분류기 뉴럴 네트워크 개발

현재까지의 진행 상황 발표

연구 연수생 이준하

환경 세팅

NVIDIA Driver 460

CUDA 11.1

Anaconda

Pytorch

Ubuntu 20.04

RTX 3090

- GPU 호환 문제
- Nvidia Driver 연결 오류
- CUDA 버전 문제

…

→ 많은 시간이 소요

데이터 구성

	TRAIN SET	VALIDATION SET	TEST SET
모자	210	24	24
외투	4984	944	889
상의	20218	4182	4176
하의	7304	2768	2735
신발	454	60	80

상의에 대한 데이터가 다른 모든 데이터의 합 보다 많다.
신발과 특히 모자에 대한 데이터가 부족하다.

데이터 전처리

Crop

Resize

265 x 256의 통일된 크기의 이미지로 전처리

학습 모델

IMAGE

JSON

CLASS

Train Set
(33170)

Validation Set
(7978)

Test Set
(7904)

Dataset

PyTorch

Alexnet

VGG 11

ResNET 50

Models

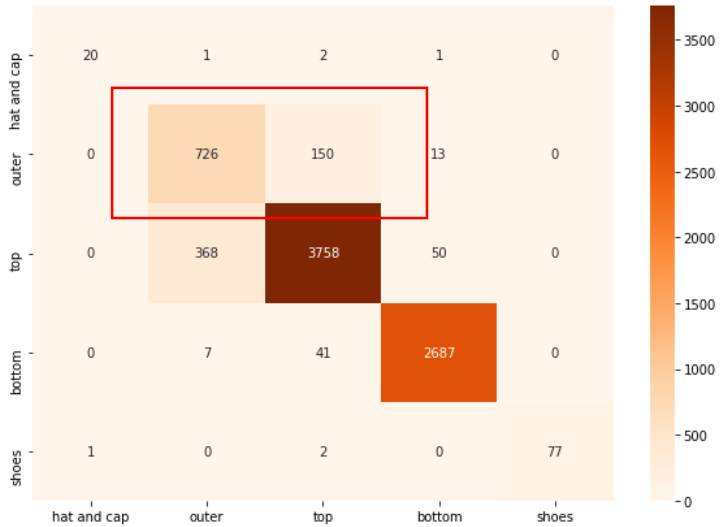
Results

Accuracy
Confusion Matrix

학습 결과

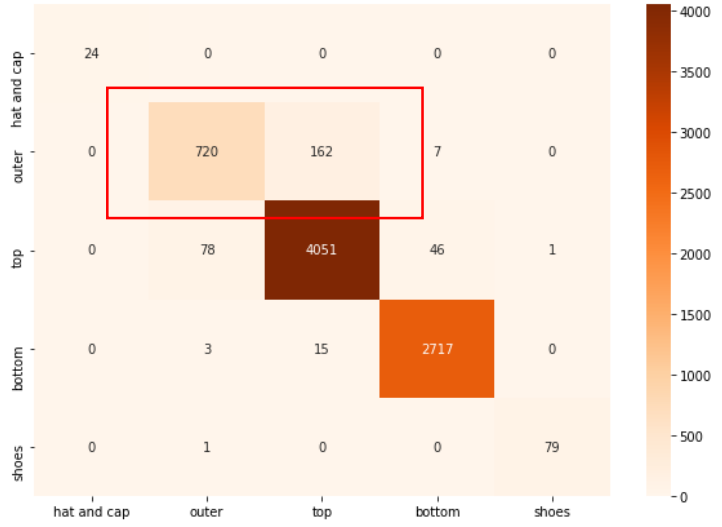
Alexnet

Accuracy of 0 : 83 %
Accuracy of 1 : 81 %
Accuracy of 2 : 89 %
Accuracy of 3 : 98 %
Accuracy of 4 : 96 %
total Accurarcy : 91 %



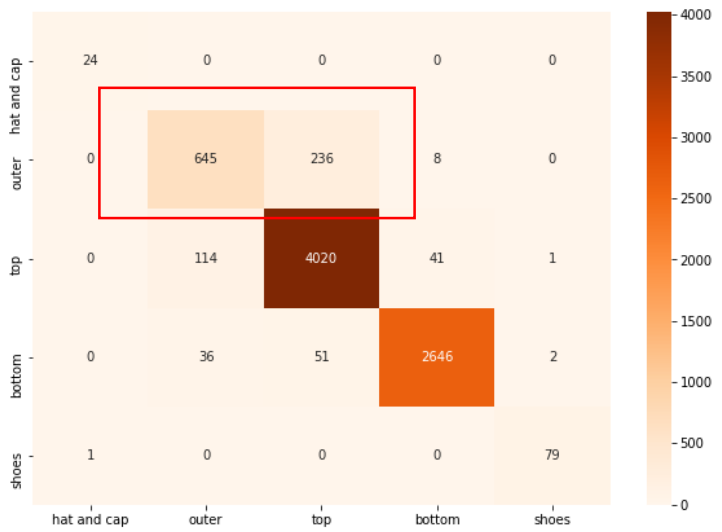
VGG 11

Accuracy of 0 : 100 %
Accuracy of 1 : 80 %
Accuracy of 2 : 97 %
Accuracy of 3 : 99 %
Accuracy of 4 : 98 %
total Accurarcy : 96 %



ResNET 50

Accuracy of 0 : 100 %
Accuracy of 1 : 72 %
Accuracy of 2 : 96 %
Accuracy of 3 : 96 %
Accuracy of 4 : 98 %
total Accurarcy : 93 %



외투에 대한 정확도가 상대적으로 많이 낮은 문제를 확인
우려했던 모자나 신발에 대해서는 좋은 정확도를 보여줌

문제점 분석

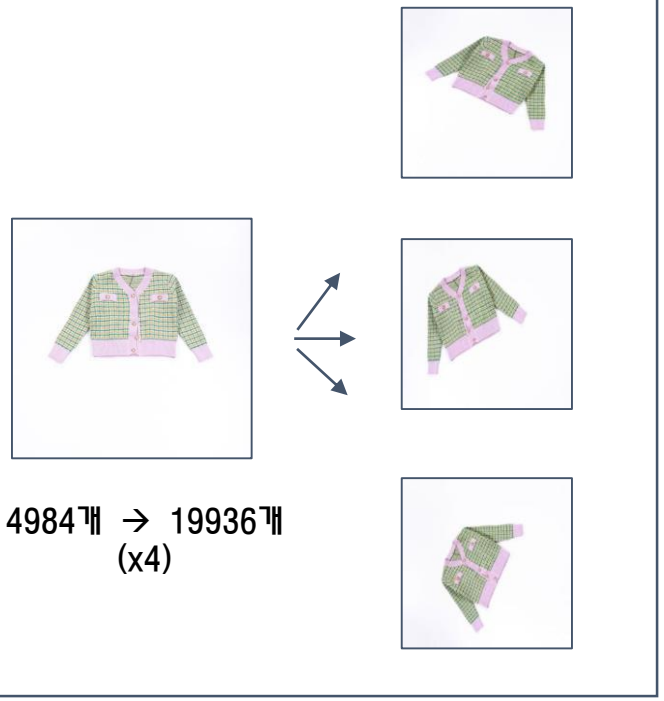
데이터 불균형?

