

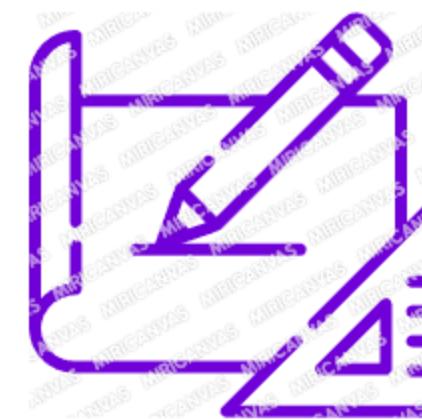
# 정장 증명사진 프로젝트

일반 증명사진에게 정장을 입혀주는 AI



발표: 나 혼자서 팀의 홍수호

# student's contents



01  
**AI 소개**

03  
**구현과정과 시행착오**

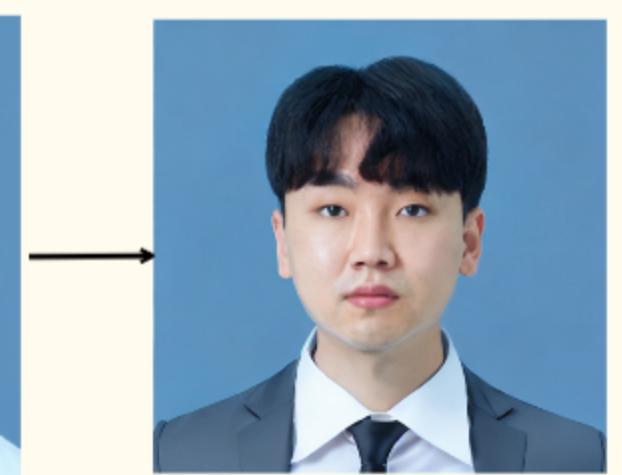
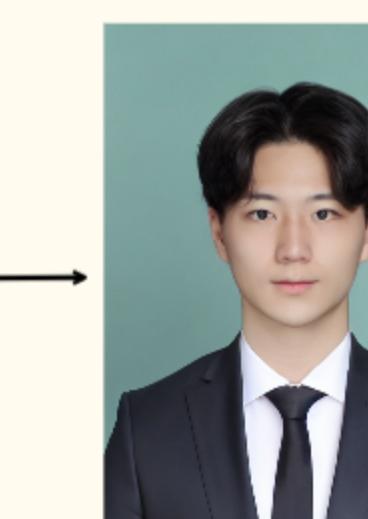
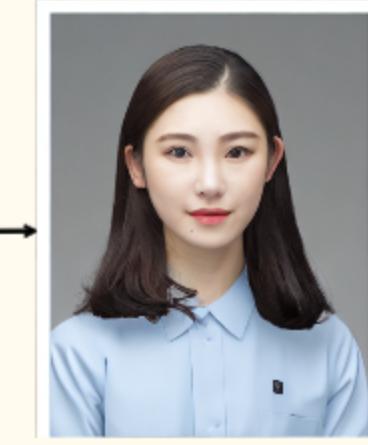
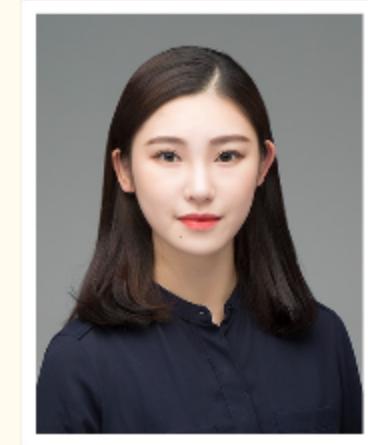
05  
**피드백**

대표 성공작 AI를 이용하여 1번사진을 2번으로 합성하였습니다.

1



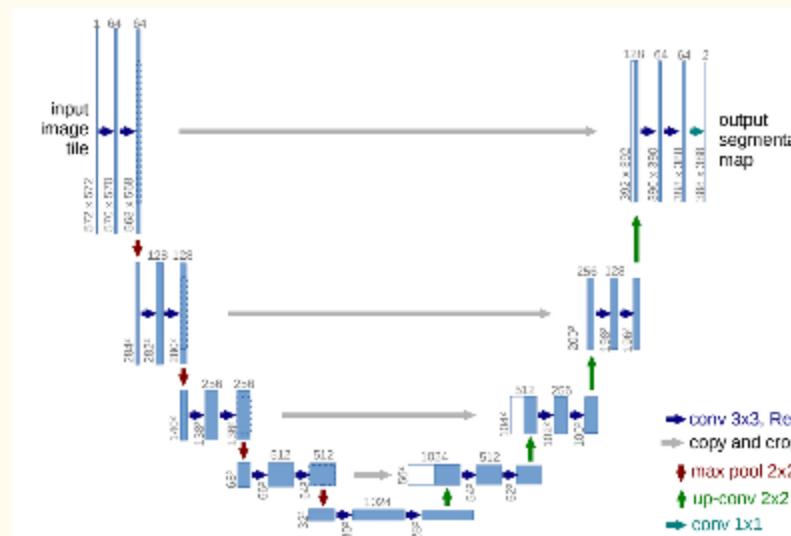
2



실패작은 10배나 더 많다



unet



유넷은 이미지 세그먼테이션을 할 때 사용되는 모델이다.  
작은 수의 데이터로 정확한 이미지 세그먼테이션이 가능하다.  
주로 미세한 분류작업이 필요한 의료 이미지 분석에서 사용이  
된다.

