EndAnd DataTeam

중고 명품 이미지 분류 및 검색 서비스

index

회사 소개

ENDAND

가입 및 제품 등록

전문가 제품 인증

자유롭게 판매

프로젝트 소개

제품의 시리얼 넘버를 선택해 주세요.

S22675

M22675

I15623

L59813

E59813

이미지에 근접한 모델명 추출 및 5순위까지 알려주는 서비스

프로젝트 소개

데이터 상황

프로젝트 소개

데이터 전처리

구글 크롤링

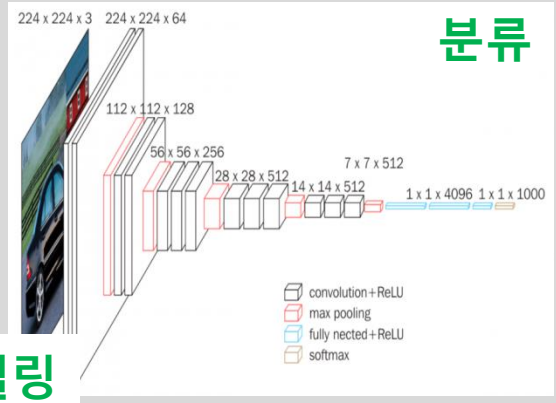
Object detectioin

Image Hash 256

중복 이미지 제거

모델링

분류



벡터 기반

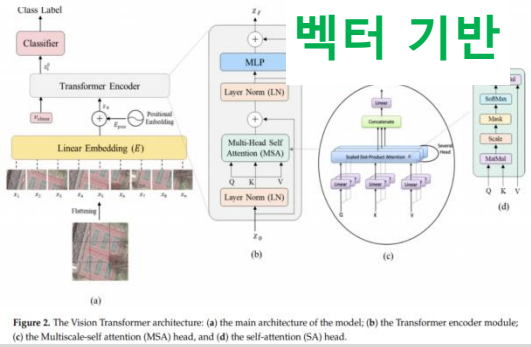


Figure 2. The Vision Transformer architecture: (a) the main architecture of the model; (b) the Transformer encoder module; (c) the Multiscale-self attention (MSA) head, and (d) the self-attention (SA) head.