모자 - 210
외투 - 4984
상의 - 20218
하의 - 7304
신발 - 454

TRAIN SET

외투와 상의의 모양이 유사
But, 상의에 비해 외투의 데이터가 현저히 적음.

(외투 사진에 대한)
Augmentation

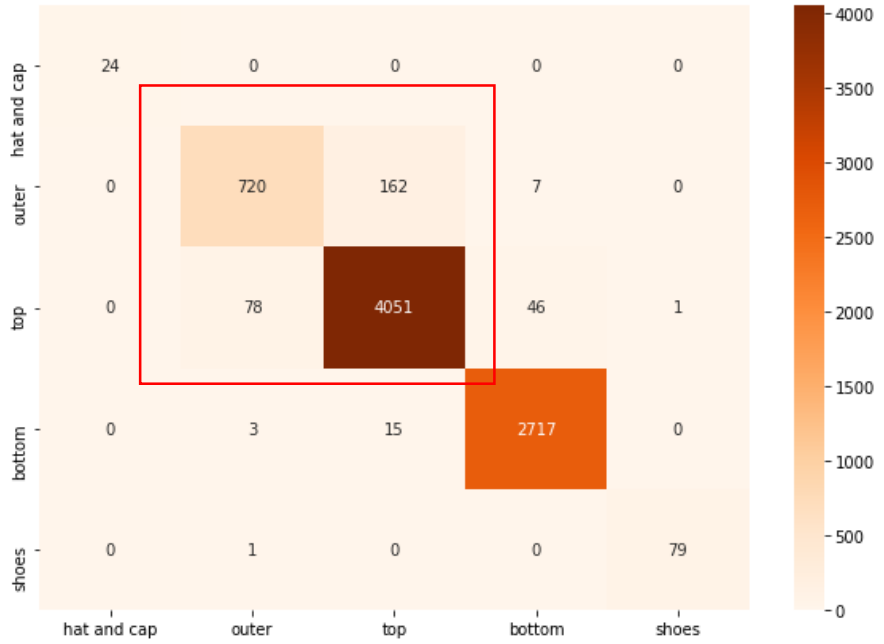


4984개 → 19936개
(x4)

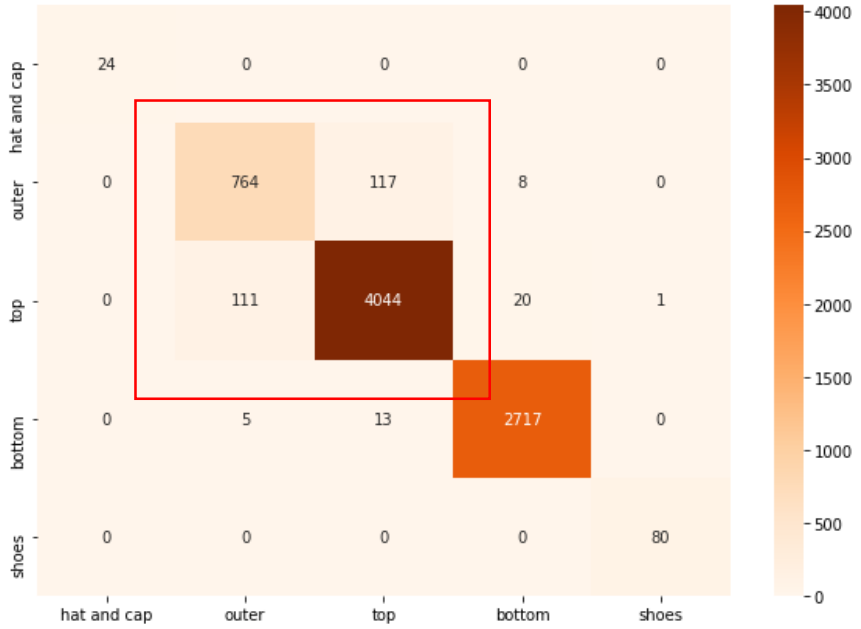
외투 사진 하나 당 3개의 랜덤 변형 이미지를 생성하여
외투 이미지를 4배로 증폭

학습 결과 - VGG

Accuracy of 0 : 100 %
Accuracy of 1 : 80 %
Accuracy of 2 : 97 %
Accuracy of 3 : 99 %
Accuracy of 4 : 98 %
total Accurarcy : 96 %



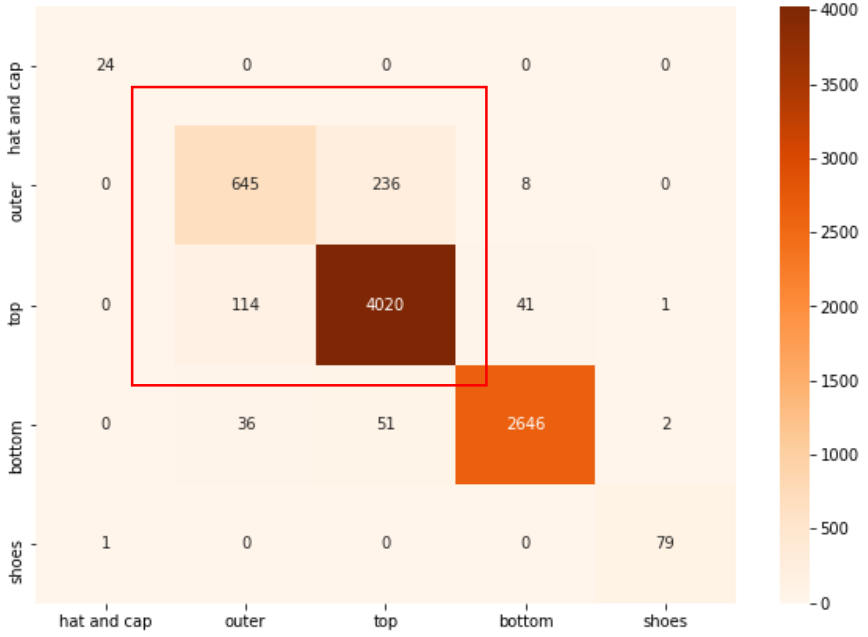
Accuracy of 0 : 100 %
Accuracy of 1 : 85 %
Accuracy of 2 : 96 %
Accuracy of 3 : 99 %
Accuracy of 4 : 100 %
total Accurarcy : 96 %