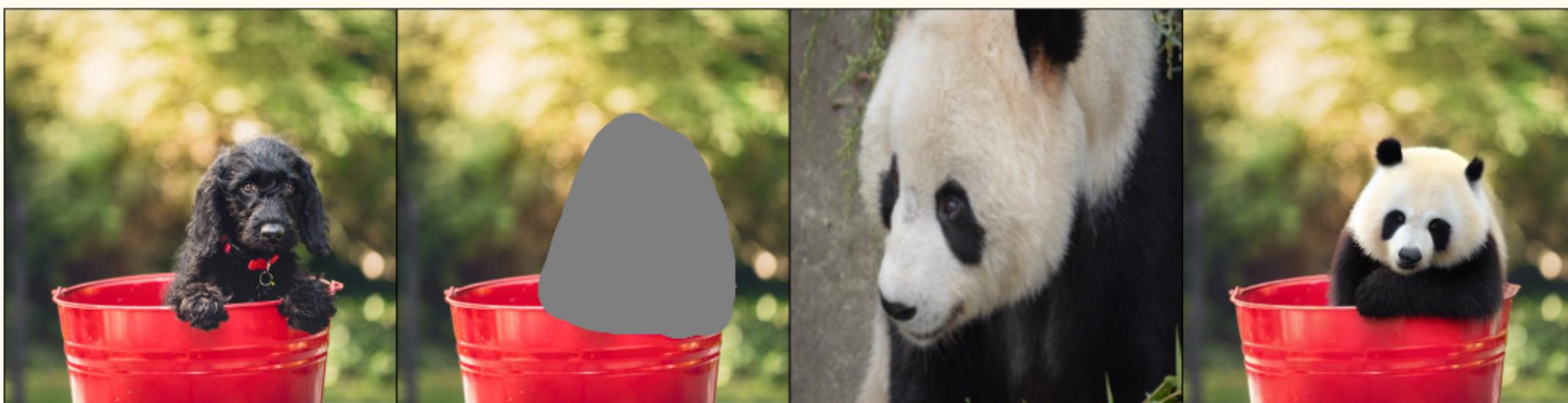
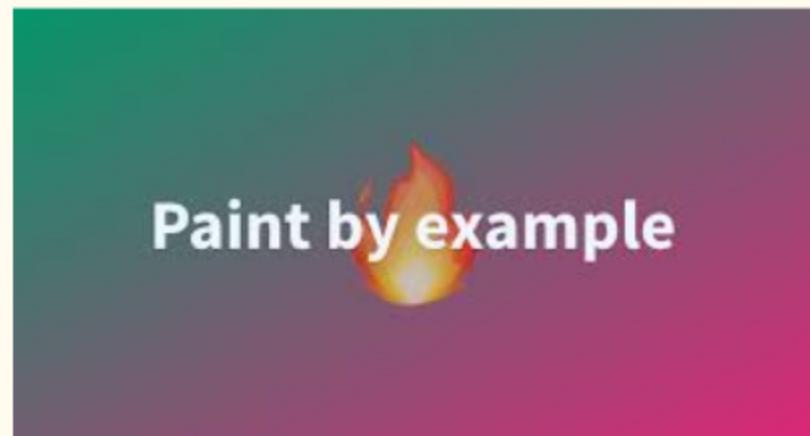
fantasy studio의 paint by example



스테이블 디퓨전 1.4버전의 체크포인트를 이어 받아 만들어  
진 모델이다. 프롬프트로 이미지를 생성하는 스테이블 디퓨전  
과 달리 이미지를 예시로 받아 마스크와 함께 조합하면 합성  
을 해준다.

유넷을 이용하여 이미지의 얼굴 마스크를 생성했습니다. 얼굴  
마스크로 사진의 얼굴을 본떠 배경과 분리하였습니다. 얼굴과  
정장 사진의 이미지를 합성하여 새로운 정장 이미지를 생성했  
습니다.

paint by example은 스테이블 디퓨전에서 전이학습됐다.  
마스크와 이미지를 합성하여 새로운 이미지를 만들어낸다.