Classifier

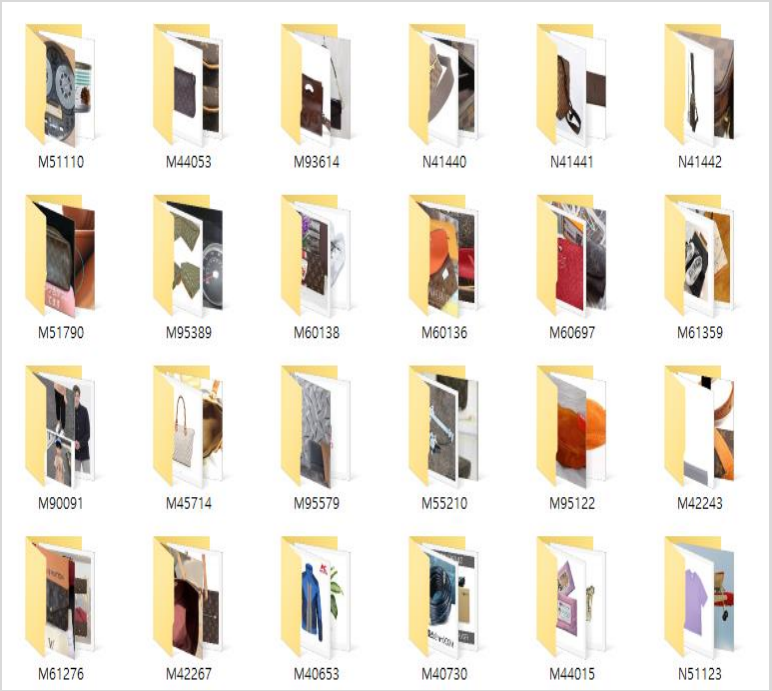
데이터 크롤링 및 전처리

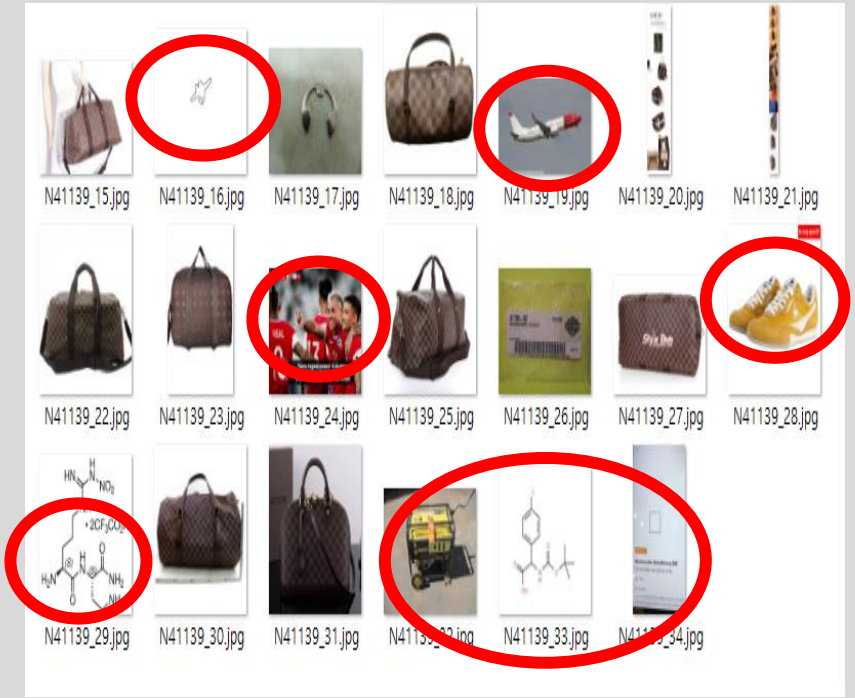
Selenium

Docker
& CVAT

YOLOv3
& Darknet

데이터 크롤링 및 전처리

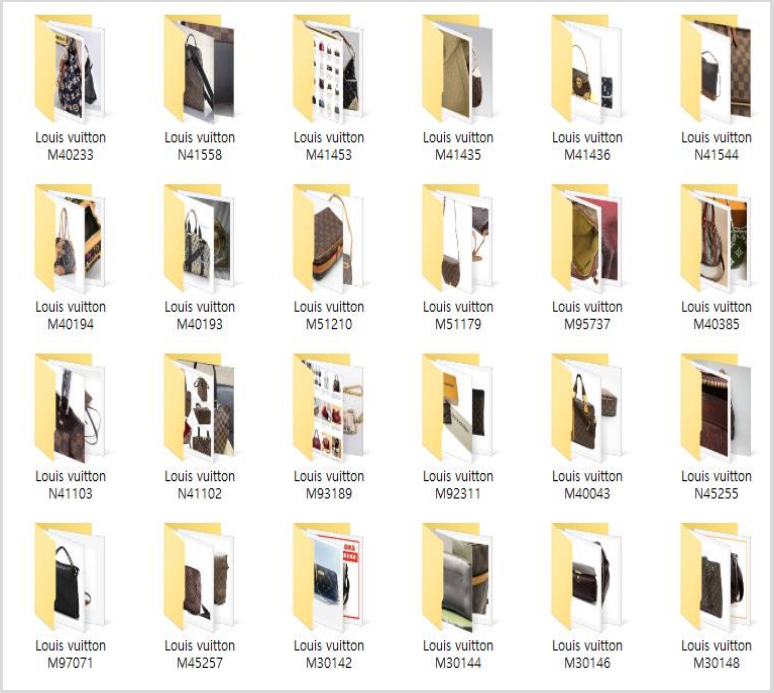