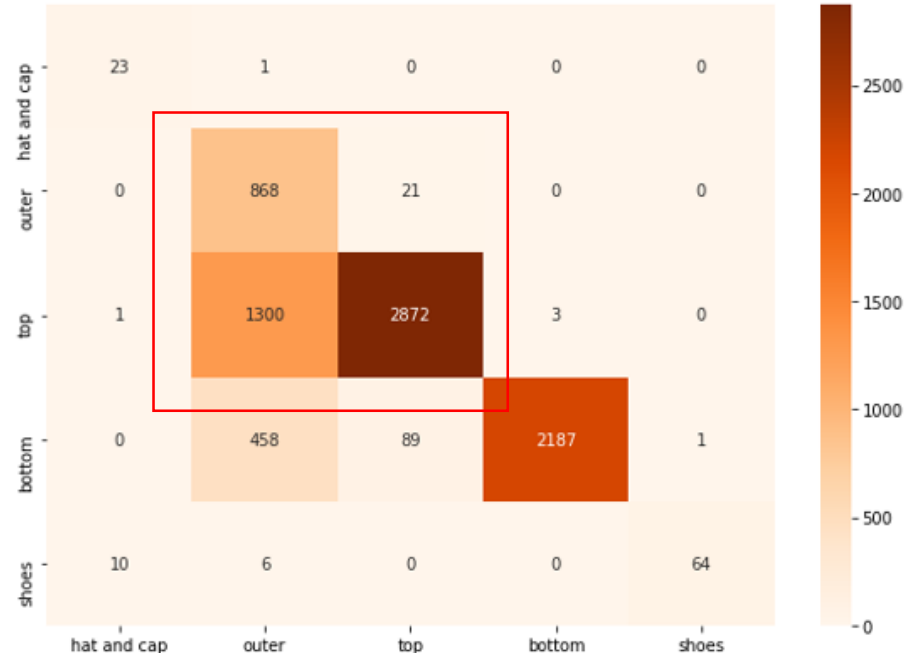
외투에 대해서는 정확도가 높아졌지만,
상의에 대한 정확도가 떨어지며 전체적인 정확도는 유사

학습 결과 - ResNET

Accuracy of 0 : 100 %
Accuracy of 1 : 72 %
Accuracy of 2 : 96 %
Accuracy of 3 : 96 %
Accuracy of 4 : 98 %
total Accurarcy : 93 %

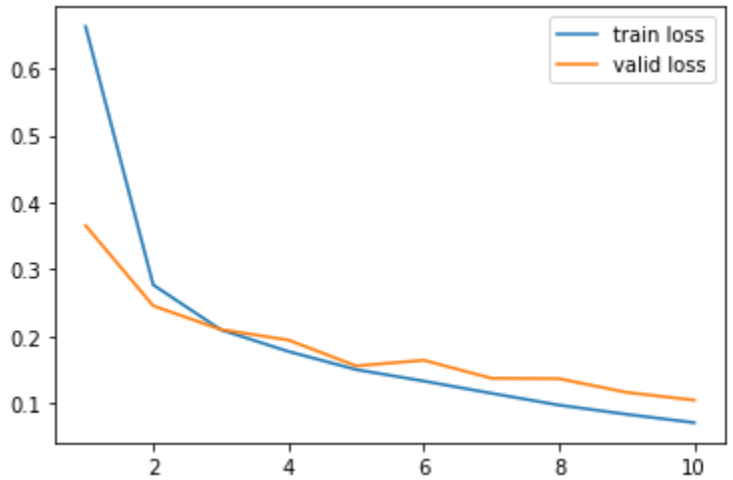


Accuracy of 0 : 95 %
Accuracy of 1 : 97 %
Accuracy of 2 : 68 %
Accuracy of 3 : 79 %
Accuracy of 4 : 80 %
total Accurarcy : 76 %



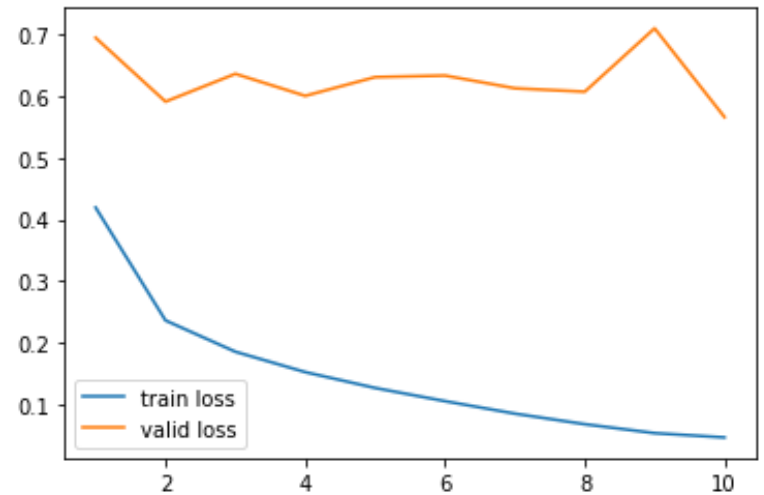
외투를 제외한 모든 클래스에 대해 학습 능력이 현저히 저하

학습 결과




```
[Epoch:    1] cost = 0.662698686   val_cost = 0.365559697
[Epoch:    2] cost = 0.276873022   val_cost = 0.245663226
[Epoch:    3] cost = 0.209794775   val_cost = 0.210123241
[Epoch:    4] cost = 0.177407265   val_cost = 0.194469556
[Epoch:    5] cost = 0.150351599   val_cost = 0.155871719
[Epoch:    6] cost = 0.13321276    val_cost = 0.164326102
[Epoch:    7] cost = 0.115010738   val_cost = 0.137452394
[Epoch:    8] cost = 0.0973432511   val_cost = 0.136791244
[Epoch:    9] cost = 0.0835831314   val_cost = 0.116319157
[Epoch:   10] cost = 0.0711068213   val_cost = 0.10472101
```

VGG




```
[Epoch:    1] cost = 0.419369876   val_cost = 0.695462823
[Epoch:    2] cost = 0.235829934   val_cost = 0.591690779
[Epoch:    3] cost = 0.184946626   val_cost = 0.636839032
[Epoch:    4] cost = 0.151817381   val_cost = 0.601049602
[Epoch:    5] cost = 0.126046717   val_cost = 0.63150835
[Epoch:    6] cost = 0.104587086   val_cost = 0.634062231
[Epoch:    7] cost = 0.0842757523   val_cost = 0.613339603
[Epoch:    8] cost = 0.0671239421   val_cost = 0.607762456
[Epoch:    9] cost = 0.0525799319   val_cost = 0.710835338
[Epoch:   10] cost = 0.0455614515   val_cost = 0.566559911
```

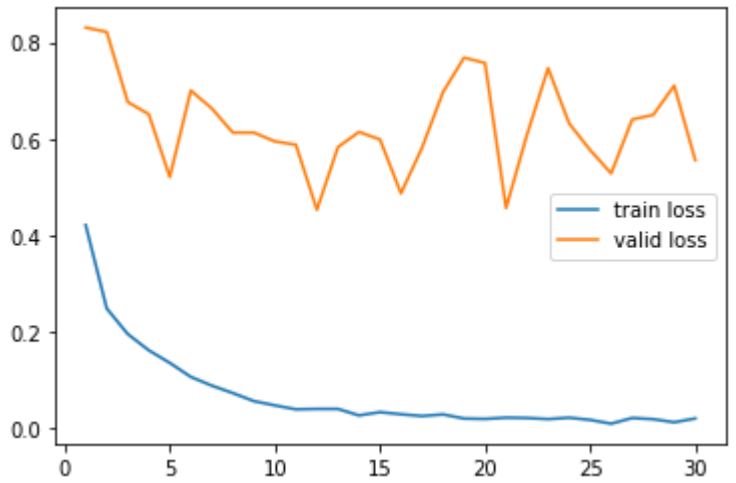
ResNET

ResNET의 학습에 문제가 있다는 것을 확인할 수 있다.