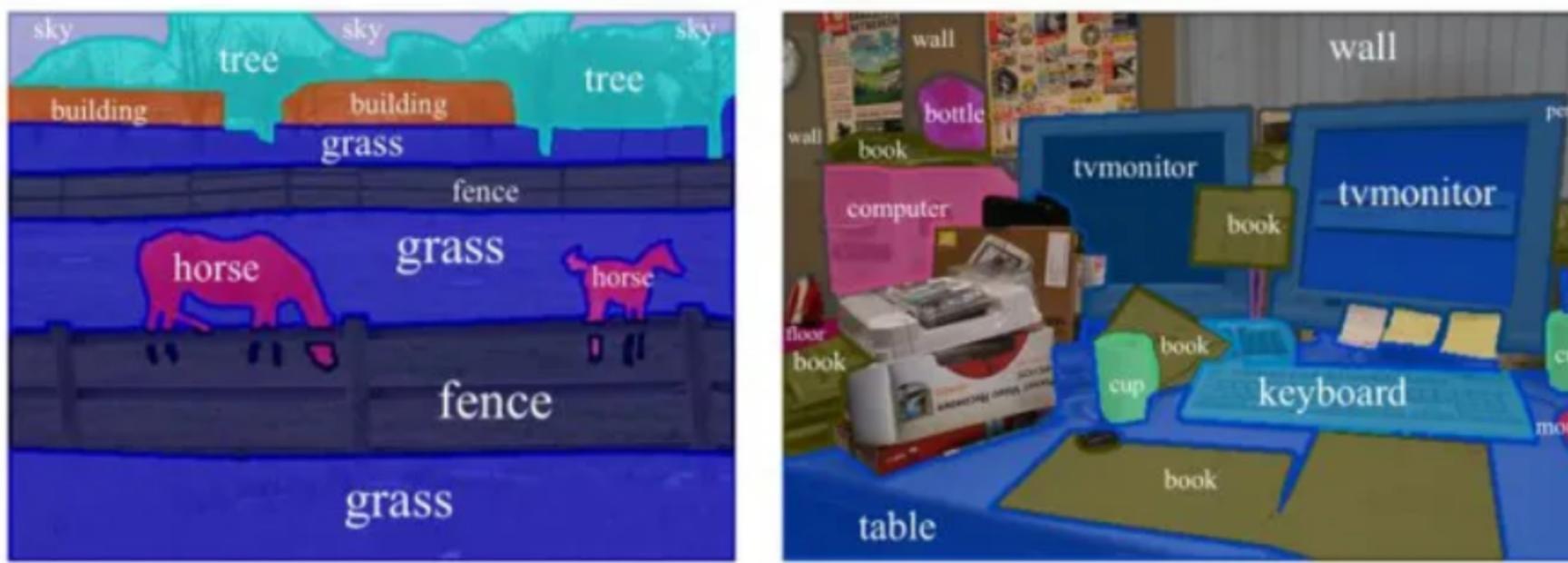


## AI 소개 - 어떻게 했니?



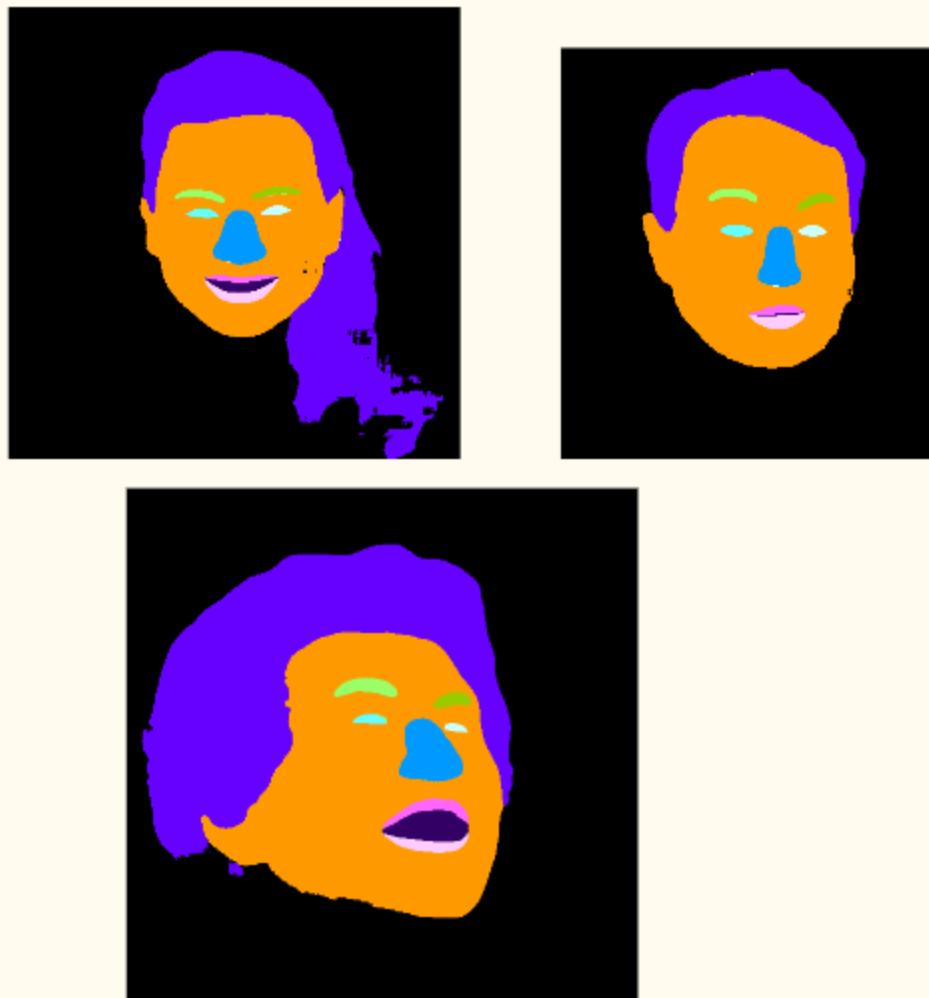
이미지 세그먼테이션은 크게 두 분류로 나눠어진다. 사진 전체를 분류해서 의미를 갖게 하는 시멘틱 세그먼테이션. 중요한 인스턴스만 분류하는 인스턴스 세그먼테이션.