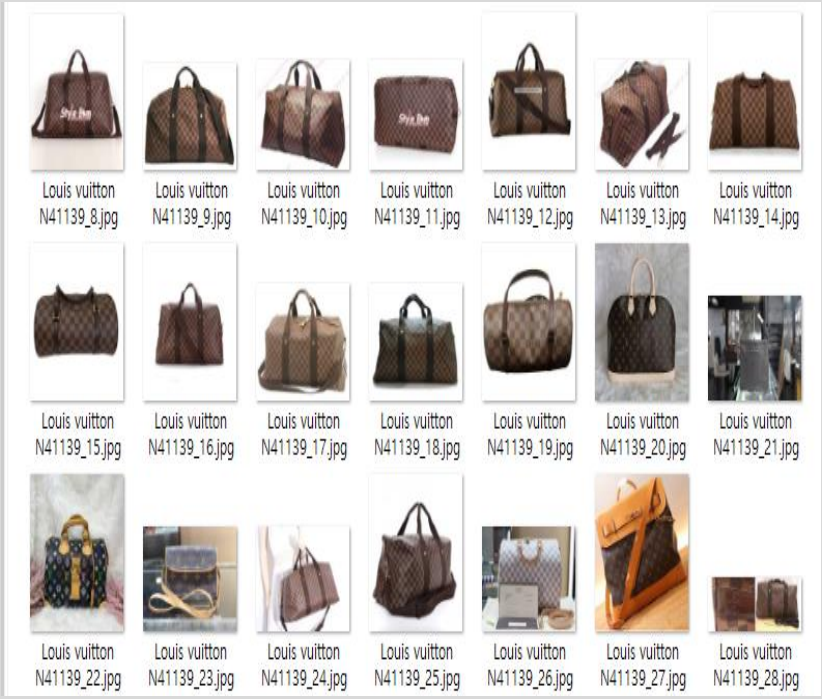


1차 크롤링 데이터 확인

Garbage data 발견

데이터 크롤링 및 전처리

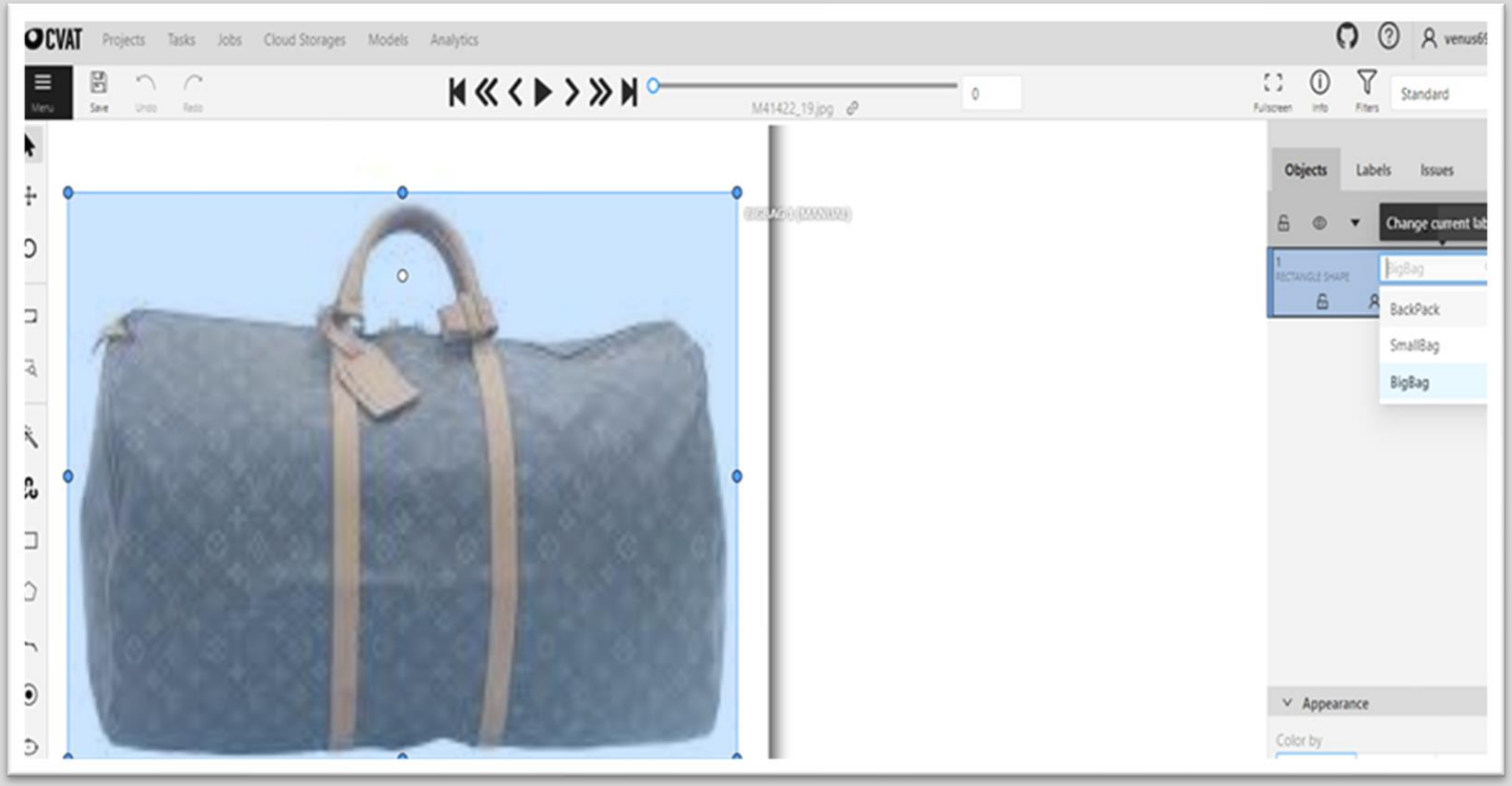




2차 크롤링 데이터

사용가능한 데이터로 정제

데이터 크롤링 및 전처리

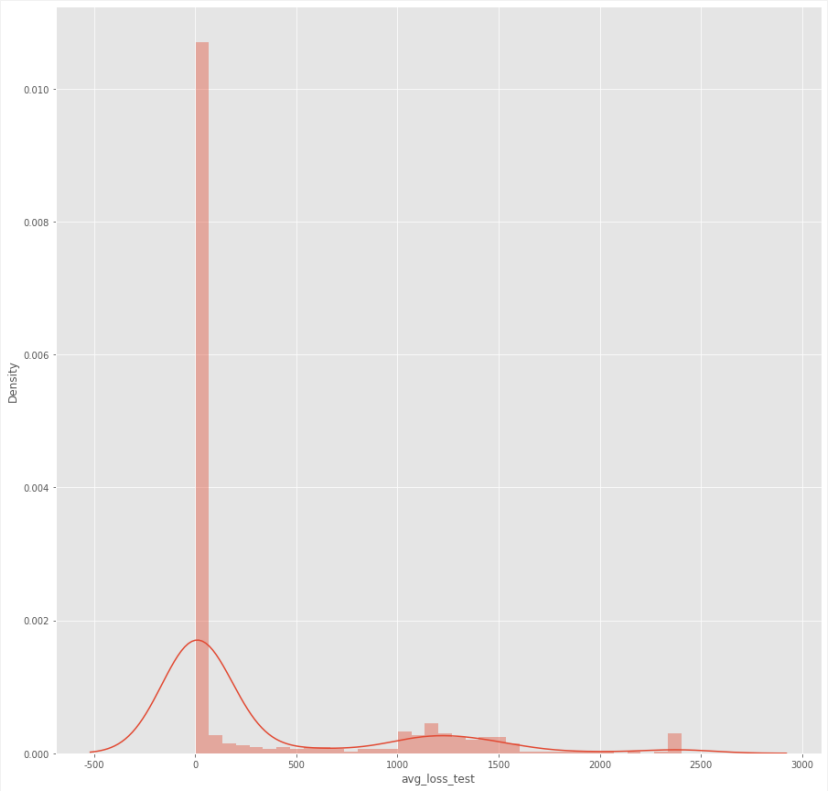

CVAT
Projects
Tasks
Jobs
Cloud Storages
Models
Analytics
Menu
Save
Undo
Redo
M41422_19.jpg
Fullscreen
Info
Filters
Standard
Objects
Labels
Issues
Change current lab
BackPack
SmallBag
BigBag
Appearance
Color by