학습 결과 - ResNET

```
[Epoch:    1] loss = 0.420798957   test_loss = 0.830705583
[Epoch:    2] loss = 0.248019651   test_loss = 0.821651518
[Epoch:    3] loss = 0.194884866   test_loss = 0.676226616
[Epoch:    4] loss = 0.161187693   test_loss = 0.651022792
[Epoch:    5] loss = 0.135068625   test_loss = 0.520698607
[Epoch:    6] loss = 0.105999425   test_loss = 0.700125813
[Epoch:    7] loss = 0.0878870562   test_loss = 0.662937522
[Epoch:    8] loss = 0.072444886   test_loss = 0.612808228
[Epoch:    9] loss = 0.0557646453   test_loss = 0.612856984
[Epoch:   10] loss = 0.0468775854   test_loss = 0.594606936
[Epoch:   11] loss = 0.0387507454   test_loss = 0.587622166
[Epoch:   12] loss = 0.0395479612   test_loss = 0.452660263
[Epoch:   13] loss = 0.0294592734   test_loss = 0.582722604
[Epoch:   14] loss = 0.0261070859   test_loss = 0.61416012
[Epoch:   15] loss = 0.0328119993   test_loss = 0.598638117
[Epoch:   16] loss = 0.0284867566   test_loss = 0.486726046
[Epoch:   17] loss = 0.0247973613   test_loss = 0.581273973
[Epoch:   18] loss = 0.0281678084   test_loss = 0.696889758
[Epoch:   19] loss = 0.0195961613   test_loss = 0.768323243
[Epoch:   20] loss = 0.0188036878   test_loss = 0.757040501
[Epoch:   21] loss = 0.0213798359   test_loss = 0.456346661
[Epoch:   22] loss = 0.0206992142   test_loss = 0.609123886
[Epoch:   23] loss = 0.0184364952   test_loss = 0.74662441
[Epoch:   24] loss = 0.0213121176   test_loss = 0.631869495
[Epoch:   25] loss = 0.0167859234   test_loss = 0.576630235
[Epoch:   26] loss = 0.00892569683   test_loss = 0.528166473
[Epoch:   27] loss = 0.0207189675   test_loss = 0.640037358
[Epoch:   28] loss = 0.0182672907   test_loss = 0.649531007
[Epoch:   29] loss = 0.0118503841   test_loss = 0.710258007
[Epoch:   30] loss = 0.0197254531   test_loss = 0.555757165
```

Epoch를 30까지 늘려도 효과가 없다.

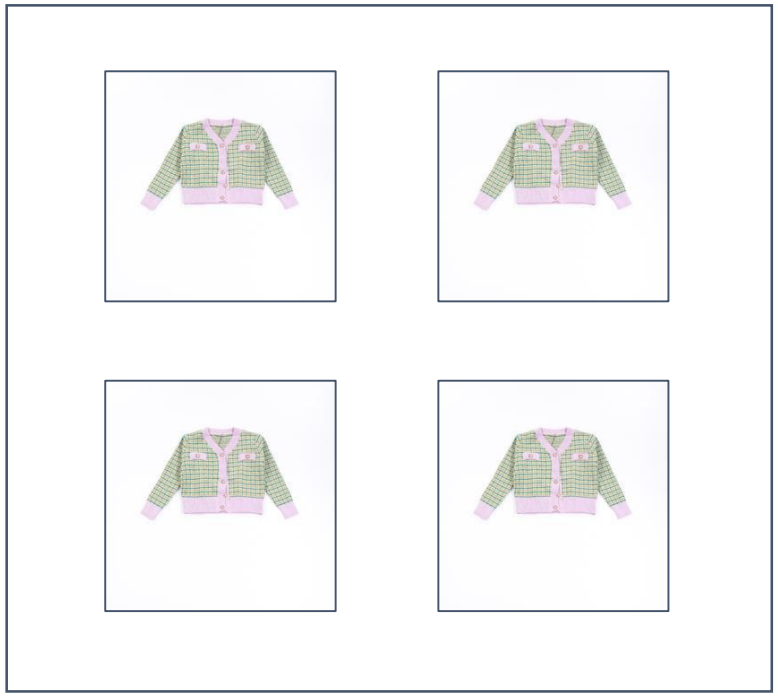


→ Overfitting

문제점 분석

- 이미지 변형을 이용한 이미지 증폭으로는 원하는 결과를 얻어내지 못하였음.
  - 어떤 방법으로 데이터셋을 변형할 것인가?
- 이미지 수가 늘어나며 ResNET에 Overfitting 문제가 발생
  - Overfitting 문제를 어떻게 해결할 것인가?

