

최종발표

Image to Image translation
(Face to animation)

조민기

목차

1. Data Processing
2. Supervised Learning
3. Unsupervised Learning
4. Tag Estimation

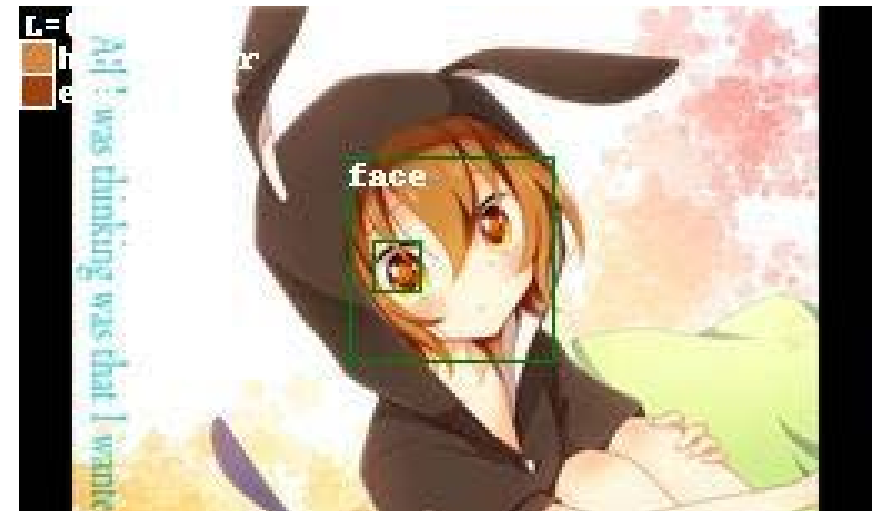
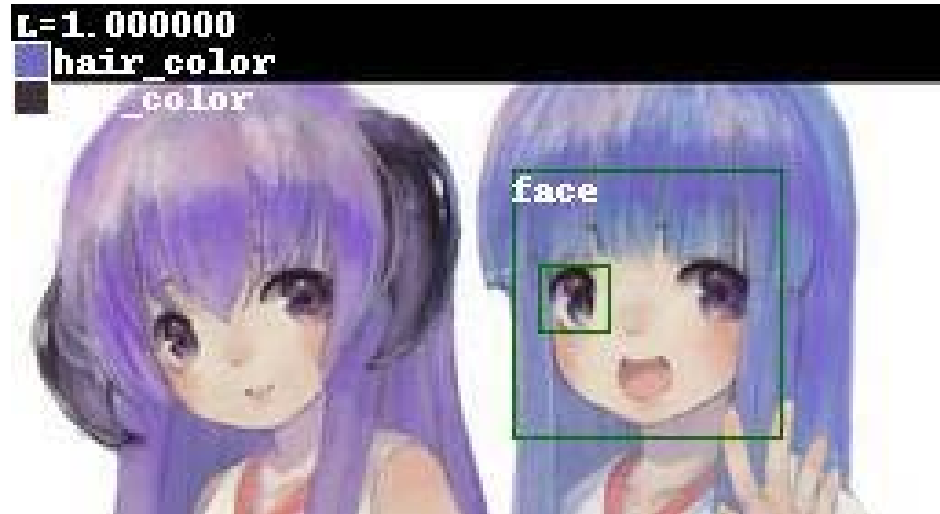
Data Processing

1. **Cartoon face detector**
2. **Anime face with attributes**
 1. Face + keypoint + attributes for pix2pix
 2. Face + keypoint for cycle GAN

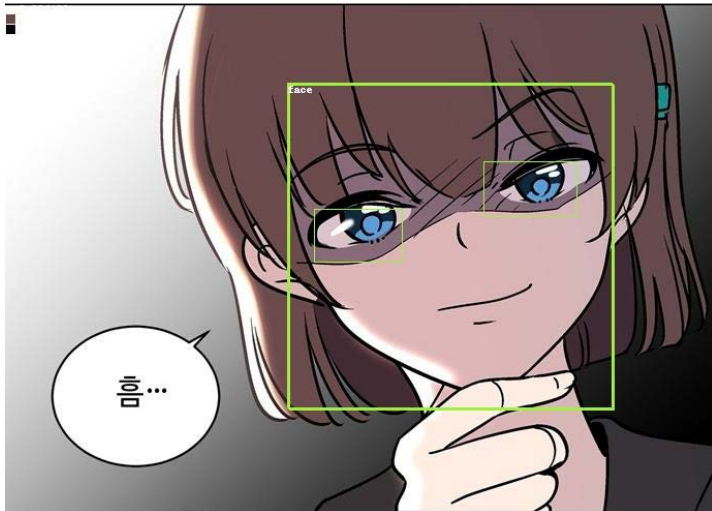
1. Cartoon face detector

Face Detector

- Cartoon, Animation에서 캐릭터의 **얼굴과 이목구비를 Detect**하고 **피부, 눈, 머리의 색깔**을 Estimate 하는 모델이다.
- Danbooru data 로 학습되었으며, Likelihood가 **가장 높은 1 지점**을 포착해낸다. (Selective search를 쓰지 않았을까 예상됨)
 - 즉, 여러 얼굴을 포착할 수 없다.




1. Cartoon face detector





1. Cartoon face detector


Contribution


- 대량의 일정 규격으로 정렬된 웹툰, 애니메이션 얼굴 데이터를 정제할 수 있음.
 - 두 눈이 다 나오며 잘리지 않은 얼굴만 추출해내므로, Face translation 등의 프로젝트 등등을 위해 사용할 수 있음.
- Danbooru 데이터의 경우 눈 코 입 디텍션 및 피부 눈 머리색 측정이 상당히 정확한 편.
 - Sentimental analysis 혹은 Facial expression 분석 등등의 프로젝트에 활용할 수 있음.

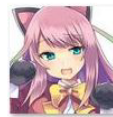
 마음의소리_캐릭터얼굴_5474개.zip

 외지주_남자얼굴_3784개.zip

 외지주_여자얼굴_1393개.zip

 웹툰_캐릭터_10classes.zip

 조석_얼굴_1176개.zip



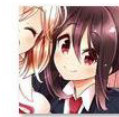
danbooru_24965
49_590ce7b9e09
8fdc96dd6dbcc2
7ad9674.jpg



danbooru_24968
24_253b55f17a4
2d816cc7bd8dc
7152fe31.jpg



danbooru_24972
08_15e85c4619b
1e337e018c8fca
0414175.jpg



danbooru_24976
38_d72cc49997a
4b54e334d91d9
a364bc7d.jpg



danbooru_24977
38_e9ce94577c5
d2259b715e05c
9925099a.jpg



danbooru_24983
47_848c3bc6de3
b9c364997b80c
604d6fb4.jpg



danbooru_24985
83_e6ac56550d2
47d84ecd68bb3
76e30b01.jpg



danbooru_24988
24_5cf4a30dd57
b2444ac28097ce
de6a2dc.png



danbooru_24988
27_5e21bf2511c
96367adfffa9b89
6ab0ce.jpg



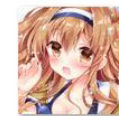
danbooru_24990
44_17040b35bef
42ea72f04d912d
b6af1a4.png



danbooru_24990
78_2c6f24f17c3
b001f056bd764f



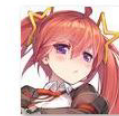
danbooru_24995
01_4f2c60a30f2c
b12d63d480d5a