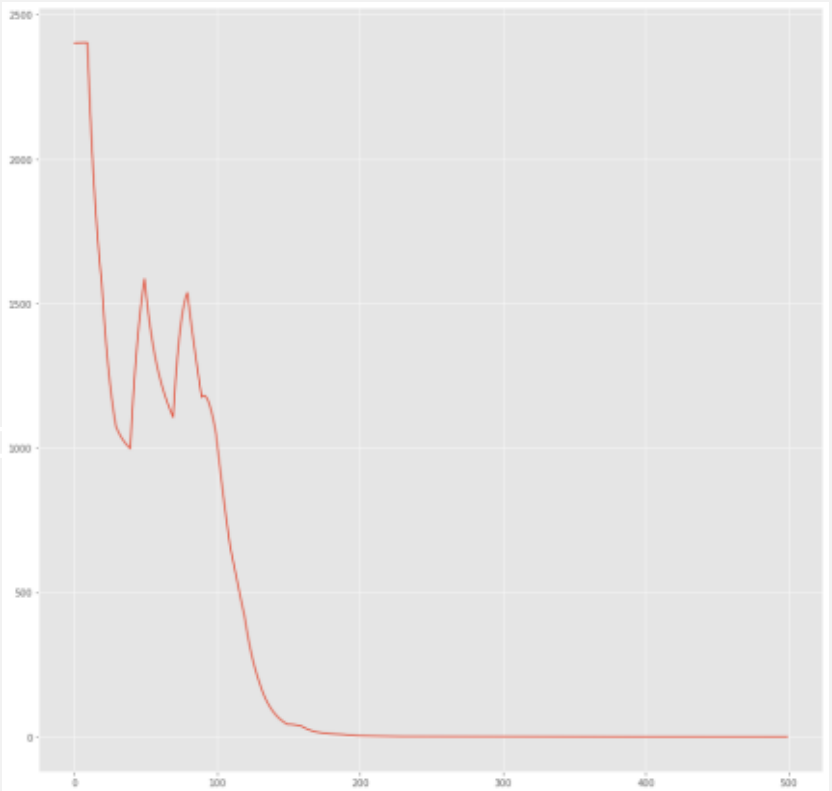
데이터 크롤링 및 전처리

인덱스설정

```
[ ]  1 idx2class = {0:"Bag",1:"SmallBag",2:"BackPack"}  => idx2class = {0:"Bag"}
     2
```

데이터 크롤링 및 전처리





Loss 값 안정화 확인

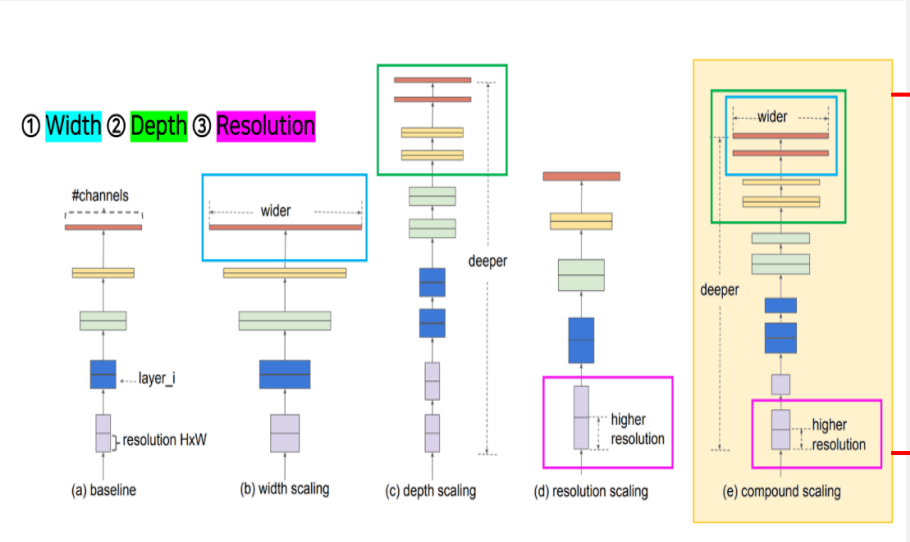
데이터 크롤링 및 전처리


```
img = cv2.imread(IMAGE_PATH)

img = cv2.imread(IMAGE_PATH,cv2.IMREAD_COLOR)
boxes = detect.detect_image(model, img)

color = (255,0,0)
# print(boxes)
if len(boxes) !=0:
  cv2.rectangle(img,(boxes[0][0],boxes[0][1]),(boxes[0][2],boxes[0][3]),color,3)



  text=idx2class[boxes[0][-1]]
  org=(boxes[0][1],int(boxes[0][4]+100))
  font=cv2.FONT_HERSHEY_SIMPLEX
  cv2.putText(img,text,org,font,1,(255,0,0),2)
  cv2_imshow(img)
```

Classification 모델 적용

EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks

[Submitted on 28 May 2019 (v1), last revised 11 Sep 2020 (this version, v5)]