시멘틱 세그먼테이션 (의미적인 분류라고 한다.)

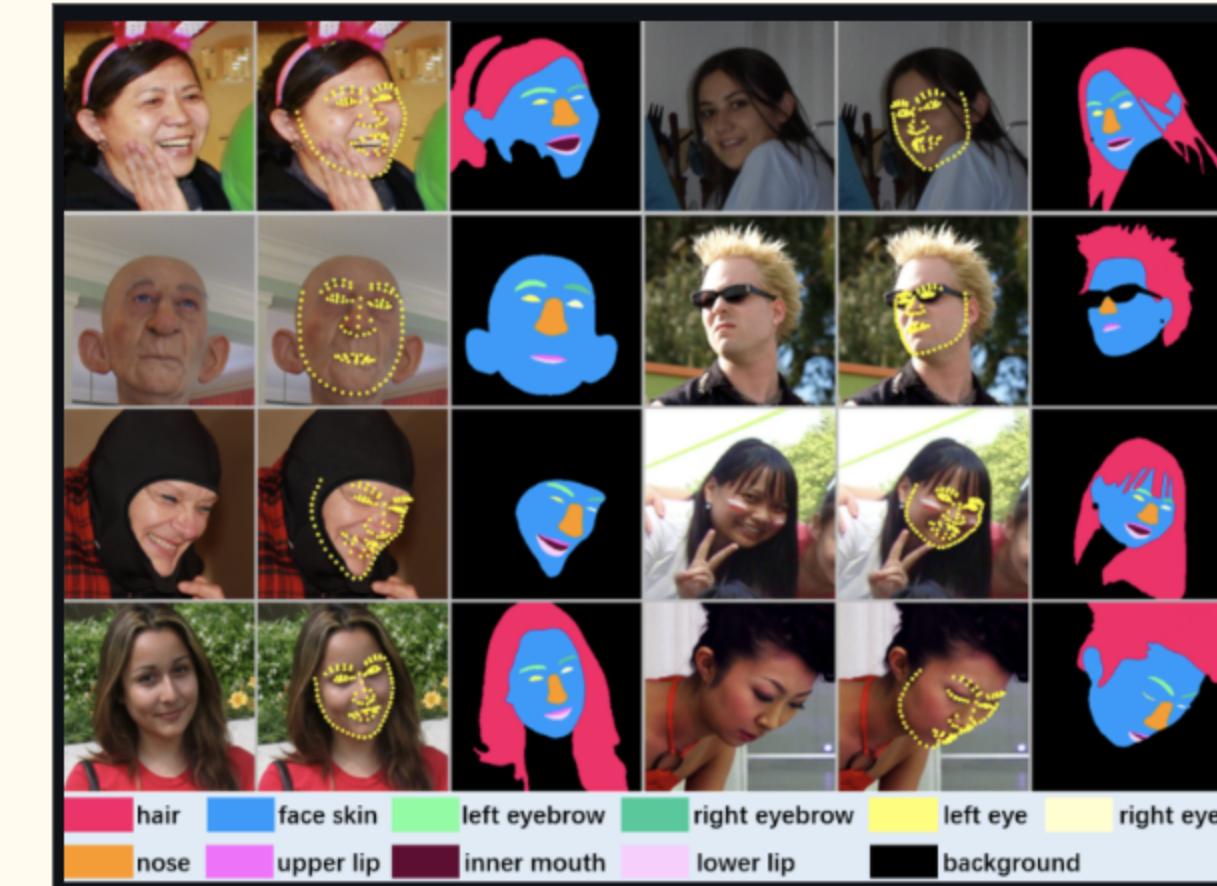


인스턴스 세그먼테이션



**face parsing**

시멘틱 세그먼테이션 중에서 얼굴을 분석하는 페이스 파싱이라는 분야의 기술을 이용해 마스크를 땠습니다. 페이스 파싱은 이미지의 피부, 눈, 입, 코와 같은 얼굴의 각 부위를 픽셀 단위로 판별합니다.