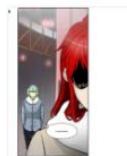
danbooru_24995
34_2a94669474a
a9f36034ccd0f3



danbooru_24996
16_3764f991d47
9c4d781d18110



danbooru_25001
73_9391dfb6e3b
55ff9226d88eb8



183559_330_cut
_024.jpg



183559_330_cut
_041.jpg



183559_330_cut
_046.jpg



183559_330_cut
_066.jpg



183559_330_cut
_075.jpg



183559_330_cut



651673_100_cut



651673_100_cut



651673_100_cut

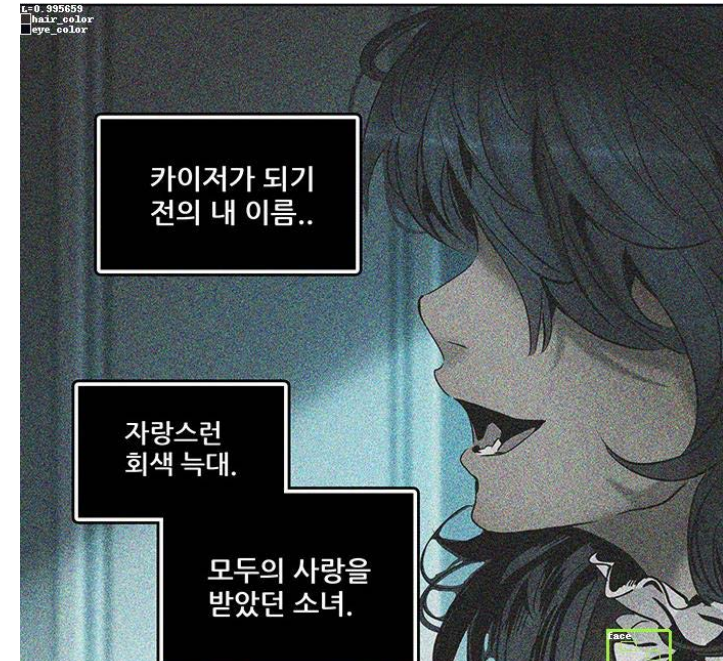


651673_100_cut

1. Cartoon face detector

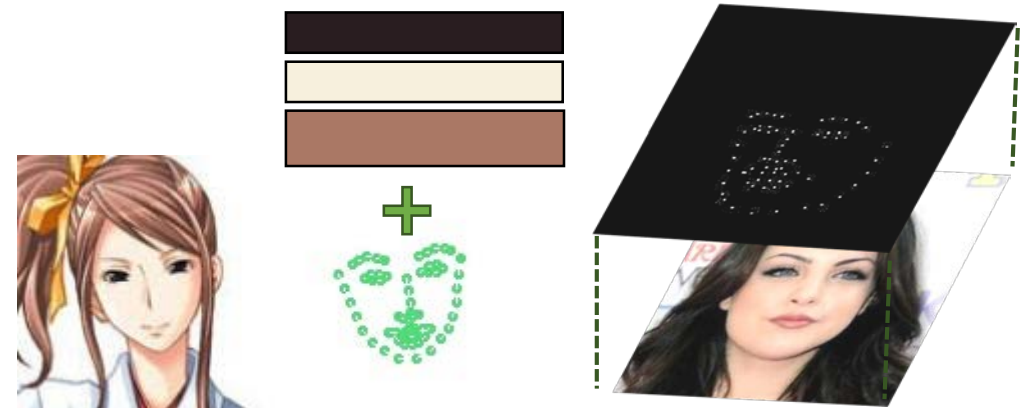
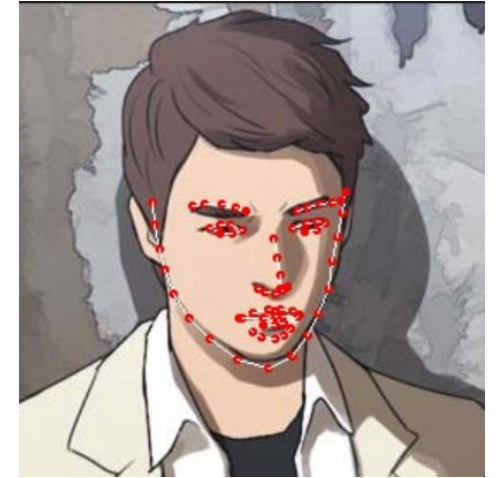
웹툰 적용 시 특징

- 얼굴의 **측면**만 보인 경우 (두 눈이 모두 보이지 않는 경우) **Detect**를 못함.
 - Face translation 등의 목적으로 데이터를 모을 때는 장점이 있음.
- 평균적으로 **15%** 정도의 컷에서 얼굴을 잡아냄
 - 30% 정도의 컷에 얼굴이 없음을 감안해도 매우 낮은 수준.
- 웹툰마다 **심한 성능 편차**를 보이며, 주로 일본 애니메이션 스타일의 웹툰에서 잘 동작함.



2. Facial Information

- 일부 Animation 및 웹툰 데이터의 경우 **실제 얼굴 데이터로 학습된 모델**을 이용해 Attribute 및 Facial information을 가공할 수 있다.
- 실제 얼굴 도메인, 만화 얼굴 도메인 **양쪽에서 추출 가능한 데이터**들이므로 이를 Image translation에 활용한다.
- 자세한 설명은 뒤에서..

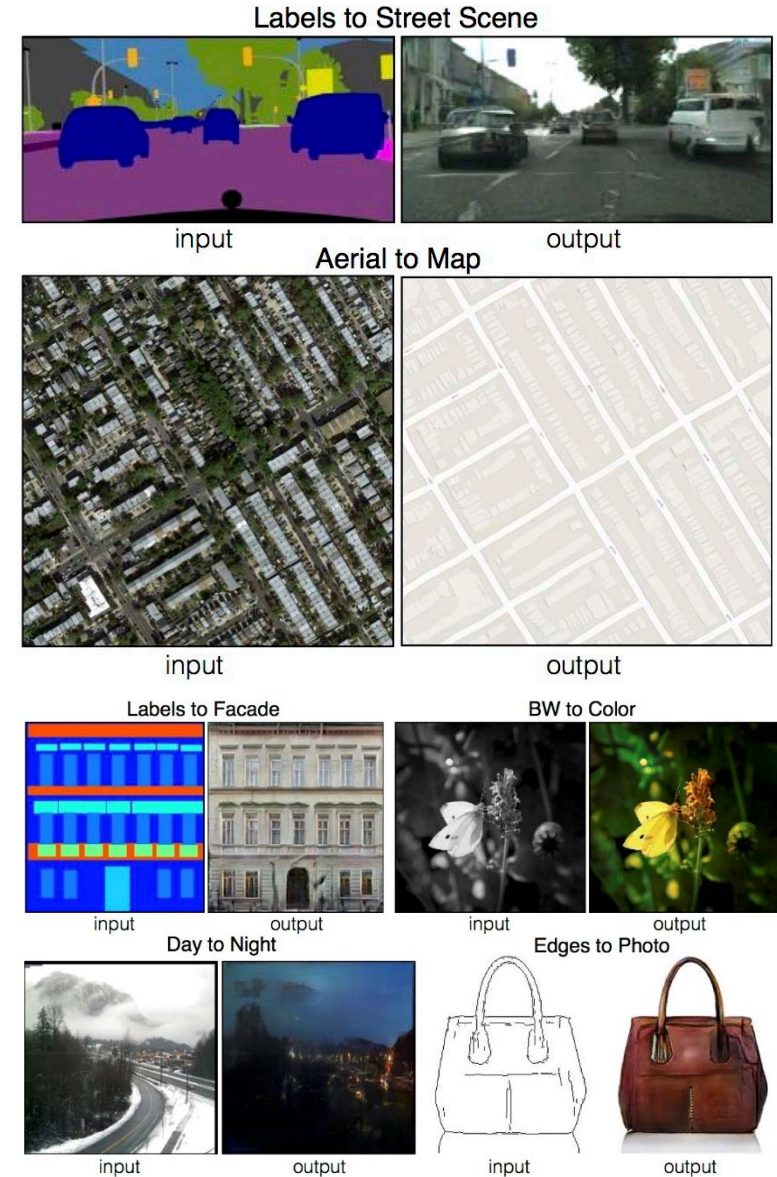


Supervised Image Translation

1. Key points to cartoon face
2. Key points + attributes to cartoon face

Supervised Image Translation

- Supervised Image Translation은 Human face image와 Cartoon face image가 공유할 수 있는 **paired 데이터 포맷**이 이 필요함
- 일반적인 경우 **Paired 데이터가 가공하기 어려워** 적용하기 쉽지 않다는 단점이 있으나, 데이터가 준비된다면 **가장 뛰어난 성능**을 보여줌.
- 앞선 데이터 처리로 Face detection, Key point estimation, color estimation이 가능해짐