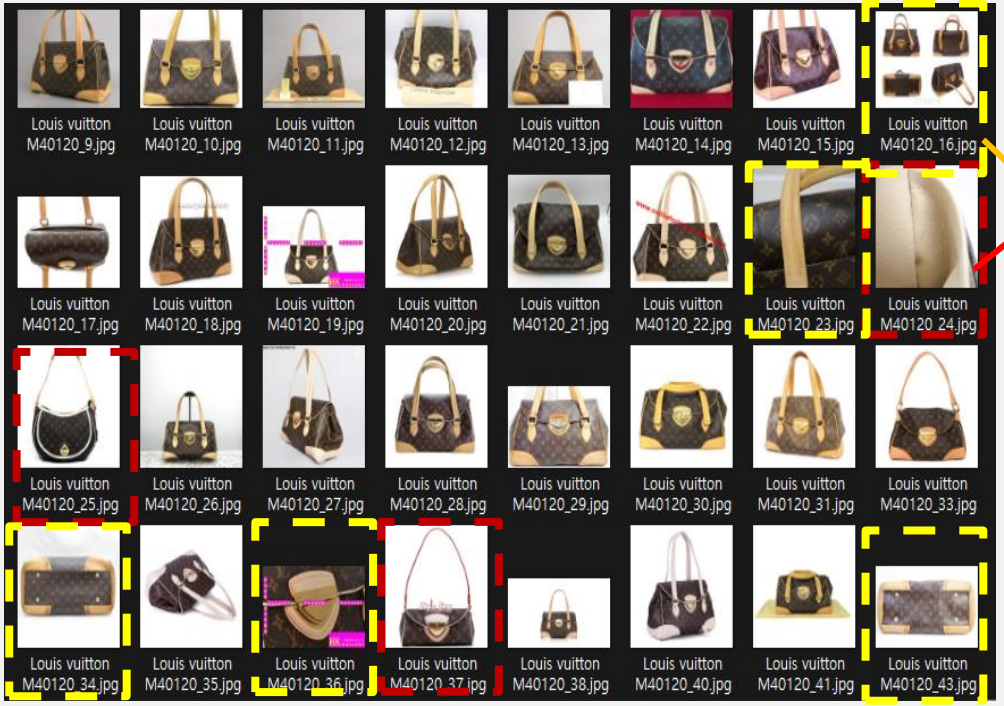
$$\text{depth: } d = \alpha^\phi$$
$$\text{width: } w = \beta^\phi$$
$$\text{resolution: } r = \gamma^\phi$$
$$\text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$
$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

모델의 정확도를 높일 때, 일반적으로 **(1) 모델의 깊이, (2) 너비, (3) 입력 이미지의 크기(해상도)**를 조절

EfficientNet은 3가지를 효율적으로 조절할 수 있는 **Compound Scaling** 방법

한정된 자원을 갖고 있는 상황에서 Depth, Width, Resolution을 적절히 조절하여 모델의 크기와 연산량을 줄이면서도 성능은 높일 수 있는 모델을 채택

Classification 모델 적용



- 첫번째 전처리

제품과 관련 없는 이미지 제거

- 두번째 전처리

제품의 일부 이미지 제거

노이즈가 있는 이미지 제거

Classification **모델 적용**

Dataset				
제품 수	Class_81	Class_286	Class_480	Class_829
제품 당 이미지 개수	50	50	50	10
총 개수	4,050	14,300	24,000	8,290