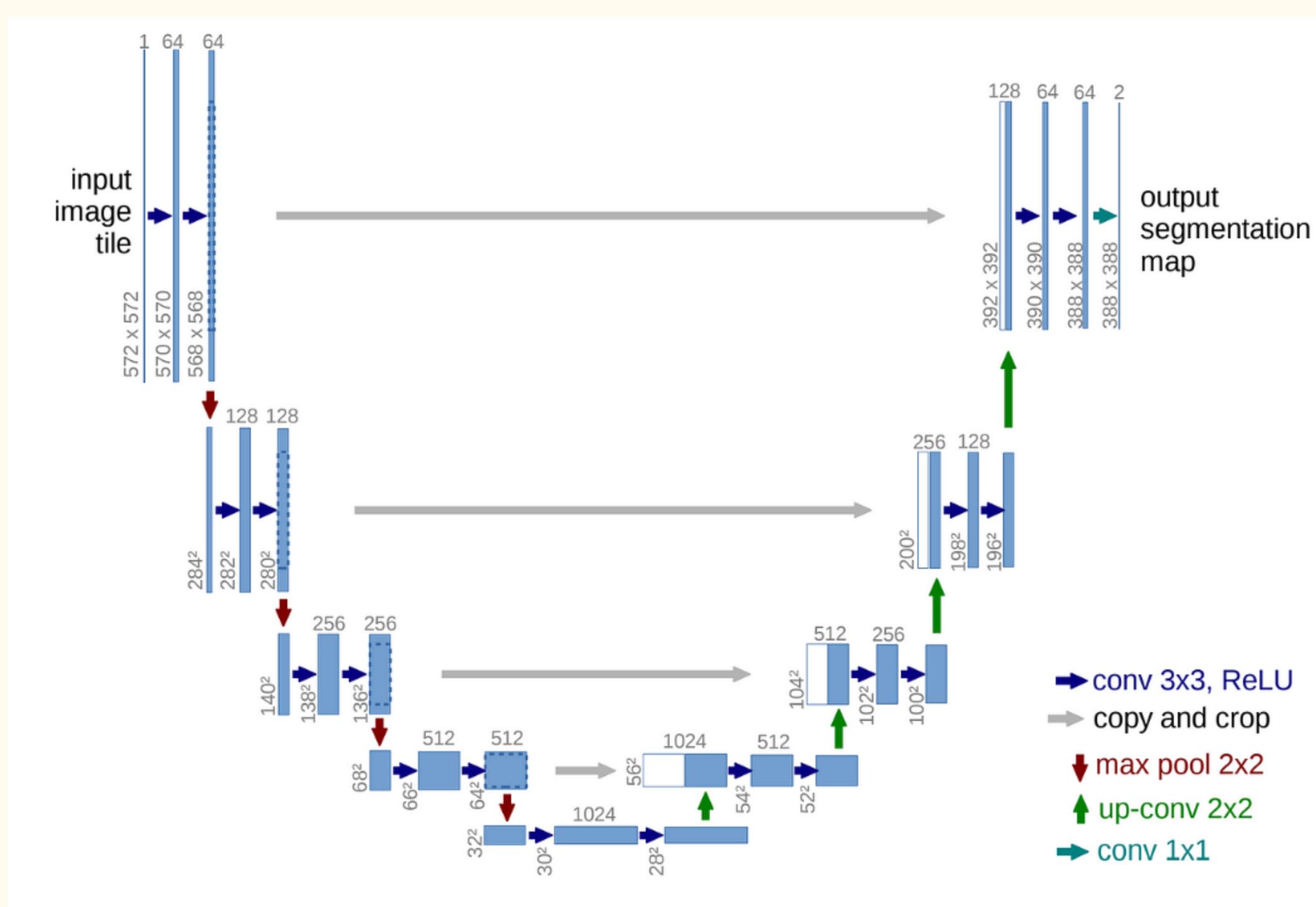
**lapa 데이터셋**

라파는 large scale landmark guided parsing dataset for face parsing입니다. 이만 이천개의 데이터셋으로 이루어져 있습니다. 11개의 부위로 나뉘어져 있습니다.

관련 사이트

<https://paperswithcode.com/sota/face-parsing-on-lapa>

## 유넷 (u모양이라서 유넷)



# 컨볼루션 함수

```
def conv_block(inputs, num_filters):
    x = Conv2D(num_filters, 3, padding="same")(inputs)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)

    x = Conv2D(num_filters, 3, padding="same")(x)
    x = BatchNormalization()(x)
    x = Activation("relu")(x)

    return x
```