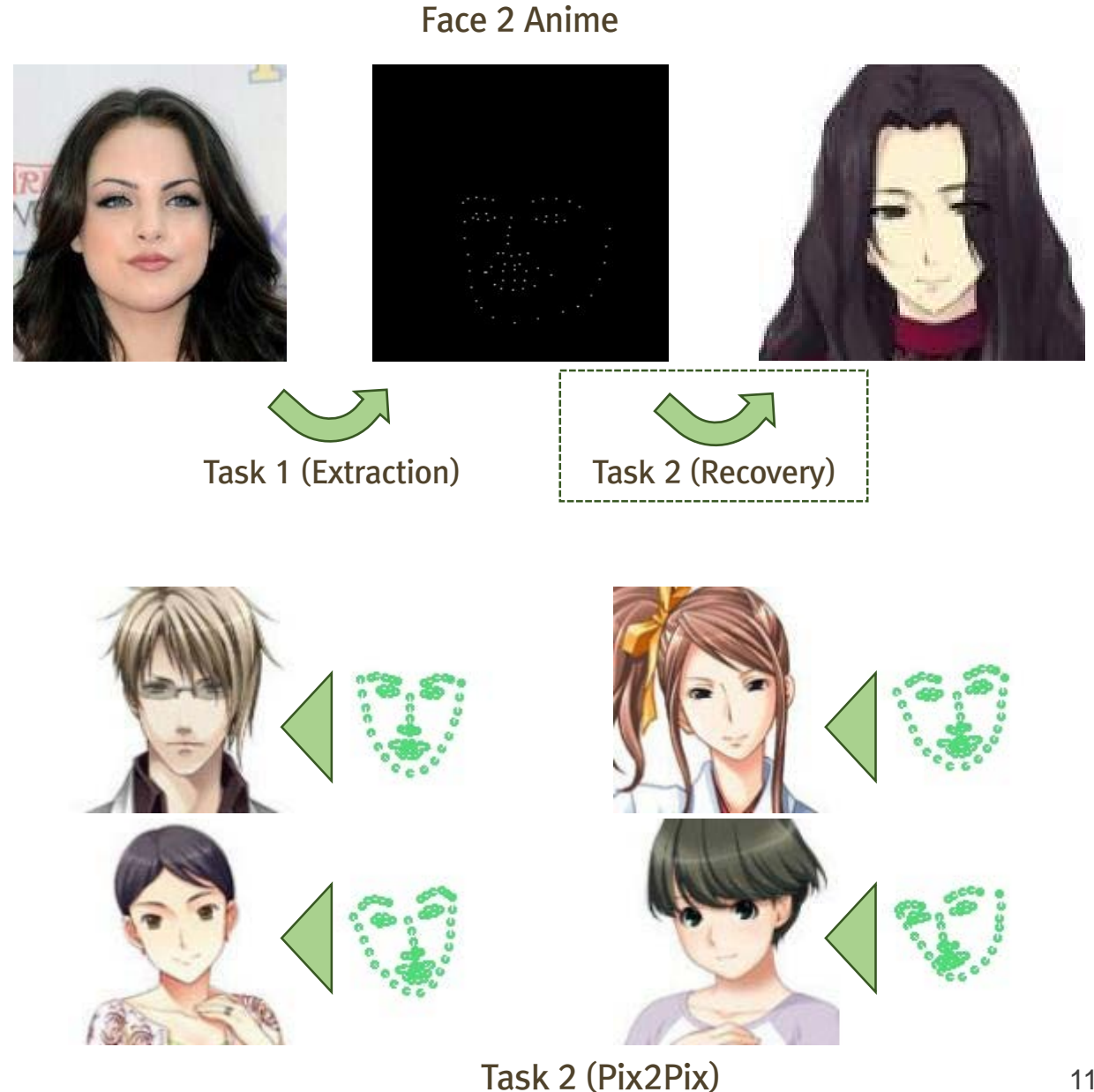


1. Pix2Pix with key points

Approach

- Face 2 Anime task를 다음 2개의 과정으로 나누어 진행할 수 있다.
 1. 사람 얼굴에서 Facial key points를 추출
 2. Facial key points로 Animation 아웃풋 생성
- 위 1번의 과정은 이미 준수한 성능을 보여주는 모델을 사용하여 수행할 수 있다.
- 2번의 경우 Key points – Target image 데이터 페어를 만들어 Supervised translation (pix2pix)를 진행할 수 있다.

목표: 이목구비의 비율과 위치가 자연스럽게 나오도록 2번 학습



1. Pix2Pix with key points

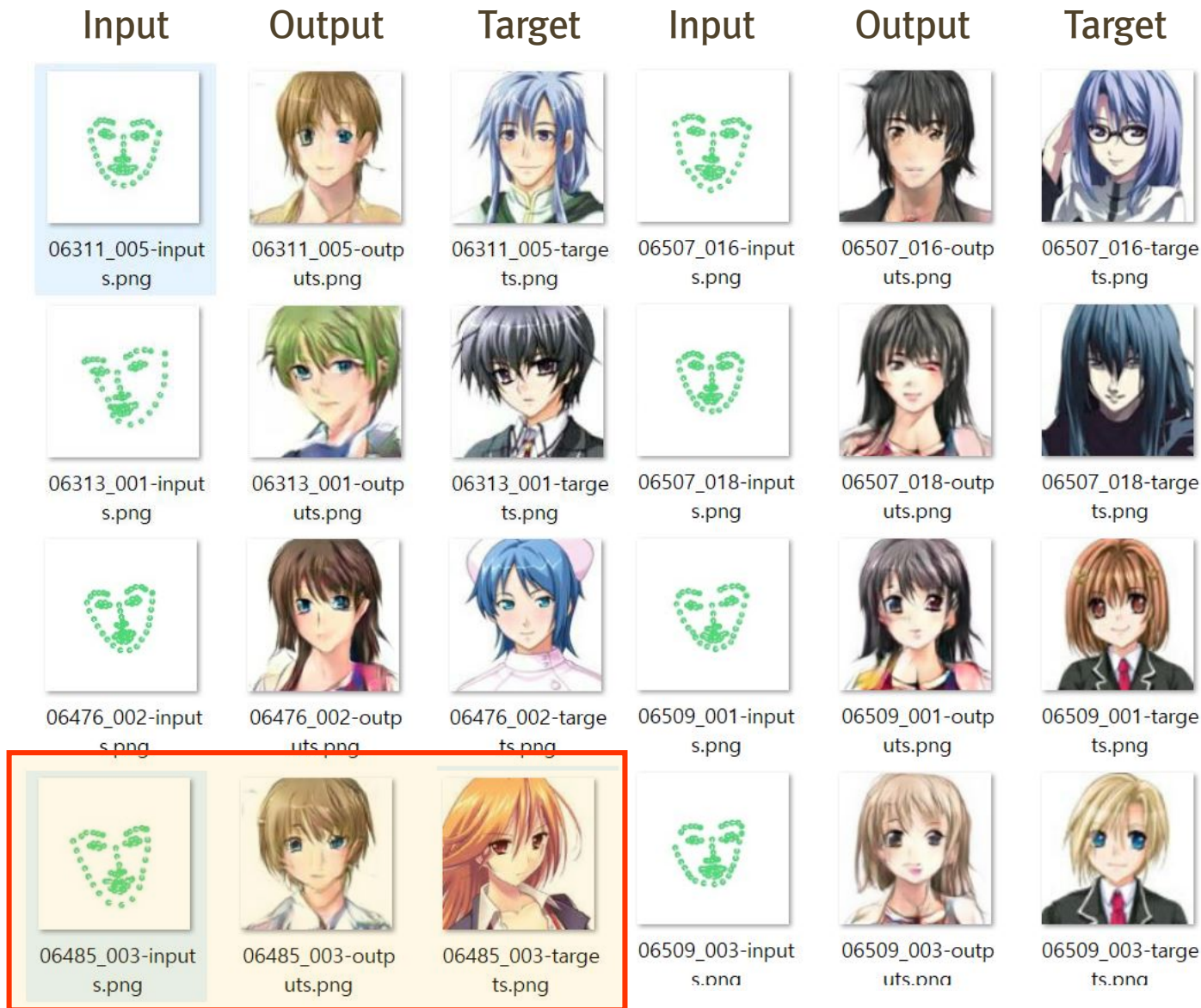
Exp1 Result

Pros

- 본 실험의 목표는 어느정도 달성됨. **이목구비의 비율, 위치가 눈에 띄게 자연스러워** 짐.
- Unsupervised learning과 비교했을 때 **Distortion이 상당히 완화됨**