제품 수 81, 286, 480 데이터의 경우 **첫번째 전처리** 후 진행

829 데이터의 경우 **두번째 전처리** 후 좋은 데이터 중 **10장씩만 정제** 학습

제품의 수를 점차 늘려가며 모델 학습을 시행

데이터가 적기 때문에 Augmentation 적극 활용

Augmentation

고객이 제품을 업로드 하는 방식에 맞추어 활용

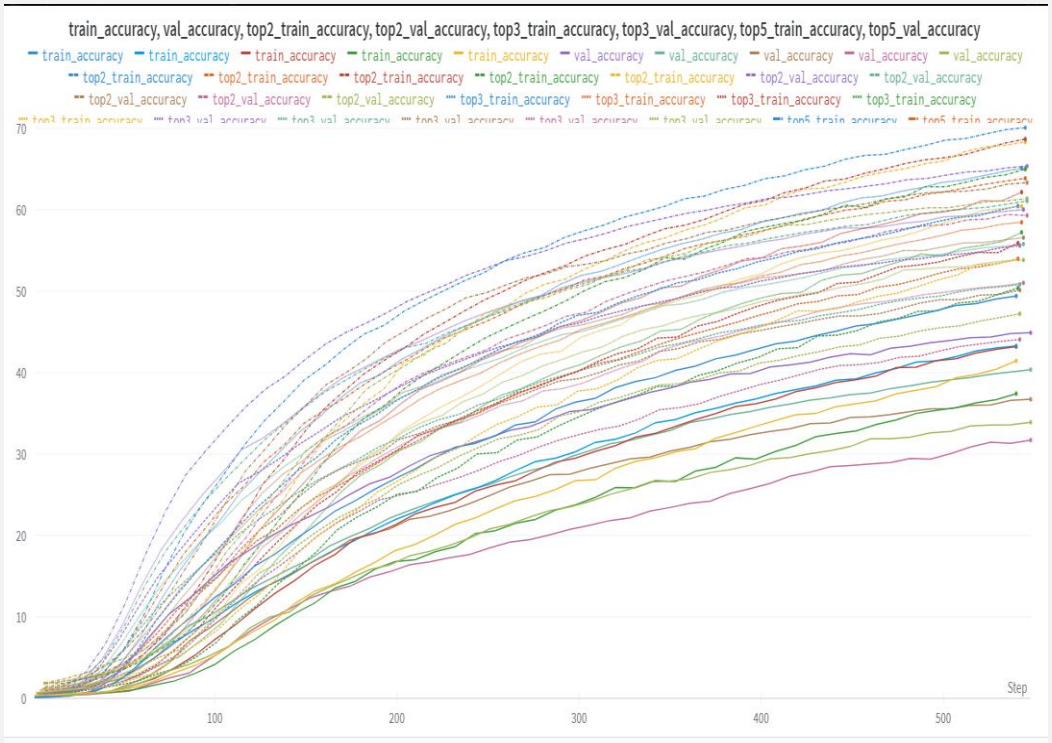
RandomCrop, **RandomBrightnessContrast** 만 적합

Classification 모델 적용

aug_class844_using10img_efficientnet-b0	finished	efficientnet-b0
aug_class844_using10img_efficientnet-b0	finished	efficientnet-b0
aug_class844_using10img_efficientnet-b3	finished	efficientnet-b3
aug_class844_using10img_efficientnet-b3	crashed	efficientnet-b3
aug_class844_using10img_efficientnet-b3	crashed	efficientnet-b3
aug_class844_efficientnet-b3	crashed	efficientnet-b3
aug_class844_efficientnet-b3	crashed	efficientnet-b3
aug_class844_efficientnet-b0	crashed	efficientnet-b0
aug_class419_efficientnet-b0	crashed	efficientnet-b0
aug_class419_efficientnet-b3	crashed	efficientnet-b3
aug_class286_efficientnetB2	crashed	efficientnet-b2
aug_class286_efficientnetB3	crashed	efficientnet-b3
aug_smoothing_0.1_clear_img_efficientnet-b2	crashed	efficientnet-b2
aug_smoothing_0.1_clear_img_efficientnet-b2	finished	efficientnet-b2
aug_smoothing_0.1_clear_img_efficientnet-b2	crashed	efficientnet-b2
aug_smoothing_0.1_clear_img_efficientnet-b2	finished	efficientnet-b2
aug_smoothing_0.1_clear_img_efficientnet-b2	finished	efficientnet-b2
aug_smoothing_0.1_clear_img_efficientnet-b2	finished	efficientnet-b2
aug_smoothing_0.1_clear_img_efficientnet-b2	finished	efficientnet-b2
aug_smoothing_0.1_clear_img_efficientnet-b2	finished	efficientnet-b2

총 횟수 : 약 100회

평균 학습 시간 : 8시간

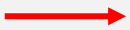


ENDAND

Classification 모델 적용

Accuracy (단위 : %)

제품 개수	Class_81	Class_286	Class_419
Top 1	36	41	41
Top 2		54	54
Top 3		60	60
Top 5	64	68	71