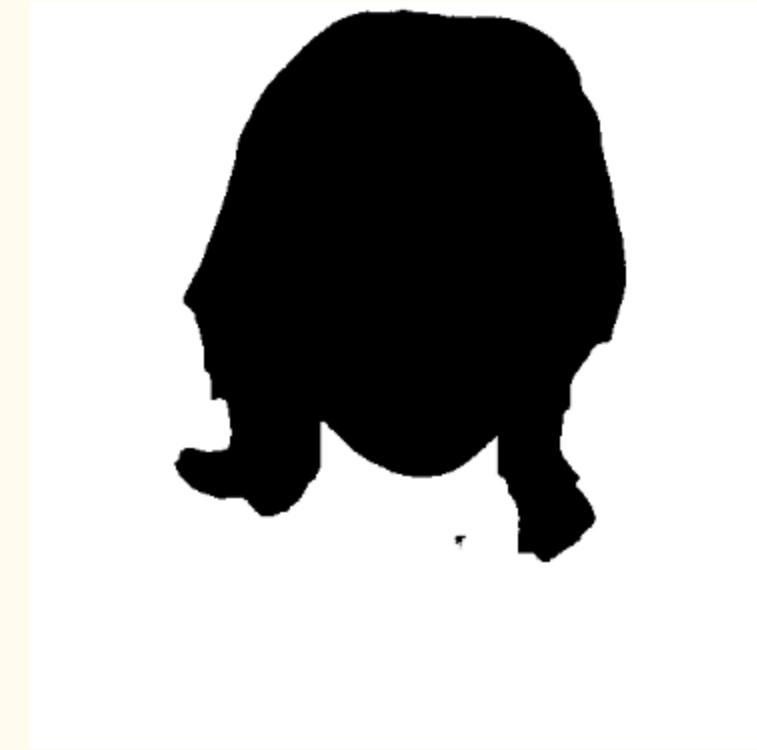
# 맥스풀링 함수

```
def encoder_block(inputs, num_filters):
    x = conv_block(inputs, num_filters)
    p = MaxPool2D((2, 2))(x)
    return x, p
```

# 업컨볼루션 함수

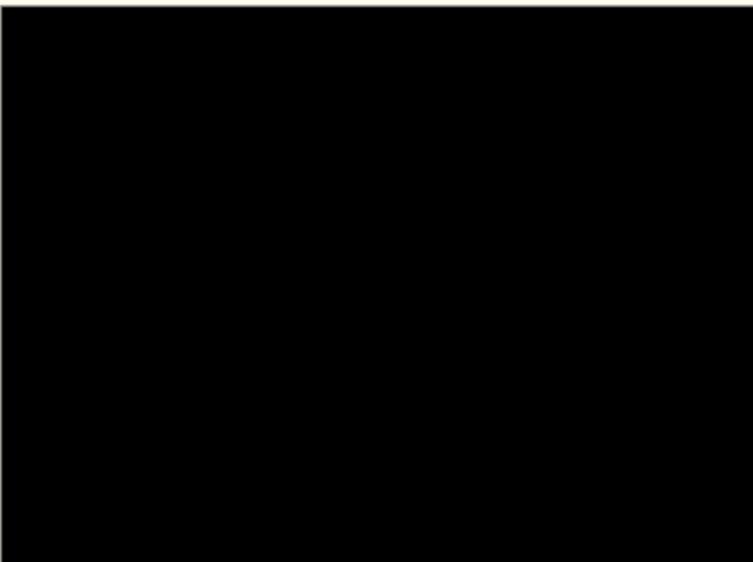
```
def decoder_block(inputs, skip, num_filters):
    # 일반적인 컨볼루션의 반대 방향으로 진행 up-conv
    x = Conv2DTranspose(num_filters, (2, 2), strides=2, padding="same")(
        inputs) # 스트라이드가 2라서 크기가 2배로 증가 32->64
    x = Concatenate()([x, skip]) # [64x64 512, 64x64 512] -> 64x64 1024
    x = conv_block(x, num_filters) # 64x64, 1024 -> 64x64, 512
    return x
```

착각. 상상은 쉽다. 얼굴 마스크를 이용해  
봤자 머리카락 사이의 공백이 남는다.



힘들게 얼굴 마스크를 구현했지만,  
쓸모가 없는걸까?

라인만 그은 마스크가 더 좋은 효과  
를 낸다. 하지만 이는 특정한 부분  
에서만 그렇다.





