다.

[그림] Bitmoji

Bitmoji 는 파라미터를 조합해서 캐리커쳐를 만드는 앱이다. 이 앱을 리버싱하여 캐리커쳐를 받아오는 API 를 추출하고, 추출한 API 를 이용해 크롤러를 만들어 학습에 사용할 이모지 데이터를 수집한다.

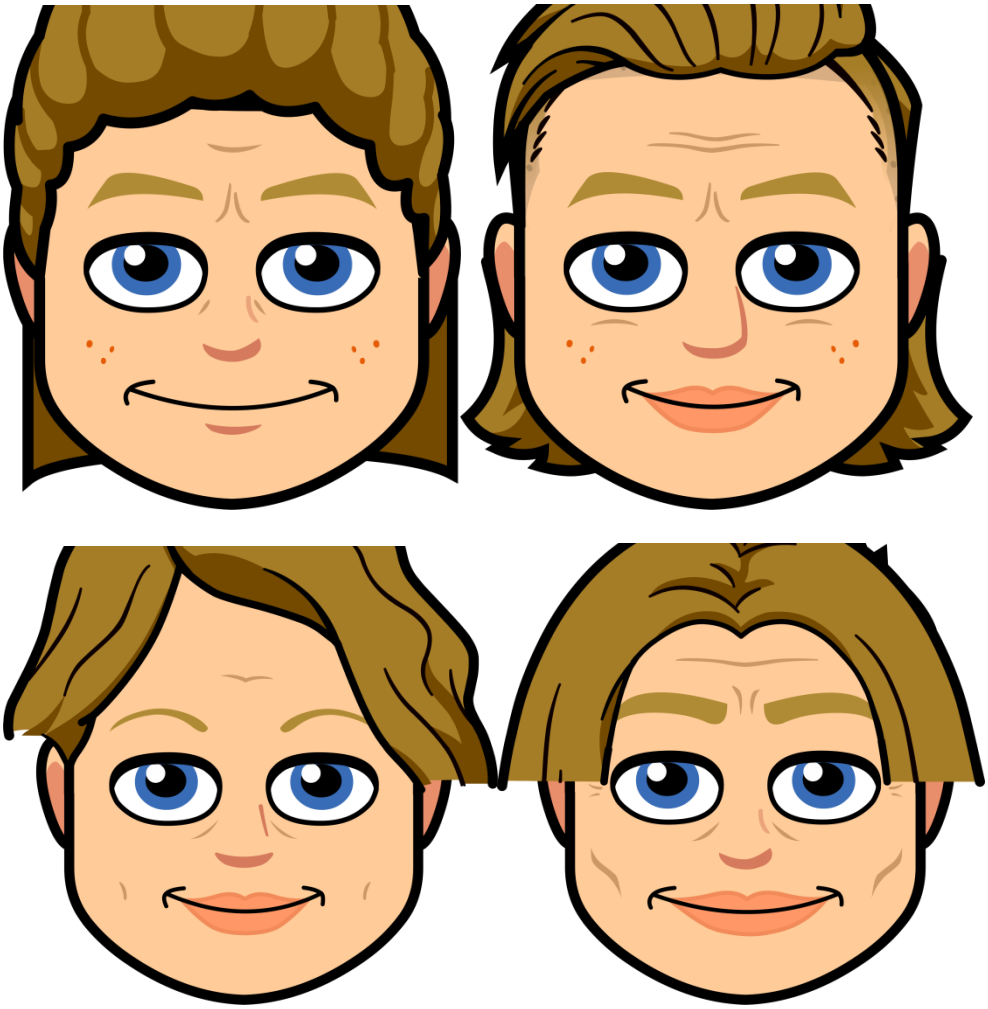
첫 번째 시도는 구 버전의 Bitmoji 를 이용한 것이다. 구 버전의 Bitmoji 는 웹 서비스 형태로도 제공을 했기 때문에 복잡한 리버싱 과정이 없어도 시행착오를 통해 API 를 추출할 수 있었다. 추출한 API 는 이모지의 표정과 자세를 결정하는 Template 부분과, 얼굴 모양과 옷 그리고 악세사리 등 이모지를 꾸미는 파라미터로 이루어진다.



[그림] 추출된 파라미터

API 를 추출하는것과 별개로, API 에 사용하는 파라미터를 추출하는 일이 잘 진전되지 않았다. 때문에 직접 코드를 보고 파라미터를 찾는 것 보다. 호출이 성공하는 API 를 이용해서 하나의 파라미터만 바꿔보는 것으로, 호출이 성공하는 파라미터만을 직접 추출했다. 이러한 추출된 파라미터는 10 가지가 있고, 생성할 수 있는 모든 이모지는 약 3826521578880(약 4 조) 가지로, 갯수만을 봤을때는 전혀 모자람이 없었다.