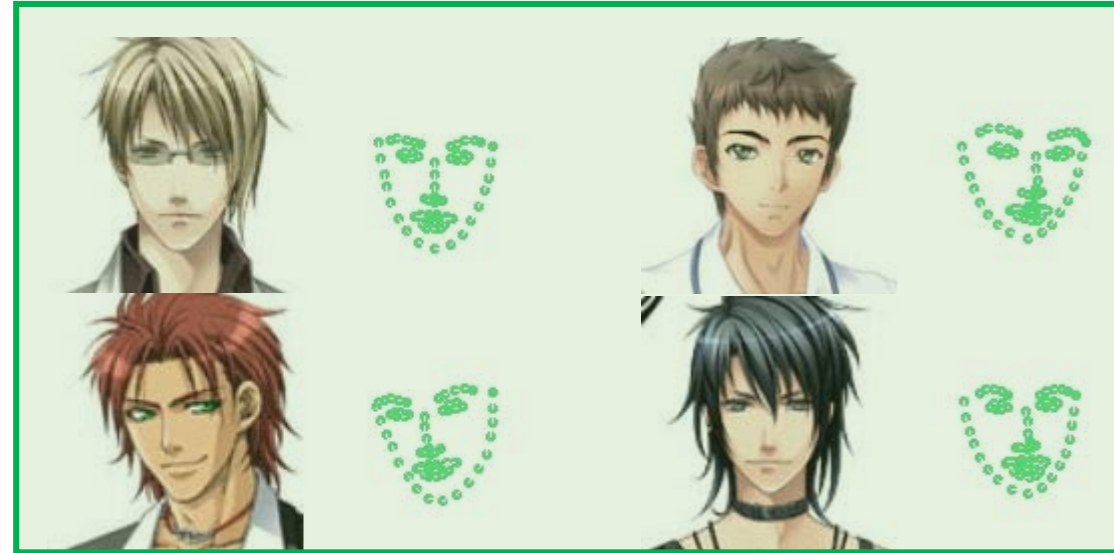
Cons

- 머리,눈 색등의 attributes 정보가 제공되지 않아서 **색깔 및 성별이 랜덤하게** 구현됨
- 박스로 표시한 예시와 같이 **코, 입이 쏠리고 턱이 2줄처럼** 나타나는 유형이 발생함.
 - 키 포인트가 타겟 이미지의 얼굴보다 더 크게 잡힌 경우 발생한다.
- 디테일은 약간 Noisy 함.



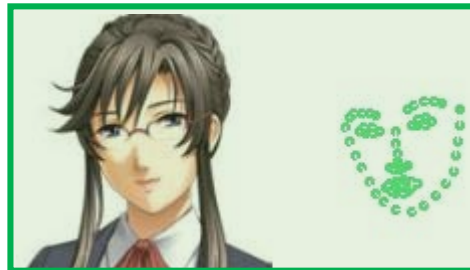
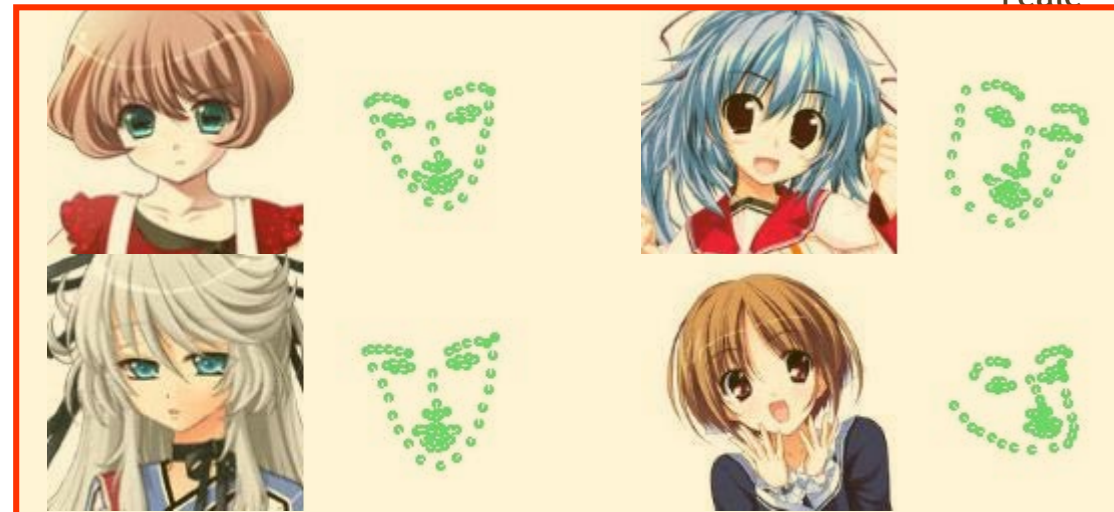
1. Pix2Pix with key points

- 약 절반 정도의 Animation데이터에서 얼굴은 detect 되었으나 Key point estimation이 매우 부정확하게 나오는 케이스가 발생한다.
- 일반적으로 남성 캐릭터의 얼굴은 정확한 반면 여성 캐릭터의 경우 다수의 데이터에서 key points가 부정확하게 찍히는 경향이 있다.
- 관찰 결과, key point의 예측에 있어서 눈이 중요한 랜드마크이며 눈이 클 경우 얼굴 범위를 크게 예측하는 경향이 있는 것으로 예상된다.
- 캐릭터의 남/여 구별이 가능해진다면 좀 더 성능을 올릴 수 있을 것이다.



Male

Female



Natural eye

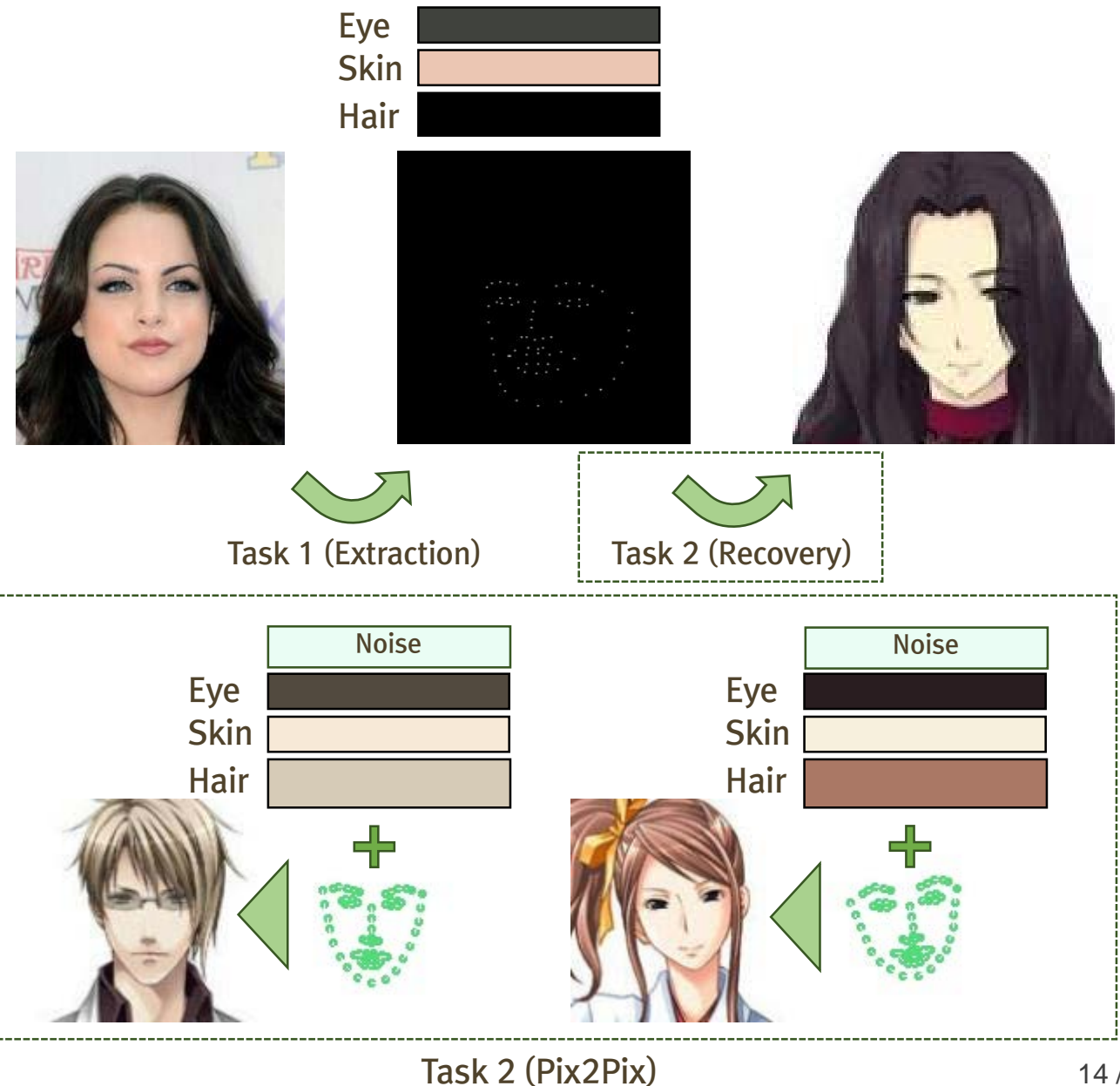
Big eye

2. Pix2Pix with key points + attributes

Approach

- 1번 실험의 단점은 인풋이미지의 **머리 색, 성별** 등을 전혀 반영하지 못한다는 점이다.
- Animation 도메인에서 여전히 성별은 구분할 수 없지만, **피부, 눈, 머리의 색깔**은 앞의 데이터 처리를 통해 측정할 수 있다.
 - 성별의 경우 코드상 구현은 되어있다.
- Key points와 color attributes를 더해서 Pix2Pix 를 수행한다.