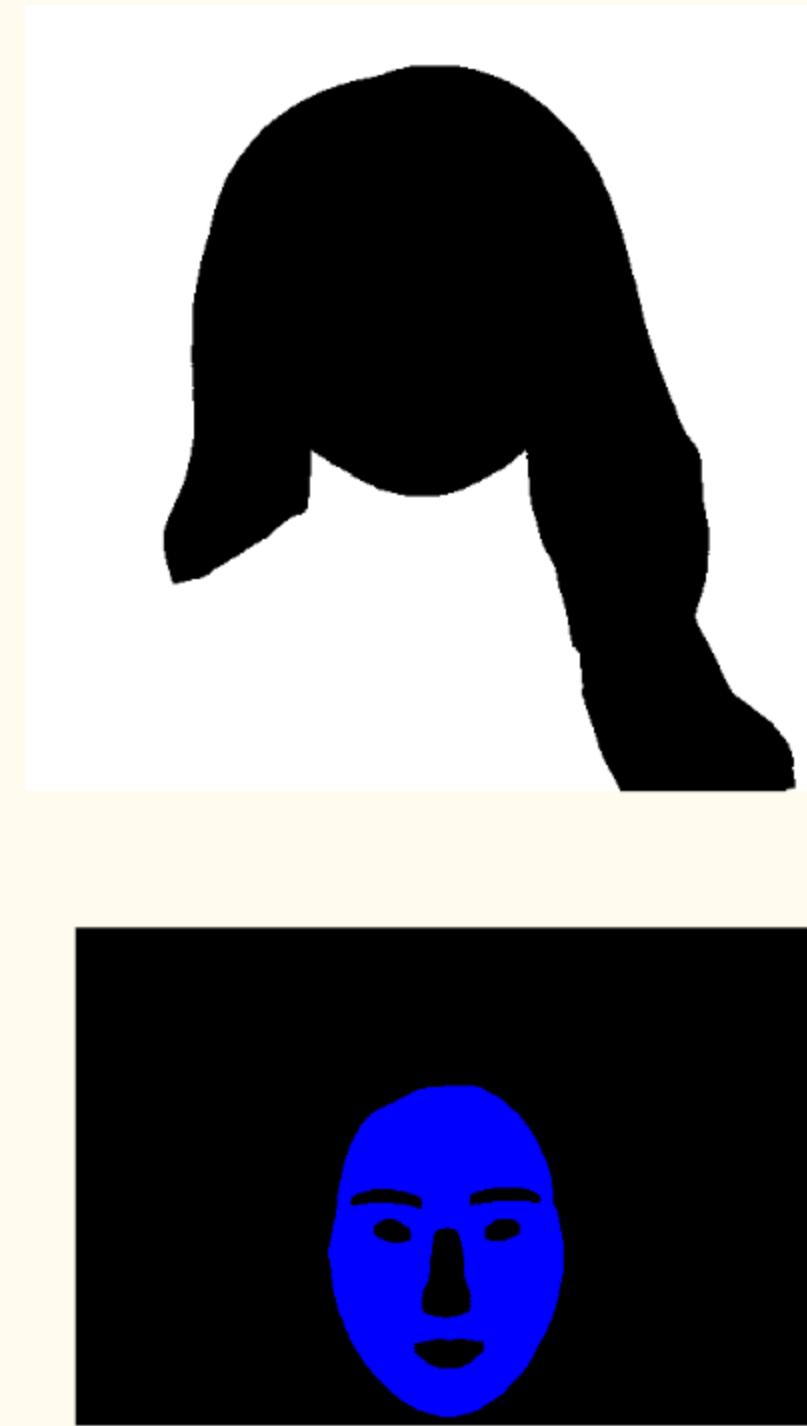
임기응변!

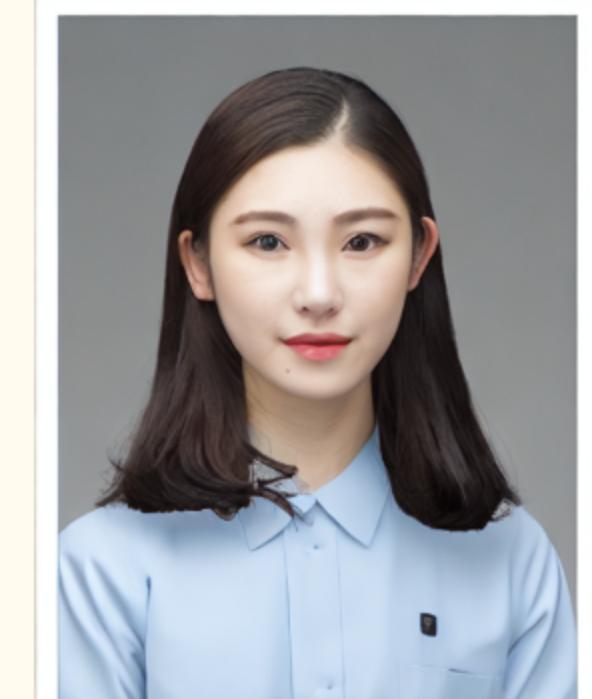
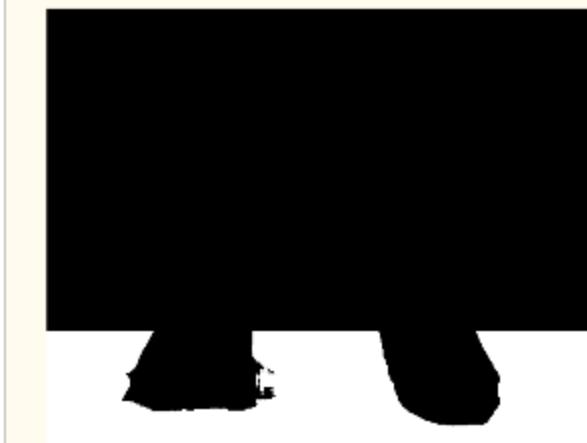
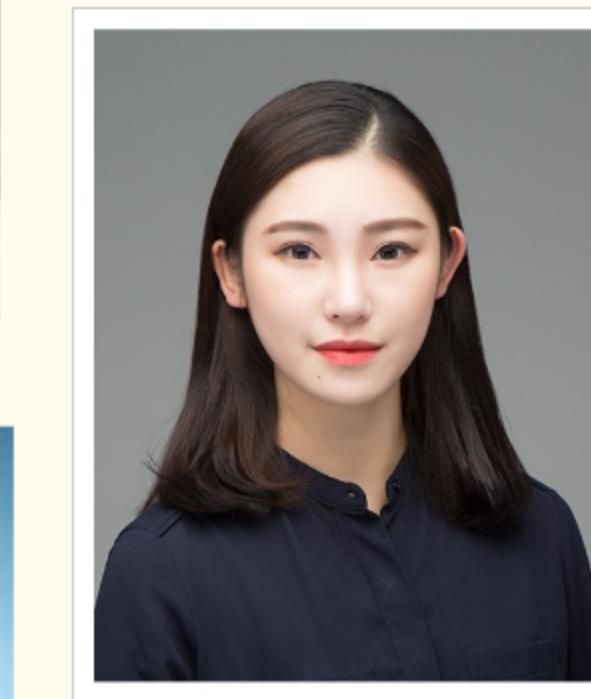
두 개의 마스크를 합치는 방식으로 문제를 해결했다.