추출한 API 를 이용해 NodeJS 에서 동작하는 크롤러를 만들었다. 크롤링 자체는 간단하기 때문에 기본으로 내장된 모듈만을 사용해서 만들 수 있었다. 추출된 이미지는 딥러닝 모델에서 사용할 수 있도록 Python 의 cv2 와 numpy 라이브러리를 통해 (152, 152, 3) 행렬 형태로 저장했다. 수집된 데이터는 다음과 같다.



구 버전의 Bitmoji 는 눈의 모양과 얼굴의 자세한 모양을 파라미터로 제공하고 있지 않기 때문에 생성된 이미지가 비슷하다는 문제점이 있었다. 이 이상 데이터를 수집하는 것은 무의미하다 판단했고, 구 버전 Bitmoji 를 버리고, 다른 방법으로 이모지를 얻는 방법에 대해 조사했다.

실제 Bitmoji 는 훨씬 자세하고 다양한 이모지를 제공한다. 위의 과정에서 얻은 것은 구 버전의 Bitmoji 서비스에서 제공하는 이미지로, 프로토타입의 의미가 강하다. 신 버전의 이모지를 크롤링하기 위해서는 Bitmoji 앱을 리버싱하고 API 와 파라미터를 추출하는 과정이 필요했다.

우선은 리버싱이 필요없는 방식으로 접근해보았다. 앱에서 이모지를 만들고, 이모지에 대한 URI 를 추출하여 파라미터를 조사하는 방식이다. 구 버전 API 를 추출하는 과정에서 알아낸 사실로 모든 파라미터는 GET 방식으로 전달됐기 때문에 URI 만 알아내면 파라미터를 추출할 수 있을 것이라 생각했다.

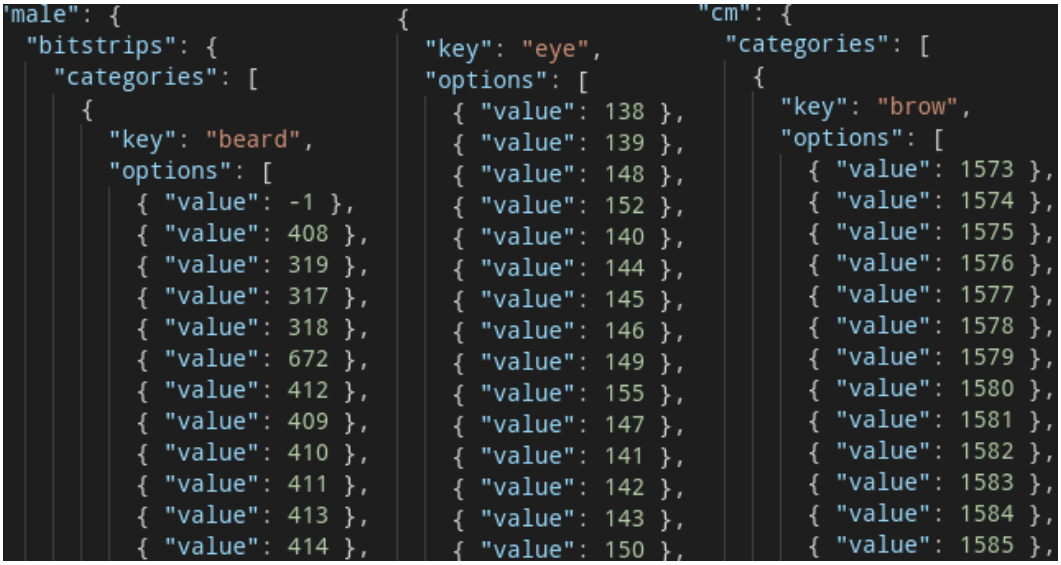
하지만 이 접근 방식은 사용할 수 없었다. 신 버전의 Bitmoji 는 생성된 이미지의 URI 를

해싱한 뒤 사용자에게 노출시키기 때문에 파라미터를 추출할 수 없었고, 크롤링에 사용할 API 를 얻는 것도 불가능했다. 실제로 앱을 리버싱해서 이미지 생성을 요청하는 부분을 찾아내야만 했다.

이를 위해 dex2jar 를 통해 Bitmoji 앱을 디컴파일하고, jd-GUI 를 이용해 API 의 호출부를 찾았다. 시행착오 끝에 아바타 생성 클래스의 메서드의 부분에서 preview 를 보여주는 코드를 찾았고, 이 코드를 이용해서 API 의 기본이 되는 틀을 만들 수 있었다.