이미지 증폭 방법 변경

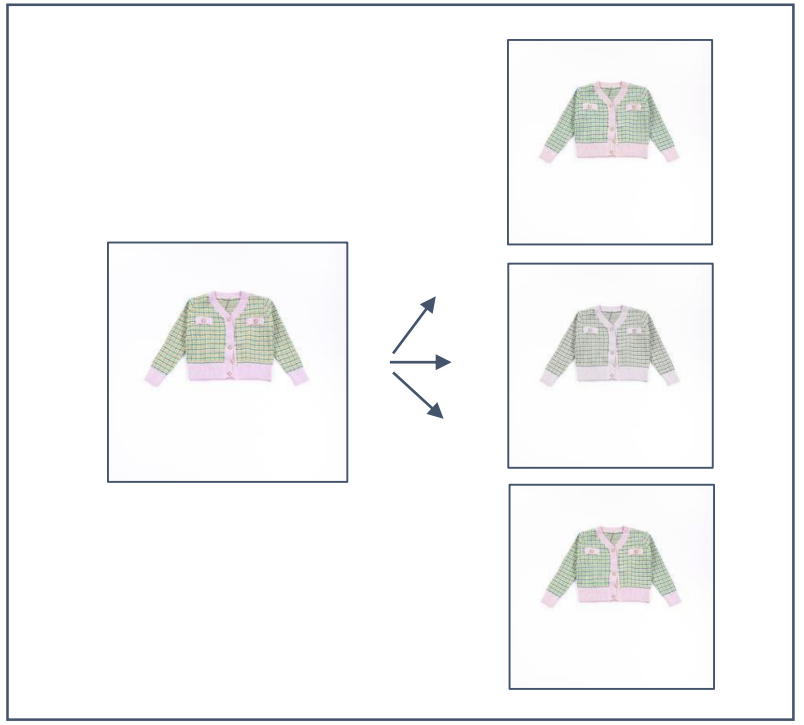


외투 이미지에 대하여

1. 하나의 이미지를 원본 그대로 4개로 복제

- Test Set에 뒤틀리거나 회전된 이미지가 없다는 문제점 개선

색상이 다양하게 변한 예시



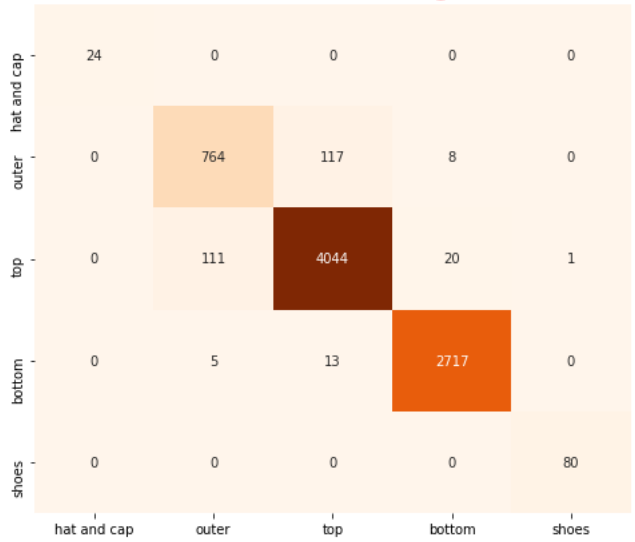
외투 이미지에 대하여

2. 채도, 색조 변경을 이용해 색상 변형 데이터 생성

- 데이터의 색상을 변형시키면 더 좋은 데이터셋이 생길 것이라고 판단

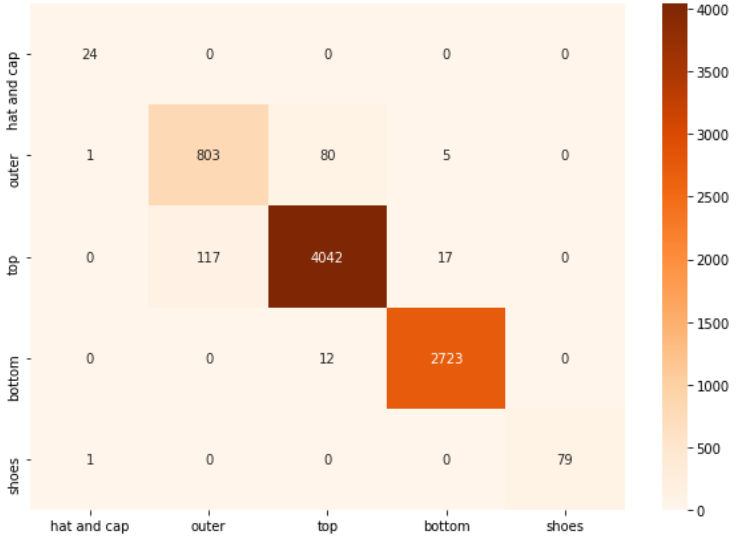
학습 결과 - VGG

Accuracy of 0 : 100 %
Accuracy of 1 : 85 %
Accuracy of 2 : 96 %
Accuracy of 3 : 99 %
Accuracy of 4 : 100 %
total Accurarcy : 96 %



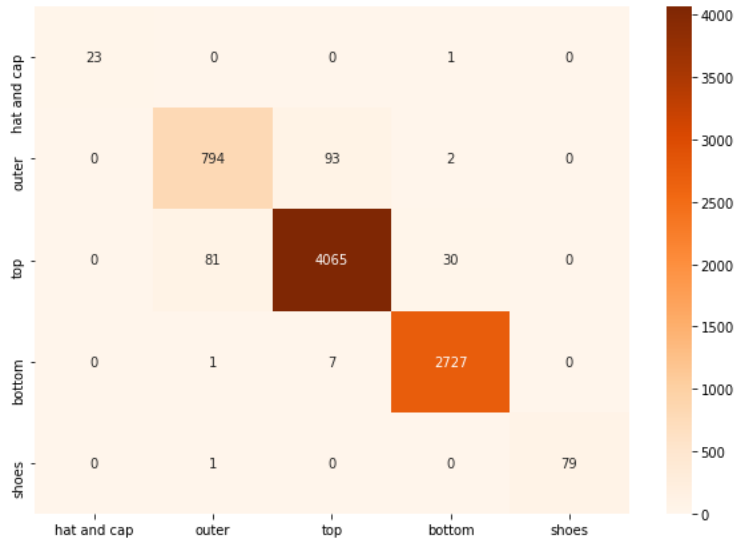
기존의 이미지 변형

Accuracy of 0 : 100 %
Accuracy of 1 : 90 %
Accuracy of 2 : 96 %
Accuracy of 3 : 99 %
Accuracy of 4 : 98 %
total Accurarcy : 97 %



원본 이미지 증폭

Accuracy of 0 : 95 %
Accuracy of 1 : 89 %
Accuracy of 2 : 97 %
Accuracy of 3 : 99 %
Accuracy of 4 : 98 %
total Accurarcy : 97 %

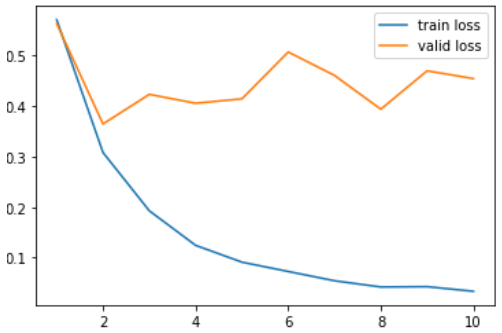


색상 변형

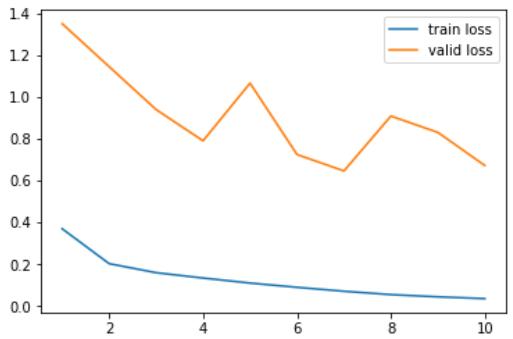
두 방법 모두 비슷한 수준의 성능 향상을 확인할 수 있었다.
(색상 변형이 미세하게 더 성능이 좋음)

ResNET Overfitting 해결

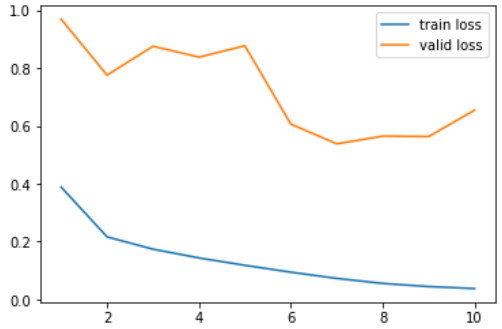
1. 기존의 50층 모델에서 층의 수를 각각 34, 18층으로 줄여서 학습 진행