머리가 긴 여성은 머리가 옷 위에 있을 수도 있어서

머리까지 땋는 용으로 얼굴 마스크를 사용할 수 있다.

라인마스크는 얼굴의 턱아래 부분을 기준으로 자동으로 그어질 수 있도록 설정했다. 얼굴 마스크 덕분이다.





목이 없는 마스크를  
이용하는 경우 생기는 참변



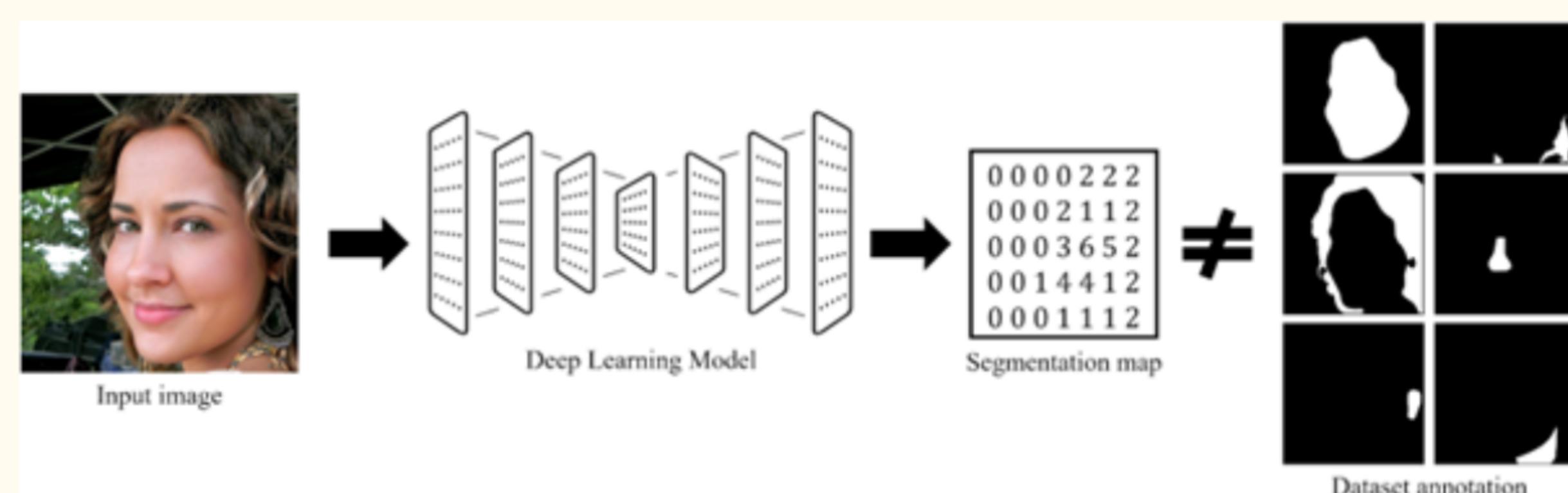
celeba mask hq



lapa



## 시멘틱 세그멘테이션의 원리



## 아쉬운 점

1. unet 튜닝을 시도해 보지 않았다.
2. 오차행렬이나 그래프와 같은 평가지표를 구현하지 않았다.

## 앞으로 해야 될 일

1. celeba mask hq로 unet 훈련 시켜보기
2. 텐서플로우 코드를 파이토치로 바꿔보기
3. unet과 face parsing0이 아닌 다른 방법으로 합성 성능 올려보기
4. paint by example 학습 시켜서 정장 퀄리티를 올려보기

Train: 18168/18168 - Valid: 2000/2000 - Test: 2000/2000

100% |██████████| 2000/2000 [09:11<00:00, 3.63it/s]

Class	F1	Jaccard
-------	----	---------

background	: 0.97273	- 0.94866
skin	: 0.94703	- 0.90372
left eyebrow	: 0.57922	- 0.43909
right eyebrow	: 0.60335	- 0.46882
left eye	: 0.56779	- 0.42782
right eye	: 0.55836	- 0.42111
nose	: 0.91294	- 0.85038
upper lip	: 0.69161	- 0.54627
inner mouth	: 0.52613	- 0.42187
lower lip	: 0.73612	- 0.60330
hair	: 0.87514	- 0.80499
Mean		
	: 0.72459	- 0.62146