[그림] preview

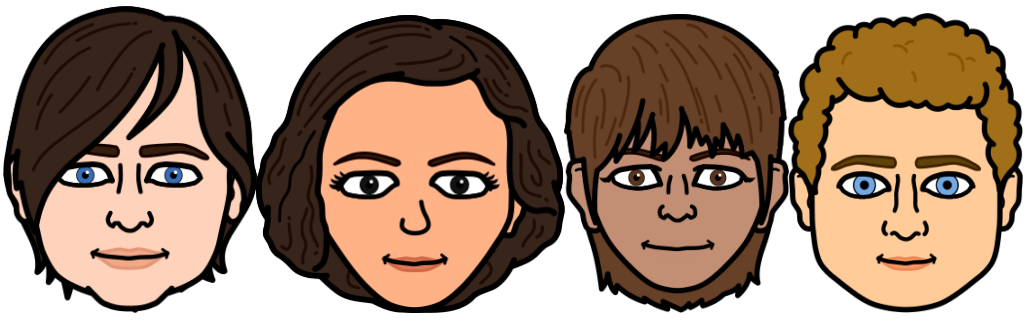
구 버전의 Bitmoji 보다 훨씬 자연스러운 얼굴이다. 신 버전의 Bitmoji 는 남성과 여성을 따로 나누어 이모지를 생성하고, 눈과 얼굴형태를 포함한 좀 더 많은 파라미터를 제공하기 때문에 추출에 성공하기만 한다면 훨씬 나은 데이터 셋을 얻을 수 있었다. 하지만 이 파라미터를 추출하는 과정이 잘 진전되지 않았다. 앱의 내부에 있는 것이 아니라, 서버에서 로드하는 방식이었기 때문이다.

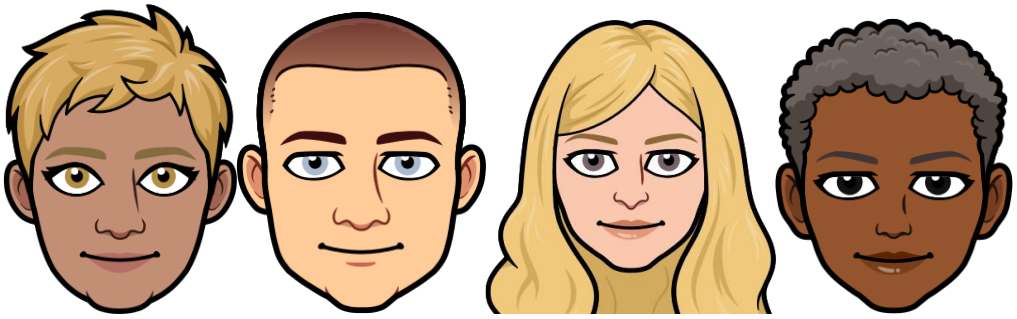


[그림] 이모지 파라미터

다행히 리버싱을 하는 과정에서 이를 파악하고, 서버에 요청하여 파라미터를 추출할 수 있었다. 구 버전의 API 와는 다르게 bitmoji, bitstrips, cm 세 곳에서 데이터를 가져오는데, bitmoji 는 구 버전에서 생성된 이미지를, bitstrips 와 cm 은 좀 더 나은 이미지를 제공했다. 세 곳 모두 여러가지 파라미터를 조합해서 이모지를 생성하는 빙식이었기 때문에, 사실상 무한한 데이터를 추출할 수 있었다.

추출한 API 와 파라미터를 이용해 크롤러를 만들었다. 32GB 의 데이터를 수집하여 전처리를 하고 (100, 152, 152, 3) 텐서(100 개의 배치)로 저장했다. 실제 학습에서 모든 데이터를 사용하지는 않았지만 유의미한 결과였다.