ResNET 50



ResNET 34



ResNET 18

여전히 Overfitting 문제가 발생하는 것을 확인할 수 있다. → 층수와는 상관이 없다

ResNET Overfitting 해결

2. 훈련 집합 전체에 대한 데이터를 색상 변형을 이용해 4배로 증폭

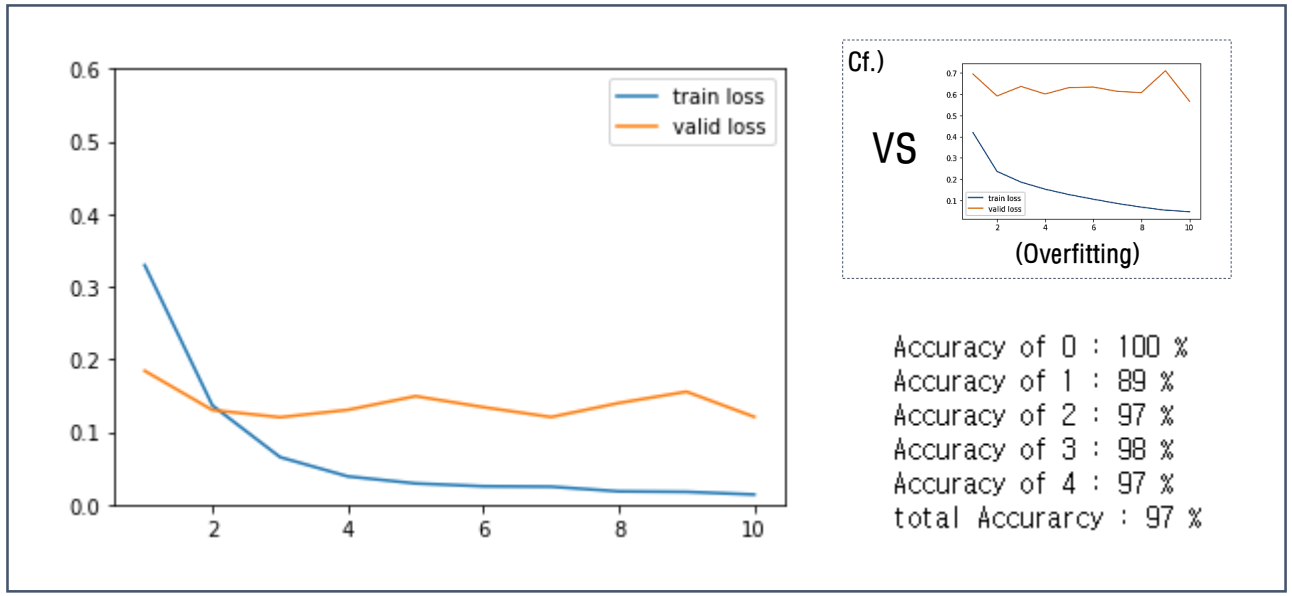
VGG에서의 실험 결과 색상 변형 데이터가 비교적 조금이나마 성능이 좋기 때문

외투의 데이터를 개별적으로 증폭시키는 것은 큰 효과가 없음

모자 - 840
외투 - 19936
상의 - 80872
하의 - 29216
신발 - 1816

TRAIN SET

ResNET 50

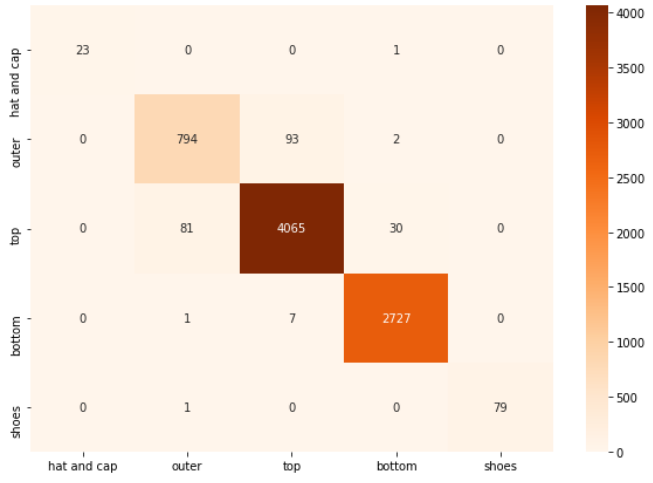


Overfitting 이 해결

증폭된 훈련 데이터셋 적용

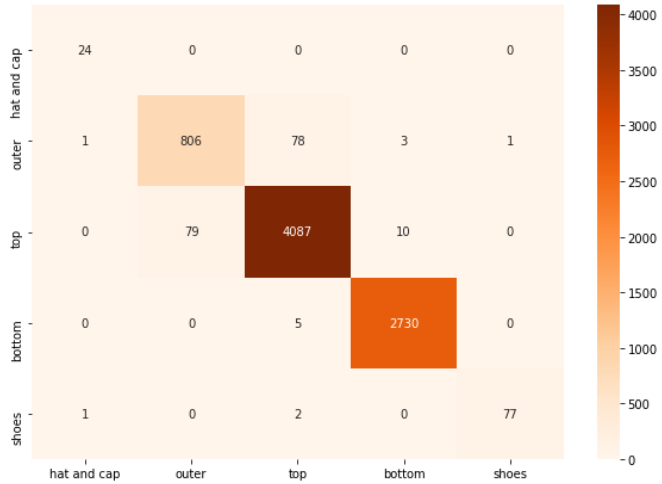
앞서 4배로 증폭시킨 훈련 집합을 VGG에도 적용

Accuracy of 0 : 95 %
Accuracy of 1 : 89 %
Accuracy of 2 : 97 %
Accuracy of 3 : 99 %
Accuracy of 4 : 98 %
total Accurarcy : 97 %



외투 데이터만 색상 변형 및 증폭

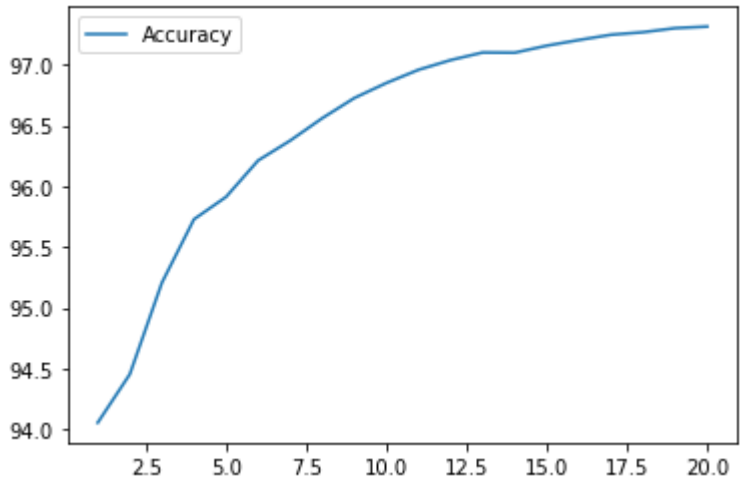
Accuracy of 0 : 100 %
Accuracy of 1 : 90 %
Accuracy of 2 : 97 %
Accuracy of 3 : 99 %
Accuracy of 4 : 96 %
total Accurarcy : 97 %