목표: **머리색, 피부색, 눈색** 등을 반영한 Pix2Pix Translation을 수행한다.



2. Pix2Pix with key points + attributes

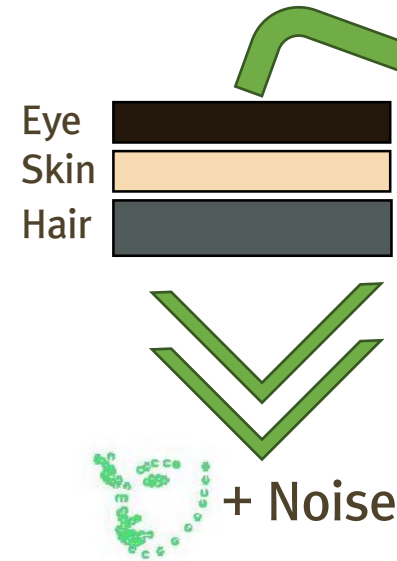
Exp2 Result

Pros

- 목표는 달성됨. 눈색, 머리색이 반영됨

Cons

- 수렴시 타겟 도메인의 데이터를 그대로 외워 버림. 머리, 피부색의 조합으로 타겟 캐릭터를 매핑하는 네트워크가 된 것으로 추측됨.
- 테스트, 트레이닝 데이터를 분리하였는데도 완벽히 기억해내는 것을 보면 캐릭터의 중복 + 사진 자체의 중복이 존재하는 것으로 보임.



recons/image/0
step 199,951 Thu Aug 23 2018 04:55:07 GMT+0900 (한국 표준시)



recons/image/2
step 199,951 Thu Aug 23 2018 04:55:07 GMT+0900 (한국 표준시)



recons/image/4
step 199,951 Thu Aug 23 2018 04:55:07 GMT+0900 (한국 표준시)

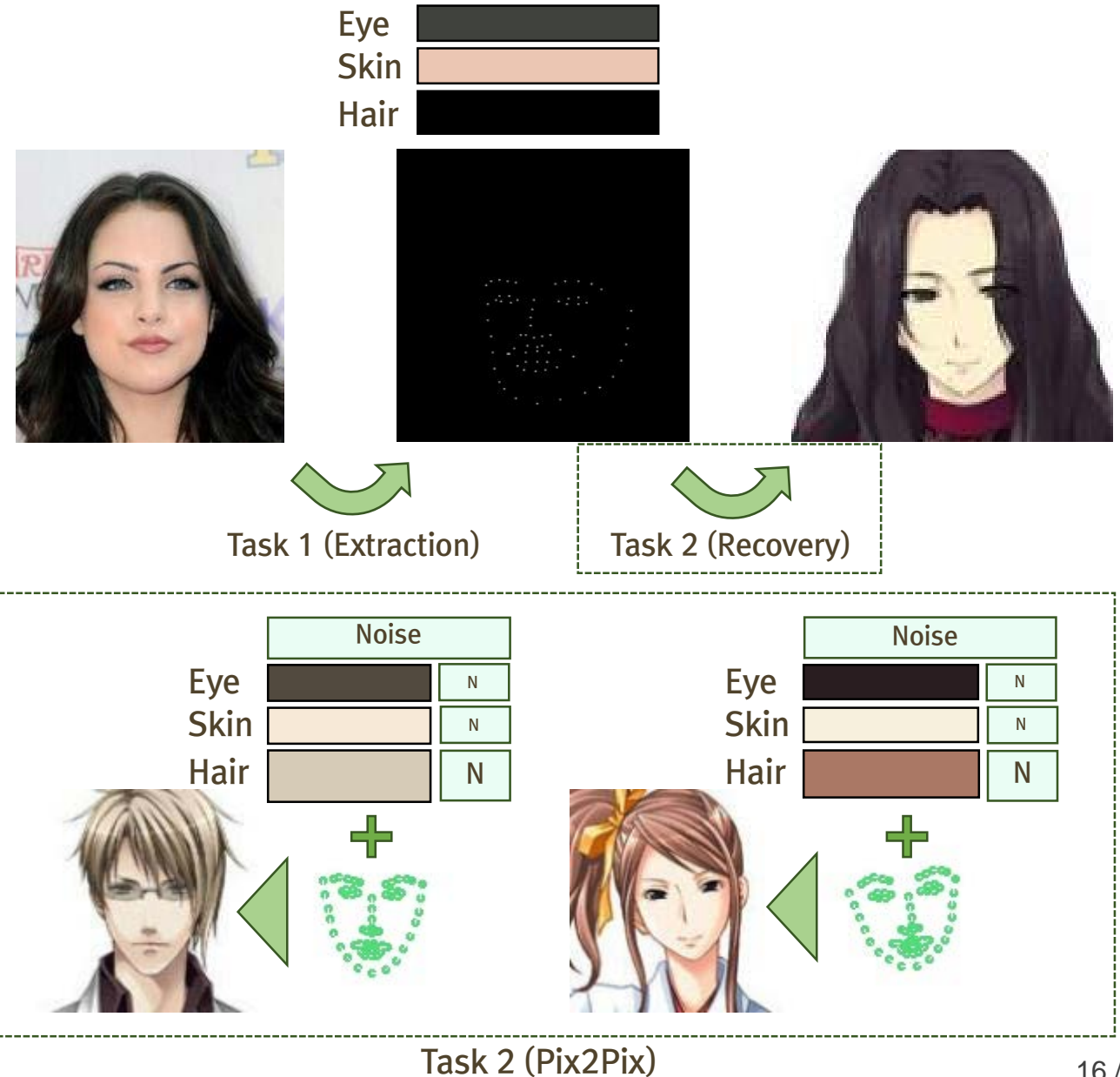


3. Pix2Pix with key points + attributes + randomness

Approach

- 2번 실험의 단점은 인풋이미지의 **머리 색, 피부 색 조합으로 답을 외운다**는 점이다.
- Noise Layer 뿐만 아니라, 머리 색 피부 색 눈 색이 비슷한 범위 내에서 **매번 다르게 입력**되도록 랜덤 노이즈를 넣는다.

목표: 타겟 이미지를 외우고 **머리색, 피부색, 눈색**으로 아웃풋을 매핑하지 않으며, 키포인트 등 모든 정보를 사용하여 이미지를 생성하도록 유도한다.



3. Pix2Pix with key points + attributes

Exp3 Result

Cons

- 살짝 흐려지는 변화가 있었으나 문제가 해결 되지 않았다.
- 두번째 세번째 줄의 경우 키 포인트 마스크가 원래 얼굴 형태를 크게 벗어남에도 완벽히 복원하는 것으로 보아, 네트워크가 **키 포인트를 전혀 반영하지 않는 것으로 예상된다.**



Supervised Image Translation

Contribution

- 본 실험의 목표는 어느정도 달성됨. **이목구비의 비율, 위치가 눈에 띄게 자연스러워** 짐.
- Unsupervised learning과 비교했을 때 **Distortion이 상당히 완화됨**

Further Work

- Keypoint 상태가 좋지 않은 인풋데이터 필터링
- Input Data의 수 및 다양성이 부족함. 데이터 처리를 통해 **Danbooru 데이터**에 대해서도 같은 작업을 할 수 있게 되었으므로 **더 많고 다양한 데이터**를 이용하여 진행하면 외우는 현상이 완화될 수 있음.
- GAN 로스 및 attributes의 공급 방법을 조절하여 **색깔 의존도**를 낮출 수 있음