[그림] Bitmoji 에서 추출한 이모지

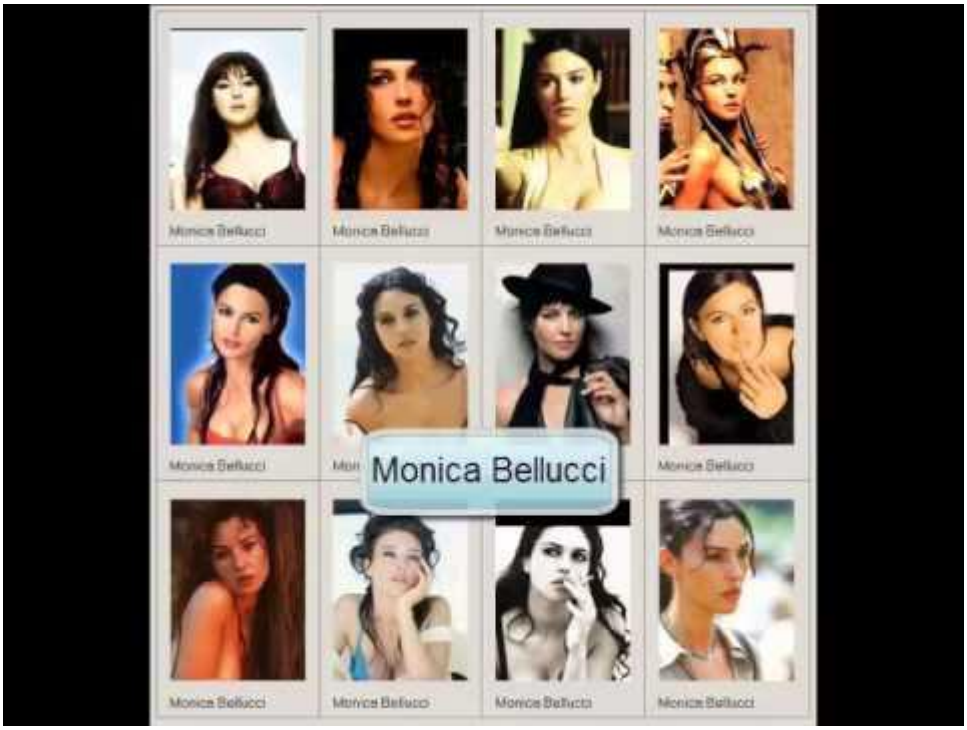
다음은 실사 이미지에 대한 부분이다. 이 프로젝트에서 사용하는 딥러닝 모델은 피쳐를 추출하는 네트워크와 이모지를 생성하는 네트워크 두 가지 모델을 동시에 사용하는데, 실사 이미지는 Feature 를 추출하는 인코더 파트의 네트워크 모델의 학습에 사용한다.

인코더는 실사 이미지로 부터 사람을 이름별로 분류할 수 있는지 학습한 뒤, 분류를 위한 Softmax 레이어 직전의 fully connected layer 를 Feature 추출기로 사용한다. 여기서는 인코더를 학습시키기 위해 실사 이미지를 이름으로 분류하는 지도 학습을 진행하기로 했다.

우선 CelebA 데이터 셋을 살펴보았다. 이 데이터는 방대하고 많은 파라미터를 제공했지만, 정작 필요한 이름에 대해 라벨링이 된 것이 없기 때문에 사용할 수 없었다.

다음으로 LFW 데이터셋을 살펴보았다. 이름으로 라벨링이 돼 있긴 했지만, 다양한 사람들을 조금씩 모아둔 데이터기 때문에 프로젝트에서 사용하는 모델에 적합하지 않았다.

다음으로 찾아본 데이터는 MSRA-CFW 데이터셋이다. 이 데이터 셋은 한 명의 사람에 대해서도 많은 갯수의 이미지를 제공하기 때문에 프로젝트에 적합했다.



하지만 이 데이터는 대부분 전신사진이기 때문에 전처리 과정이 필요했다.

얼굴을 추출하기 위해 opencv 의 haarcascade_frontalface_default.xml 와 CascadeClassifier 를 사용했다. 한 이미지에서 여러개의 얼굴이 탐지되거나 아예 얼굴 탐지에 실패한 경우를 제외하고 약 12 만장의 얼굴 데이터를 추출할 수 있었다.

이모지 데이터와 마찬가지로 (100, 152, 152, 3) 행렬과 추가적으로 이름에 대한 라벨 벡터의 형태로 데이터를 저장했다.

3.2.1 Img2Caricuture Network 구성

3.2.1.1 Domain Transfer Network 실사 이미지를 캐리커쳐로 생성하기 위한 Domain Transfer 의 기본 네트워크 구조는 [5]의 Domain Transfer Network 를 참고하여 구현하였다. 아래의 그림은 네트워크의 구조도와 학습에 사용된 Loss 함수를 간략하게 나타낸다.

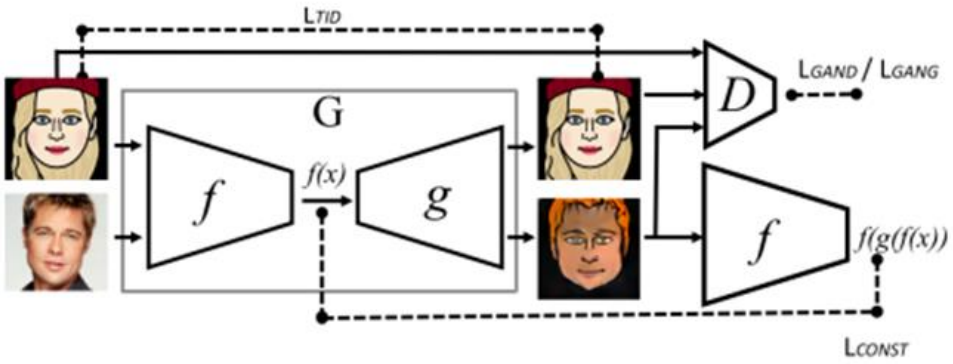


Figure 1 Domain Transfer Network 의 구조

네트워크에서 G(생성자)는 인코딩을 담당하는 특징 추출 함수 f 와 추출된 특징으로부터 Target Domain 에 해당하는 이미지를 생성하는 함수 g 로 구성되어 있다. 구현된 f 와 g 의 상세 구조는 Figure 2 와 같다.