모든 데이터 색상 변형 및 증폭

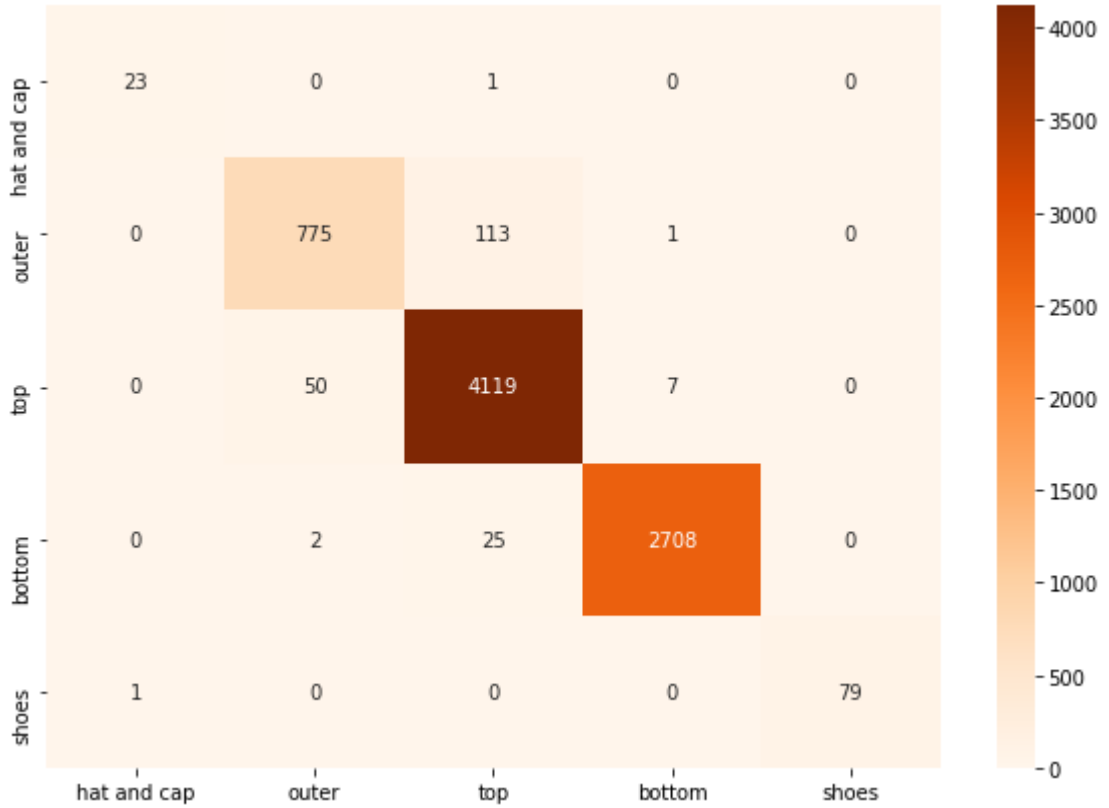
근소하게 성능이 향상

모든 데이터 색상 변형 및 증폭 * 20 epochs

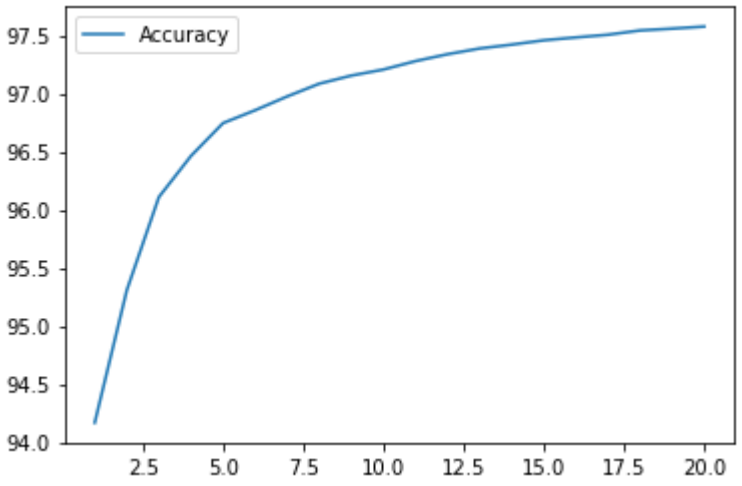



```
Accuracy of 0 : 95 %
Accuracy of 1 : 87 %
Accuracy of 2 : 98 %
Accuracy of 3 : 99 %
Accuracy of 4 : 98 %
total Accuracy : 97 %

Num  of total : 7704 / 7904
Num of 0 : 23 / 24
Num of 1 : 775 / 889
Num of 2 : 4119 / 4176
Num of 3 : 2708 / 2735
Num of 4 : 79 / 80
```

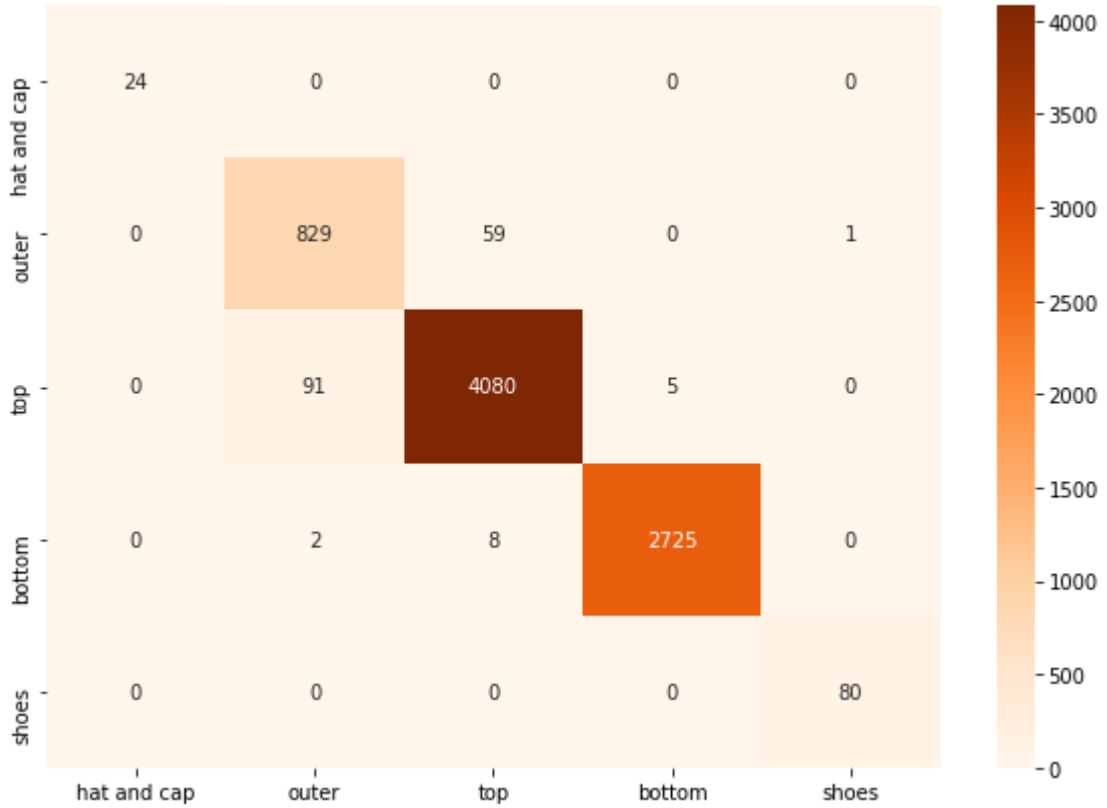


(모든 데이터 색상 변형 및 증폭 + 외투 추가 증폭) * 20 epochs
[모든 데이터 * 4 + 외투 데이터만 변형된 그대로 * 4]