Unsupervised Image Translation

1. Key points to cartoon face
2. Key points + attributes to cartoon face

Unsupervised image translation

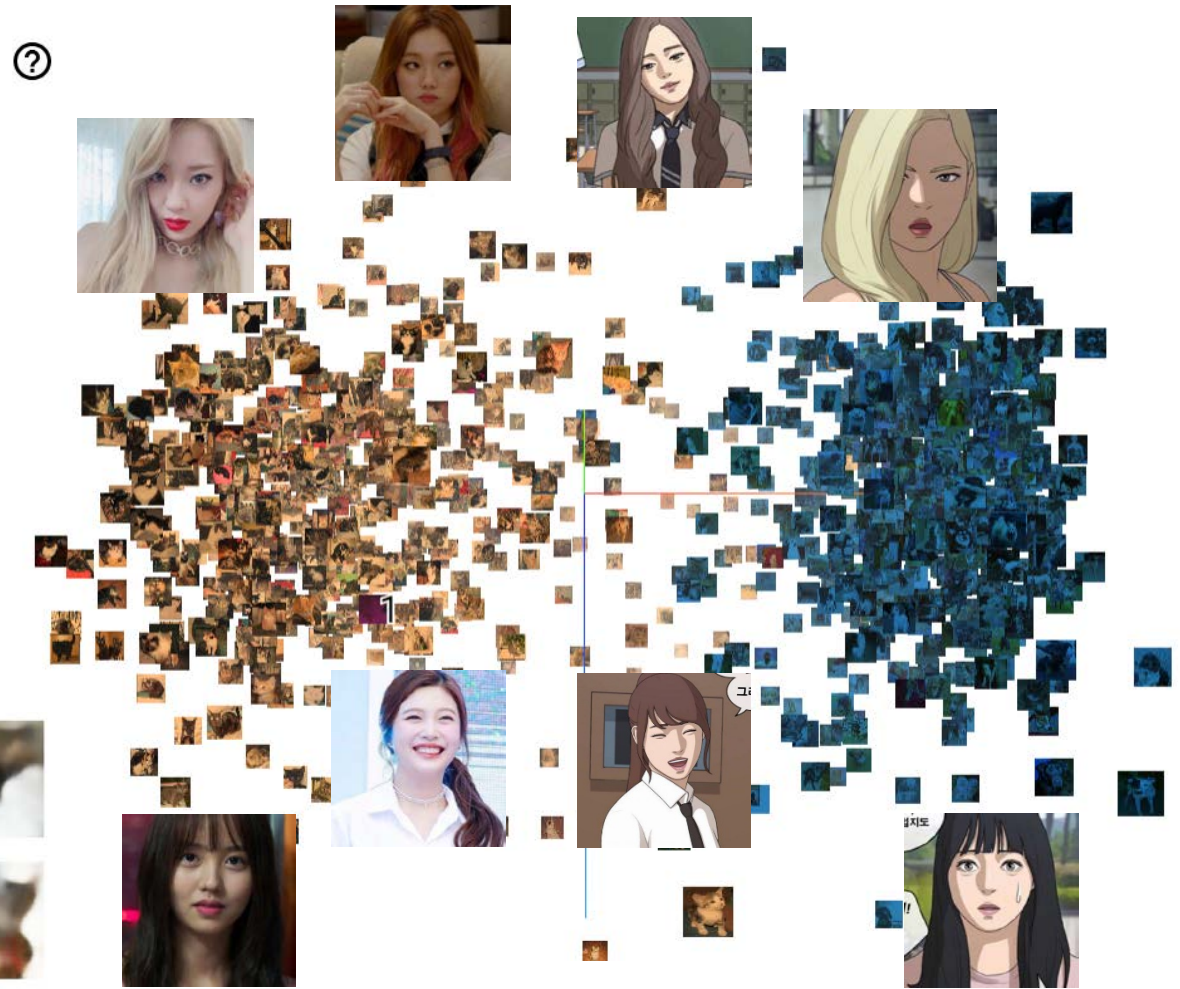
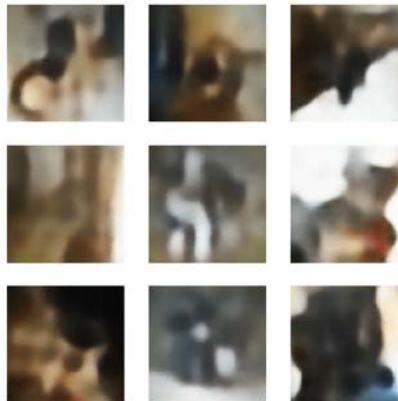
- Unsupervised의 경우 데이터 페어가 따로 필요 없기에 모델을 학습시키기 용이하다는 점이 있다.
- 대신 데이터 페어가 없으므로 소스 이미지가 전환되어야 할 명확한 타겟이 없으므로 학습 자체가 힘들고 퀄리티가 떨어지는 경향이 강하다.
- 기존 모델들의 경우 얼굴 형태가 무너지며 이미지가 상당히 왜곡되는 현상이 있었다.



Cycle GAN850k step, 80 epoch (9일)

1. Unsupervised image translation V1~V3

- Source domain과 Target domain의 이미지들을 **Latent space** 상에서 **분리**하고, 같은 shape를 갖는 이미지들을 모아서 **embedding** 하거나 source to target으로 **mapping**하는 네트워크를 구성
- Embedding의 경우 남은 2주간 공부+개발하기에 무리가 있다고 생각되어 이후 대학원에서 연구해보는 것으로 연기 됨.
- Mapping의 경우 같은 Shape를 갖는 이미지 끼리 매핑하도록 네트워크를 학습시킬 메트릭, 로스를 찾아내지 못해서 **학습이 진행되지 않음**.



2. Cycle GAN with attributes

Current work

- 기존 Cycle GAN의 경우 이미지의 텍스처는 잘 구현
해내지만 형태가 뭉개지는 현상이 강함.
- 구체적으로는 얼굴 형태가 찌그러지고 이목구비가 쓸
리는 등의 현상으로 얼굴의 전체적인 비율이 자연스럽
지 않음.



UGATIT, 동양인 얼굴 & 조석얼굴

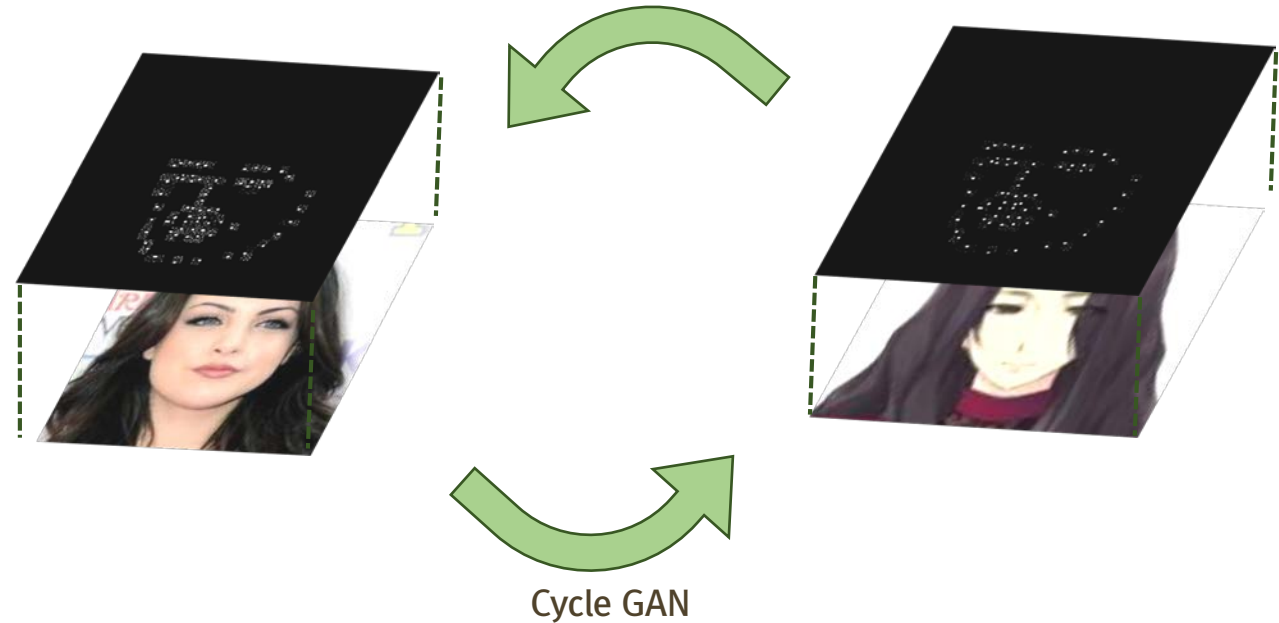


850k step, 80 epoch (9일)

2. Cycle GAN with attributes

Approach

- Source, Target 두 도메인 모두에서 Key point를 뽑을 수 있으므로 네트워크가 **이목구비와 키포인트의 관계를 고려**하여 이미지를 생성하도록 유도한다.
- 이전 UGATIT 프로젝트에서는 **CAM을 통해 어텐션**을 주는 방법으로 비슷한 접근을 하였는데, 랜드마크를 사용하면 **더 섬세한 조정**이 가능할 수 있다.
- Facial Landmark 레이어를 이미지 RGB 채널 위에 Concatenate하여 4d array 데이터를 가공, 학습에 사용하였다.