==========Figure==========

함수 f 는 사전에 pre-train 을 하며, 전체 네트워크의 학습이 진행될 때는 Discriminator(판별자)와 함수 g 의 가중치들만이 갱신된다.

3.1.1.1 Loss function

기존의 GAN 에서의 식별자는 입력 이미지 실제 이미

지인지 거짓 이미지인지 2 진 분류하는 과정을 통해 학습해 나간다. 그러나 Domain Transfer Network 에서의 식별자는 입력 이미지가 Target-Domain 인지, Target-Domain 이미지가 G 를 거쳐 생성된 이미지인지, Source-Domain 이미지가 G 를 거쳐 생성된 이미지인지에 대한 3 진 분류를 수행해가며 학습을 진행한다.

생성자를 학습시키기 위한 Loss function 은 세 가지로 나뉘어져 있다. L_{GANG} , L_{CONST} 그리고 L_{TID} 이다.

L_{GANG} 는 G 를 거쳐 생성된 이미지가 얼마나 판별자를 속였는지에 대한 피드백으로 학습이 진행되는, 보통의 GAN 과 같은 형태의 오차 함수이다.

L_{CONST} 는 Source-Domain 이미지에서 추출된 특징과, 생성자를 거쳐 생성된 Target-Domain 이미지의 특징이 서로 일치하는지에 대한 함수로, 이를 통해 생성자 G 내의 함수 g 를 원본의 특징이 유지된 채 캐리커쳐 이미지를 생성하도록 학습시킨다.

L_{TID} 는 Target-Domain 이미지가 G 에 입력 이미지로 들어왔을 때에는 생성자를 거치더라도, 원본의 이미지의 특징 및 형태가 생성된 이미지에도 유지되어야 함으로, 이를 보정하기 위한 오차 함수이다.

최종적으로 생성자 G 의 학습을 진행시키기 위한 오차 함수는 위 세 가지 함수를 합한 형태의 함수가 된다.

판별자(Discriminator)를 학습시키기 위한 loss function 역시 세 가지로 구성되어 있다. 판별자의 경우 softmax 함수를 통해, 입력으로 들어온 이미지가 원본 source 이미지인지, 원본 target 이미지인지, 생성자를 거친 target 이미지인지에 대한 삼진 분류를 수행하고, 이를 통해 cross entropy 오차를 계산하여 합삲ㅏㄴ 값이 판별자의 loss 값이 된다.

최종적인 오차함수의 수식은 아래와 같다.

$$L_{\mathrm{D}} = -\mathbb{E}_{x\in\mathbf{s}} \log D_1(g(f(x))) - \mathbb{E}_{x\in\mathbf{t}} \log D_2(g(f(x))) - \mathbb{E}_{x\in\mathbf{t}} \log D_3(x)$$

$$L_{\mathrm{GANG}} = -\mathbb{E}_{x\in\mathbf{s}} \log D_3(g(f(x))) - \mathbb{E}_{x\in\mathbf{t}} \log D_3(g(f(x)))$$

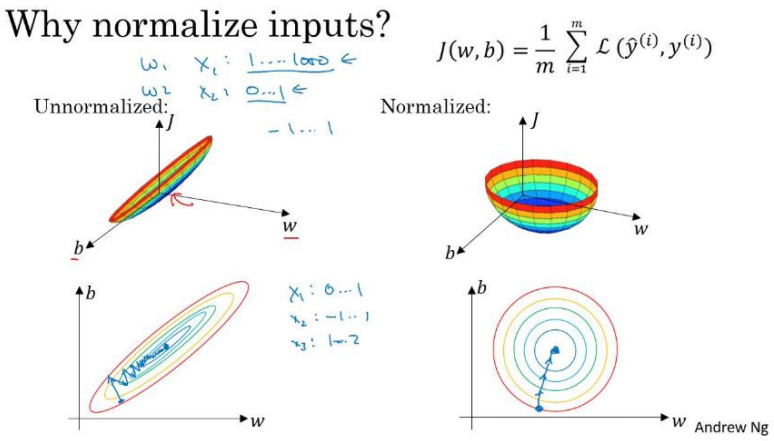
$$L_{\mathrm{CONST}} = \sum_{x\in\mathbf{s}} d(f(x), f(g(f(x))))$$