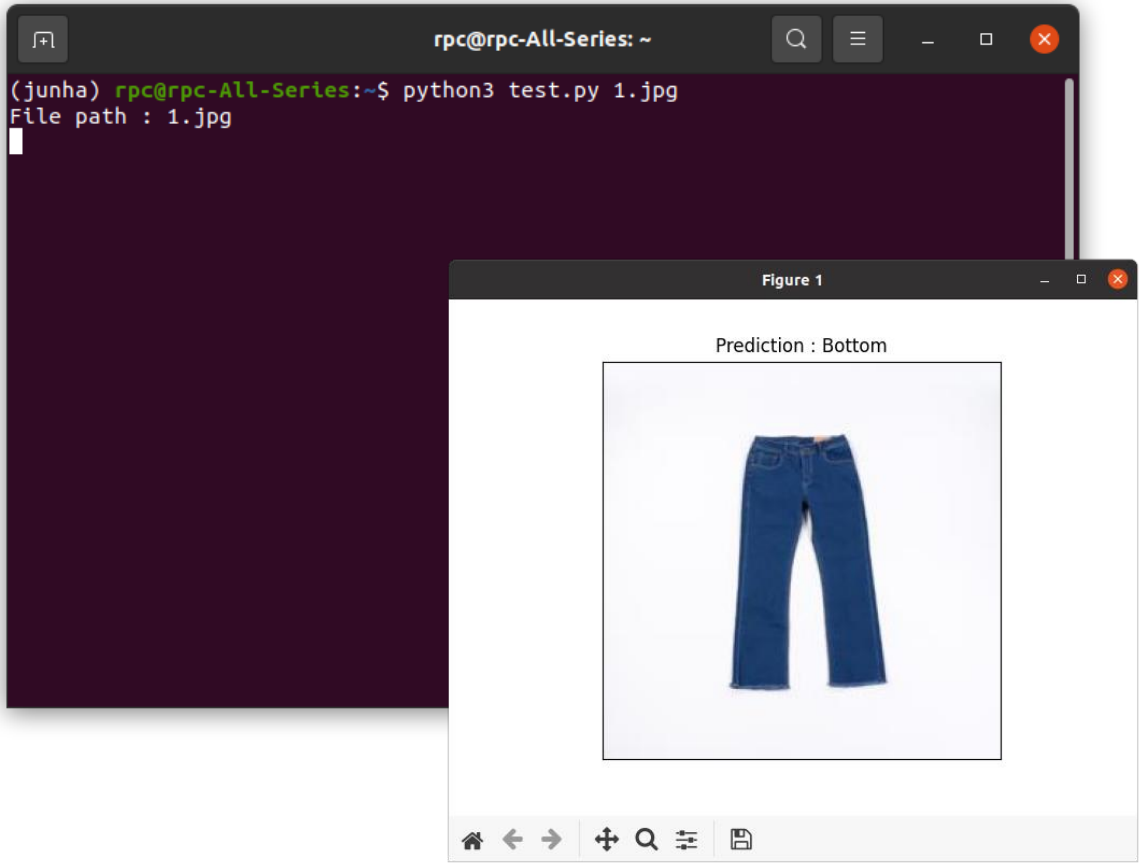


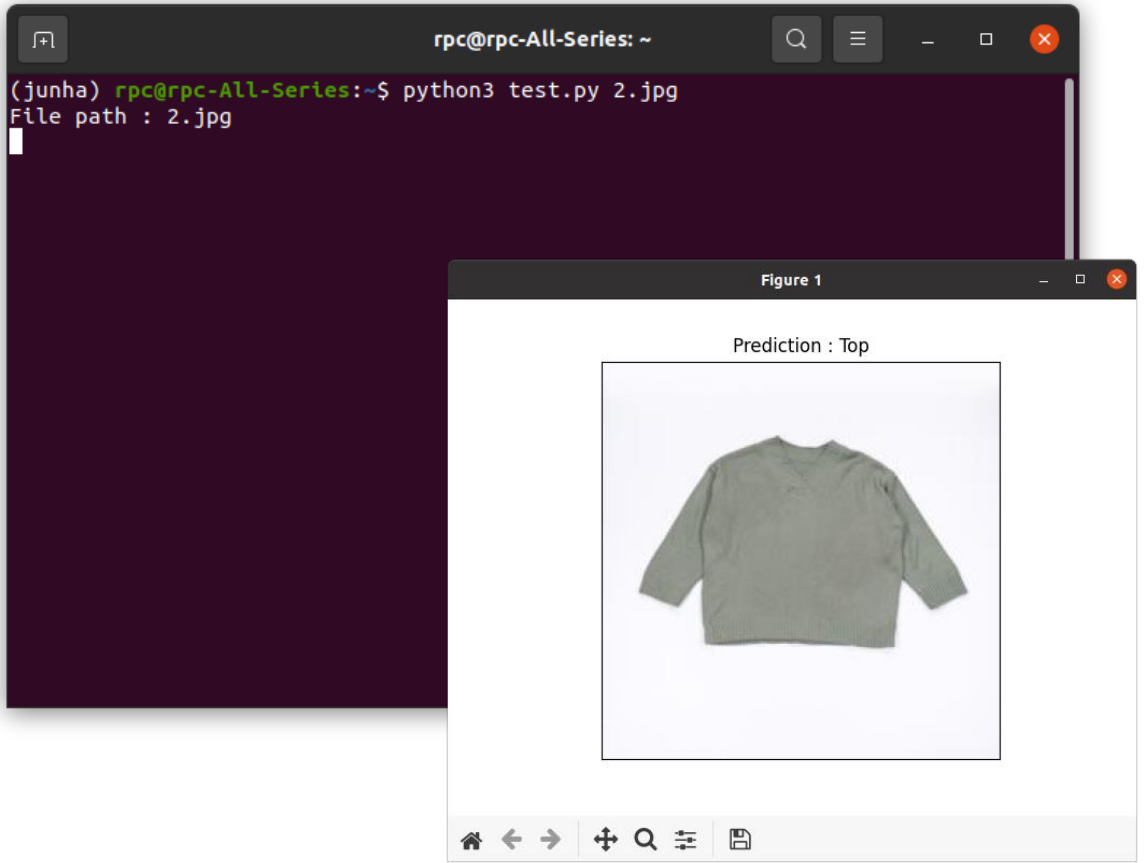
Accuracy of 0 : 100 %
Accuracy of 1 : 93 %
Accuracy of 2 : 97 %
Accuracy of 3 : 99 %
Accuracy of 4 : 100 %
total Accuracy : 97 %

Num of total : 7738 / 7904
Num of 0 : 24 / 24
Num of 1 : 829 / 889
Num of 2 : 4080 / 4176
Num of 3 : 2725 / 2735
Num of 4 : 80 / 80

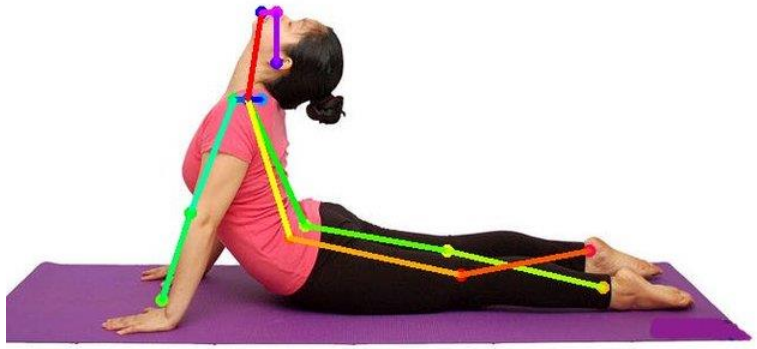


결과물


rpc@rpc-All-Series: ~
(junha) rpc@rpc-All-Series:~$ python3 test.py 1.jpg
File path : 1.jpg
Figure 1
Prediction : Bottom


rpc@rpc-All-Series: ~
(junha) rpc@rpc-All-Series:~$ python3 test.py 2.jpg
File path : 2.jpg
Figure 1
Prediction : Top

NEXT



Pose Estimation