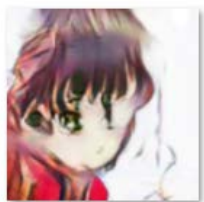
2. Cycle GAN with attributes

Exp Result

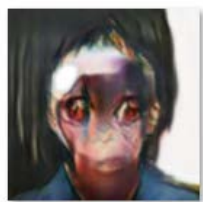
- 160만 스텝 이후 학습을 할 수 없었음. 현 상황으로는 텍스처가 뭉개지지만 **이목구비가 이탈하지 않는 모습**을 보여줌.
- 키포인트가 없던 Cycle GAN의 비슷한 스텝과 비교해 봤을 때 형태를 많이 유지하고 있음.



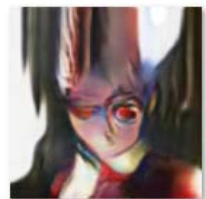
AtoB_021162.jpg



AtoB_021163.jpg



AtoB_021164.jpg



AtoB_021172.jpg



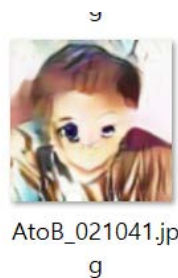
AtoB_021173.jpg



AtoB_021174.jpg



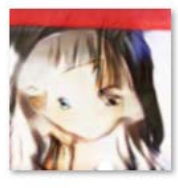
AtoB_021040.jpg



AtoB_021041.jpg



AtoB_021042.jpg



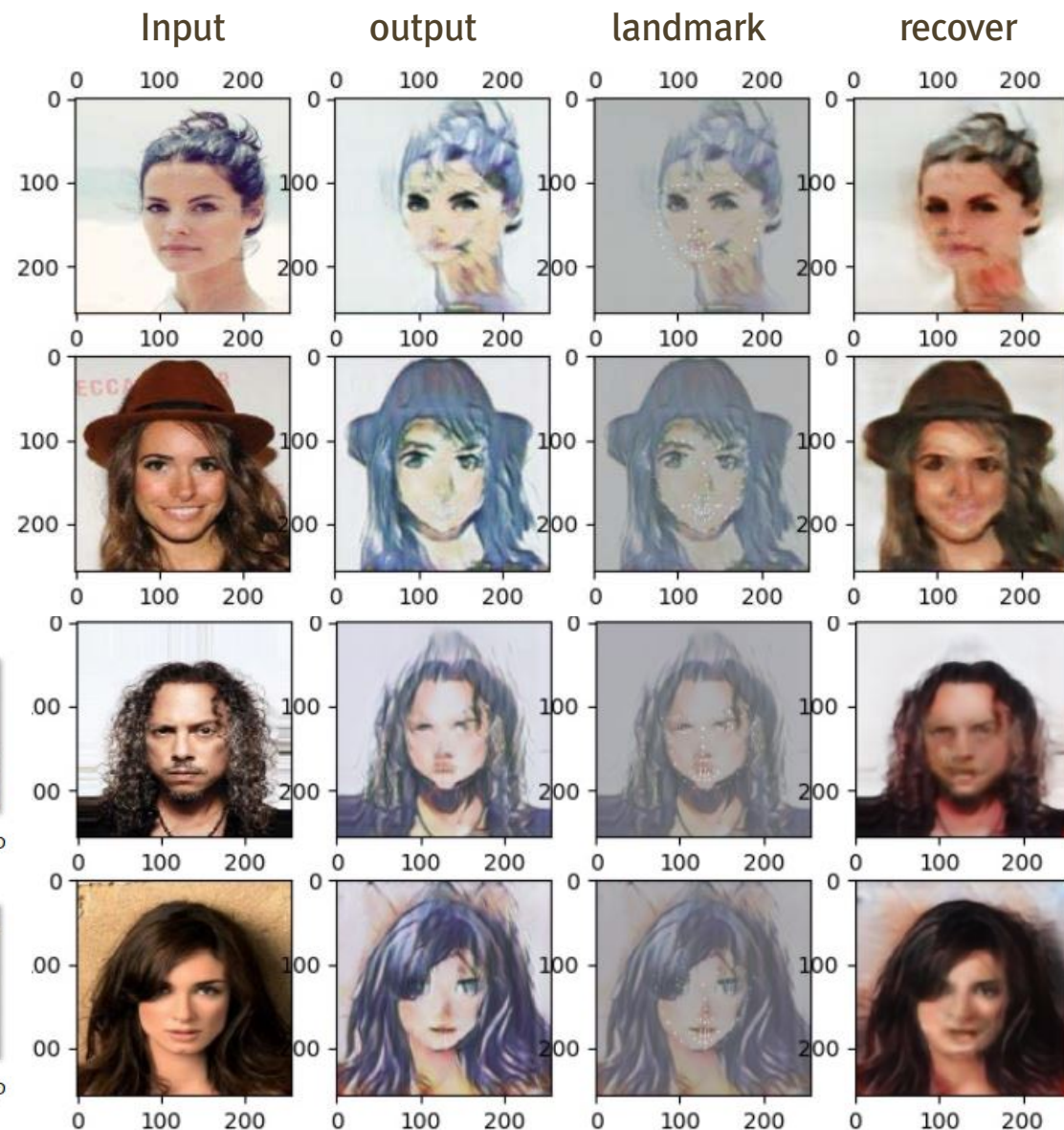
AtoB_021050.jpg



AtoB_021051.jpg



AtoB_021052.jpg



Cycle GAN + Landmarks 140k step

Neat Cycle GAN 120k, 200k step

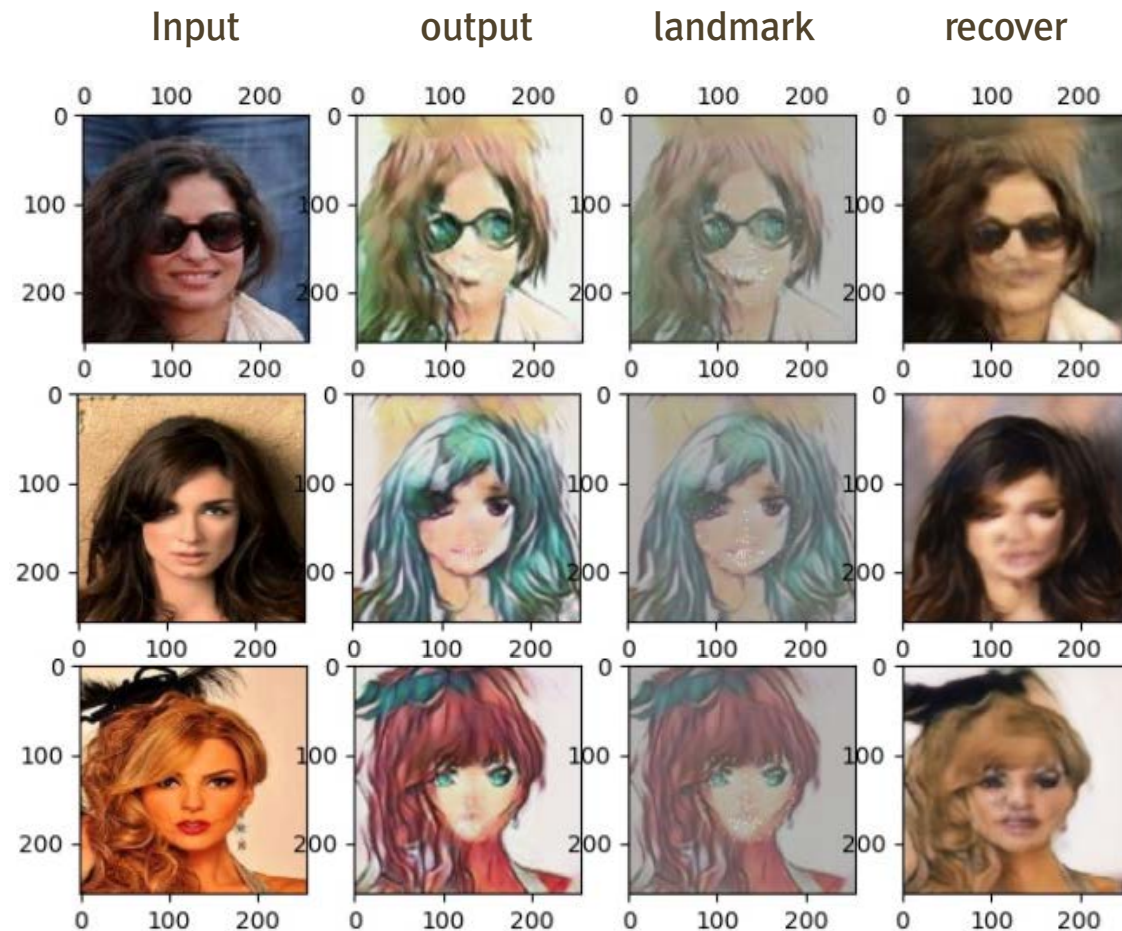
Unsupervised image translation

Exp Result

- 전체적으로 **이목구비의 위치** 뿐만 아니라 **안경, 헤어 스타일, 머리 장식** 등이 구현이 되는 모습을 보여줌.
- 다만 CelebA에 없고 Animation 데이터에만 있는 형광색의 머리색과 눈 색이 구현됨. Unsupervised 특성상 각 도메인의 분포를 따르게 되기 때문에 어쩔 수 없는 현상으로 예상됨.