$$L_{\mathrm{TID}} = \sum_{x\in\mathbf{t}} d_2(x, G(x))$$

3.1.1.2 정규화

학습에 사용된 데이터는 이미지 데이터로, rgb 3 가지 채널에 0~255 까지의 값이 저장된다. 이를 정규화 하지 않은 채 학습을 진행하게 되면, 학습을 진행함에 있어, weight 가 학습되는데 지장을 입히고, 지역 최적에 상태에 빠지게 될 가능성이 있으며, 입력 데이터간의 영향력이 편이하게 차이날 수 있기 때문에, 정규화를 수행하였다. 정규화의 결과는 0~255 사이의 값이 -

1~1 사이의 값이 되도록 (x/127.5 -1, x 는 rgb 의 채널값)의 수식을 사용하여 정규화하였다.



3.1.1.3 Batch normalization

DCGAN 에서 자주 쓰이는 테크닉으로, Discriminator 의 입력층과 Generator 의 출력층을 제외한 모든 레이어의 다음에 사용되었다. 이는 레이어에 들어온 인풋을 평균이 0 이고 분산이 1 인 값으로 정규화를 시키는 것인데, 앞의 정규화와는 다르게 네트워크에 입력하는 값을 정규화 하는 것이 아니라, 네트워크 내에서 각 레이어에 들어가는 인풋을 정규화 시키는 개념이다.[6]

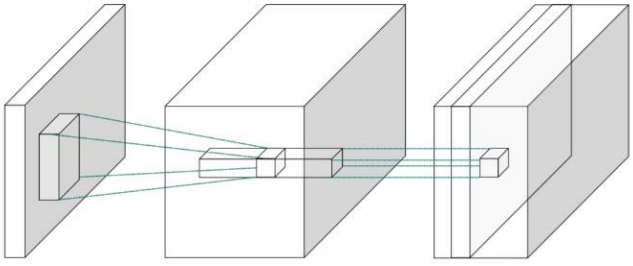
3.1.1.4 Drop out[1]

Image-to-Image 논문의 저자에 따르면, GAN 을 학습시킬 때 있어, 생성자에게 Drop out(50%)를 주는 것이 학습 성능을 향상 시킨다고 나와있다. 이는, GAN 에게 임의의 노이즈를 주는 효과가 있기 때문이라는 것이 입증되어 있어, 본 프로젝트에서도 DTN Generator 에 Drop out 레이어를 추가하였다.

3.1.1.5 1x1 Convolution[7]

1 layer fully connected neural network 라고도 불리는 1x1 Convolution network 를 Generator 와 Discriminator 에 적용하였다. 1x1 Convolution 은 여러개의 featuremap 으로부터 비슷한 성질을 묶어내어, featuremap 의 차원을 줄여 연산량을 줄이는 효과를 얻기위해 사용하였다. 또한 Generator 와 같이 deconvolution 이 필요한 네트워크에서는 마찬가지로 featuremap 의 차원을 늘려

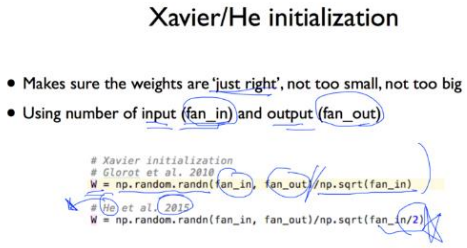
네트워크의 깊이가 깊어져 학습이 수월해지도록 하였다.



1x1 conv layer

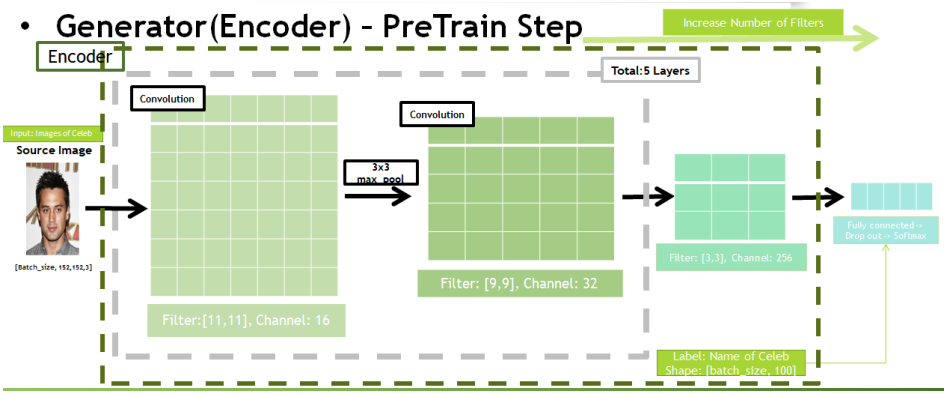
3.1.1.6 Xavier initialization

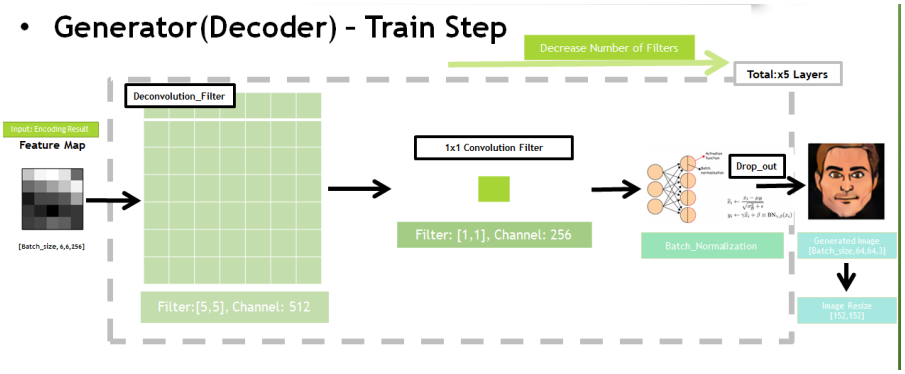
딥러닝 모델에 있어, 모델의 가중치들의 초기 값을 선정하는 것은 학습의 성능에 영향을 미치기 때문에, 너무나도 중요하다. 이를 위해 2015 년에 발표되어 기존의 RBM 가중치 초기화 방식보다 더 높은 성능을 입증한 Xavier initialization 을 사용하였다.

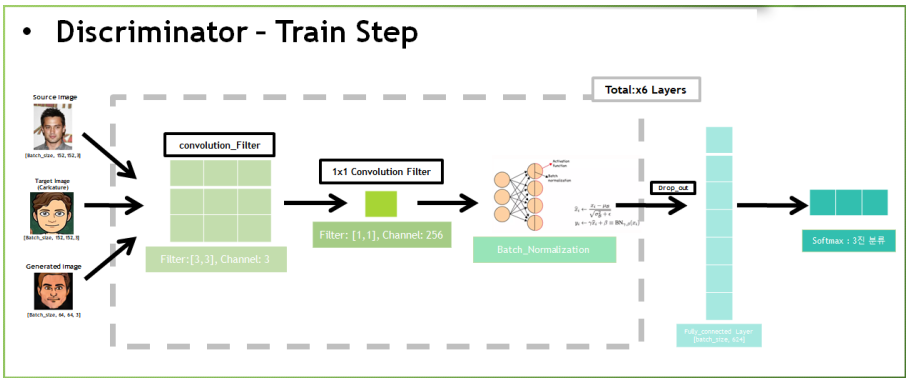


초기화 하는 방식은 (입력값과 출력값 사이의 임의의 수)/(출력/입력값 제곱)이다.

3.1.1.7 실제 구현된 네트워크 구조







4. 실험 및 평가

Target domain 의 데이터는 Bitmoji 에서 얻은 이모지 데이터로 (100, 152, 152, 3)의 행렬을 이용한다. 이는 각각(배치 사이즈, 가로 폭, 세로 폭, RGB)를 표현하는 데이터다. Bitmoj 어플리케이션에서 이미지를 요청하는 부분을 추출하고, 이를 이용해서 크롤러를 만든다. 크롤러가 모은 약 32GB 의 png 데이터를 위의 행렬의 형태로 변환하여 저장한다. 실제 딥러닝에 사용한 데이터는 10 만장으로 약 8GB 다.

i. Source domain 의 데이터는 MSRA-CFW 데이터셋을 사용한다. 원본 데이터는 얼굴 부분에 대한 crop 이 전혀 돼있지 않기 때문에 이를 추출하는 과정에서 일부 데이터가 유실(얼굴이 여러개 인식되거나 인식에 실패한 경우)되고, 유실된 후에는 한 명당 평균 70 개의 이미지가 남는다. 이 이미지는 이모지와 마찬가지로 (100, 152, 152, 3)의 행렬로 저장되고, 라벨은 배치로만 묶어서 길이가 100 인 배열로 저장된다. One Hot 에 대한 표현으로 딥 러닝 모델에서

변환하기 때문에 사용한 라벨의 수를 따로 저장한다. 실사 이미지는 약 9GB 를 사용할 수 있었지만, 학습 시간이 오래걸려 3.5GB 의 데이터만을 사용했다.