Further Work

- Output 데이터의 키포인트가 여전히 사람 얼굴형을 하고 있음. 키포인트 레이어를 완전히 **Identity mapping** 하고 있는지 확인이 필요함.
- 기존 Cycle GAN이 텍스처를 온전히 복원하는데 까지 9일(800만 스텝)이 소요되었음. 반면 본 실험은 재구현도 학습이 덜 된 상태이므로 최소한 2배 이상 트레이닝을 더 해봐야 함.
- **키포인트로 어텐션**을 주는 것도 좋은 방법.



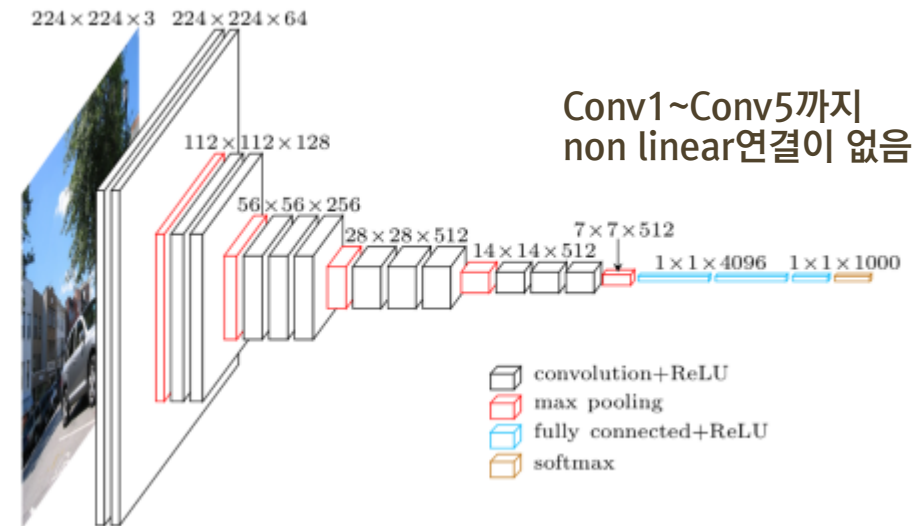
Cycle GAN + Landmarks 160k step

Tag Estimation

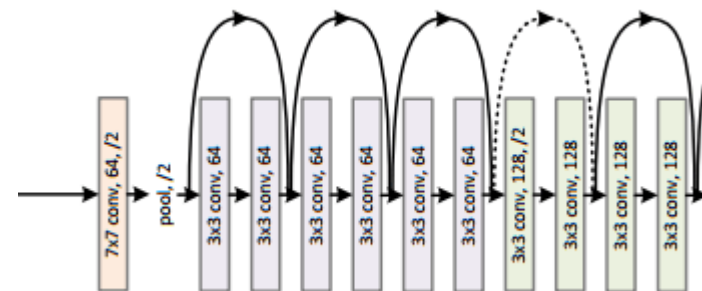
1. Backbone Network Trainer

1. Backbone Network Trainer

- Tag estimator 프로젝트의 큰 Contribution중 하나가 만화 이미지를 해석할 수 있는 네트워크 트레이닝이다.
- CV 모든 분야에서 사람들이 가장 성능이 뛰어난 네트워크를 쓰는 것이 아니라, 각 Task 마다 사람들이 선호하는 네트워크 구조가 있다.
 - Style transfer의 경우 2018 NIPS에 발표된 연구가 VGG16을 사용함.
 - Embedding의 경우 2017ICML에 발표된 SOTA 연구가 Inception V2를 사용함.
 - 세그멘테이션, 디텍션, 어텐션 등등의 Vision task에서는 주로 Resnet 을 사용함.
- 따라서 텐서플로우가 공식 지원하는 대중적인 네트워크 구조를 학습시킬 수 있는 학습기를 만들었다.



특히 V2. 뒤엔 FC가 없고
앞엔 skip connection이 없음



이미지 해상 능력이 셋중 가장
높고 개조하기 매우 쉬움

1. Backbone Network Trainer

- 우선 가장 대중적인 3개 모델(VGG, Inception, ResNet)에 대해 구현되어 있으며 **텐서플로우가 하단 링크에서 지원하는 모델은 4줄정도 수정으로 학습할 수 있다.**
<https://github.com/tensorflow/models/tree/master/research/slim>
- ImageNet 데이터로 pretrained 된 모델이라 수렴 속도가 빠르다. VGG_19 모델로 140만 스텝을 학습한 결과 정확도 약 88% 정도를 기록한다.

```
Step: 1424950k      Avg. cost = 0.33000      Acc: 0.89700
Step: 1424960k      Avg. cost = 0.33411      Acc: 0.89559
Step: 1424970k      Avg. cost = 0.32413      Acc: 0.90147
Step: 1424980k      Avg. cost = 0.36813      Acc: 0.87941
Step: 1424990k      Avg. cost = 0.34280      Acc: 0.89118
Step: 1425000k      Avg. cost = 0.35787      Acc: 0.88235
Accuracy: 0.882353
```

Model	TF-Slim File	Checkpoint	Top-1 Accuracy	Top-5 Accuracy
Inception V1	Code	inception_v1_2016_08_28.tar.gz	69.8	89.6
Inception V2	Code	inception_v2_2016_08_28.tar.gz	73.9	91.8
Inception V3	Code	inception_v3_2016_08_28.tar.gz	78.0	93.9
Inception V4	Code	inception_v4_2016_09_09.tar.gz	80.2	95.2
Inception-ResNet-v2	Code	inception_resnet_v2_2016_08_30.tar.gz	80.4	95.3
ResNet V1 50	Code	resnet_v1_50_2016_08_28.tar.gz	75.2	92.2
ResNet V1 101	Code	resnet_v1_101_2016_08_28.tar.gz	76.4	92.9
ResNet V1 152	Code	resnet_v1_152_2016_08_28.tar.gz	76.8	93.2
ResNet V2 50^	Code	resnet_v2_50_2017_04_14.tar.gz	75.6	92.8
ResNet V2 101^	Code	resnet_v2_101_2017_04_14.tar.gz	77.0	93.7
ResNet V2 152^	Code	resnet_v2_152_2017_04_14.tar.gz	77.8	94.1
ResNet V2 200	Code	TBA	79.9*	95.2*
VGG 16	Code	vgg_16_2016_08_28.tar.gz	71.5	89.8
VGG 19	Code	vgg_19_2016_08_28.tar.gz	71.1	89.8
MobileNet_v1_1.0_224	Code	mobilenet_v1_1.0_224.tgz	70.9	89.9
MobileNet_v1_0.50_160	Code	mobilenet_v1_0.50_160.tgz	59.1	81.9
MobileNet_v1_0.25_128	Code	mobilenet_v1_0.25_128.tgz	41.5	66.3
MobileNet_v2_1.4_224^*	Code	mobilenet_v2_1.4_224.tgz	74.9	92.5
MobileNet_v2_1.0_224^*	Code	mobilenet_v2_1.0_224.tgz	71.9	91.0
NASNet-A_Mobile_224#	Code	nasnet-a_mobile_04_10_2017.tar.gz	74.0	91.6
NASNet-A_Large_331#	Code	nasnet-a_large_04_10_2017.tar.gz	82.7	96.2
PNASNet-5_Large_331	Code	pnasnet-5_large_2017_12_13.tar.gz	82.9	96.2

감사합니다

Trial and Error

- Cycle GAN 및 Image 2 Image translation에서 학습을 진행해도 계속 여러 이미지가 겹쳐서 보이는 현상이 있었다.
- 이유는 명확히 모르겠으나 경험적으로는 **Batch Size 1**로 잡고 학습을 해야 생성이 바르게 잘 된다고 한다.



Trial and Error

- 4D 데이터를 만들면서 Land mark를 1, Default 배경을 0으로 주고 학습을 했을 때 번번히 웨이트가 폭발하는 현상이 있었다.
- 정확히 이유는 모르겠으나, 데이터의 ¼를 차지하는 한 레이어에서 99%가 이미지를 가리지 않고 0이다 보니 항상 Negative value를 생성하도록 gradient를 받아서 explode 한 것이 아닐까 예상함.
- Default를 127(normalize시 0)으로 주니 폭발하지 않게 됨.