ii. 실험환경

DeskTop Setting

CPU: i5-7500

RAM: 32GB

GPU: GTX-1080TI

OS : Window 10

Learning Parameters

Batch Size : 100

Number of Celeb Labers : 100

Size of Celeb Data : 5600

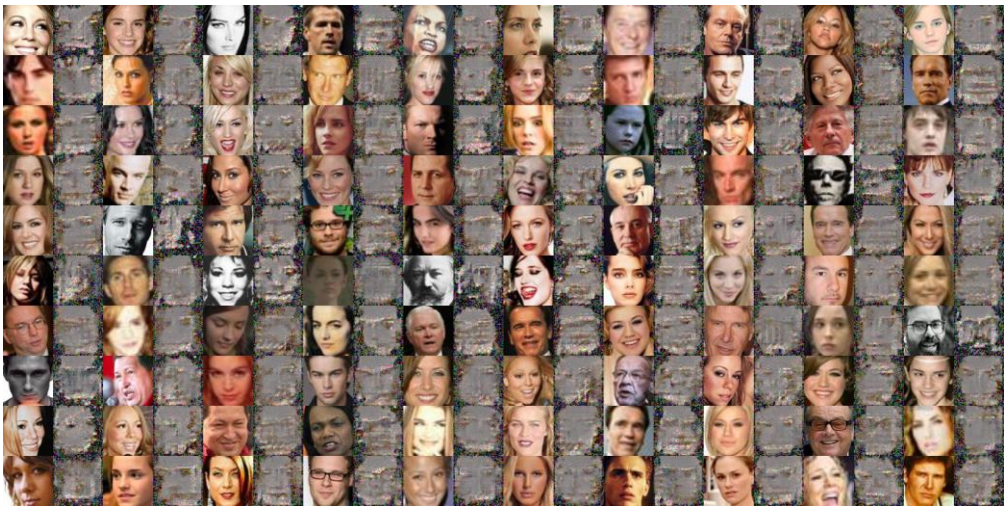
Size of Caricature Data : 100000

Pretrain iter : 5000

Train iter : 50000

Learning Rate : 0.015

iii. 결과



학습 결과, 얼굴형태를 유지한 이미지를 생성하는 것에는 성공하였으나, 눈 코 입등의 위치를 정확히 잡거나, 캐리커쳐 형태의 이미지를 생성하는 것을 동시에 만족하는 이미지를 생성하는 것에는 실패하였다.

5. 결론

프로젝트를 통해 실사 이미지를 캐리커쳐로 변경하는 딥 러닝 모델을 작성한다. 이 과정에서 기존의 Cycle GAN 과 같은 접근법을 사용하기 보다는, Feature 를 추출하는

또 다른 네트워크와 인코더, 디코더를 이용한 Domain Transfer Network 구조를 이용한다. Domain Transfer Network 를 이용하면 파라미터를 조합해서 원본 이미지에 가까운 이미지를 생성하는 방식이 아닌, 임의의 파라미터 조합을 이용해 학습 데이터를 추출하고, 추출된 데이터를 이용해 이미지를 생성하는 네트워크를 만들 수 있다.

이번 프로젝트에서 작성한 네트워크 모델은 Feature 을 추출하는 역할만을 하는 네트워크를 미리 학습한 뒤, 학습된 네트워크를 이용해 또 다른 네트워크를 학습시키는 구조다. 이번 프로젝트에서는 이 과정을 분리해서 작성했지만, 동시에 학습할 수 있다면 더 나은 결과를 얻을 수 있을것이라 생각한다.

참고 문헌

[1] Phillip Isola, Jun _Yan Zhu, Tinghui Zhou, Alexei A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks", arXiv:1611.07004v2: 22 Nov 2017

[2] Marleana L.Itz, Stefan R.schweinberger, Jurgen M.Kaufmann. "Caricature generalization benefits for faces learned with enhanced idiosyncratic shape or texture", Cogn Affect Behav Neurosci (2017) 17:138-197

[3]Wengling Chen, James Hays. "SketchyGAN:Towards Diverse and Realistic Sketch to Image Synthesis," arXiv:1611.07004v2: 9 Jan 2018

[4] Ziqian Zheng, Haiyong Zheng, Zhibin Yu, Zhaorui Gu, Bing Zheng. "Photo-to-Caricature Translation on Faces in the Wild," arXiv:1611.07004v2: 29 Nov 2017

[5]Yaniv Taigman, Adam Polyak & Lior Wolf, "Unsupervised Cross-Domain Image Generation," Under review as a conference paper at ICLR 2017

[6] Sergey Ioffe, Christian Szegedy, "Batch NormalizationL Accelerating Deep Network Training by Reducing Internal Covariate Shift"

[7] Christian Szegedy, Sergey Ioffe, Vincent Vincent Vanhoucke, Alexander A.Alemi," Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,"