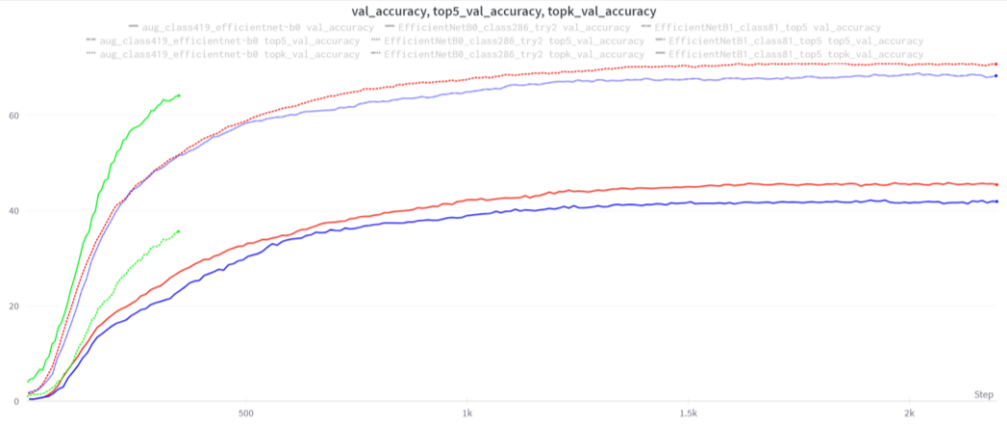
Accuracy (단위 : %)

제품 개수	Class_829
Top 1	70.742
Top 2	82.349
Top 3	86.607
Top 5	88.942

- 제품의 시리얼 넘버를 추천해주는 시스템을 구축하는데 상위의 여러 개를 추천해주는 방식도 고려하여 Top 1~5까지의 정확도를 측정

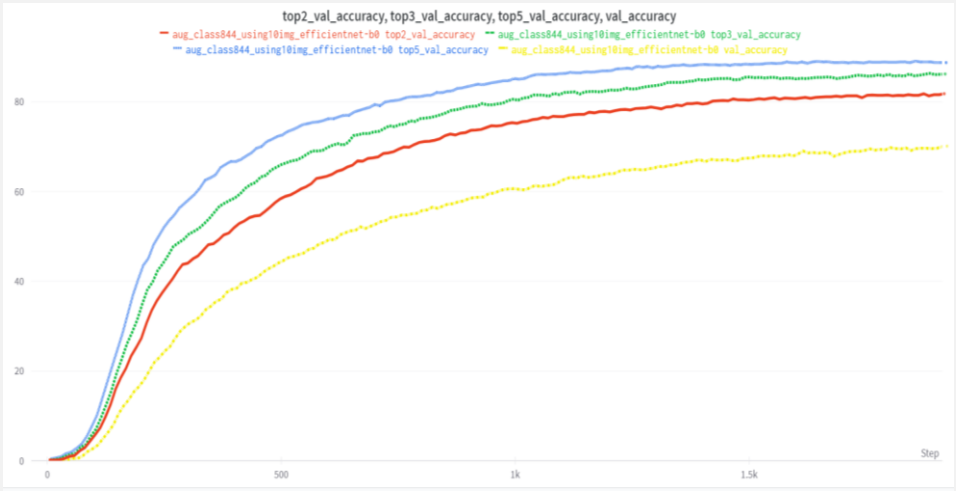
- 829 데이터가 제품의 개수가 많고, 제품 당 사진 수가 적음에도 불구하고 정확도가 높게 나옴

Classification 모델 적용



Top 1, 5 accuracy

- Class : 419
- Class : 286
- Class : 81

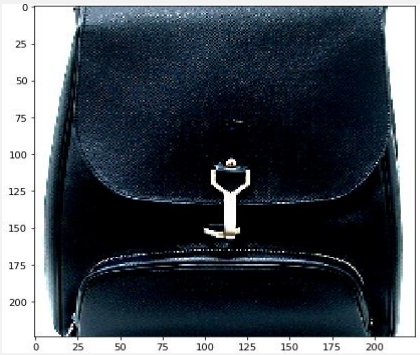


Class 829 Top 1~5

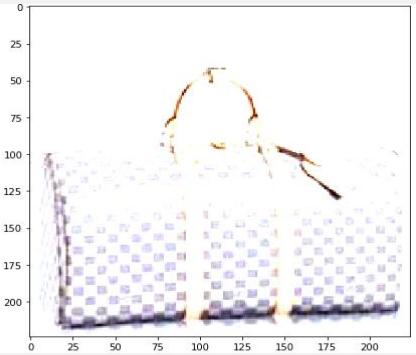
- Top 1
- Top 2
- Top 3
- Top 5

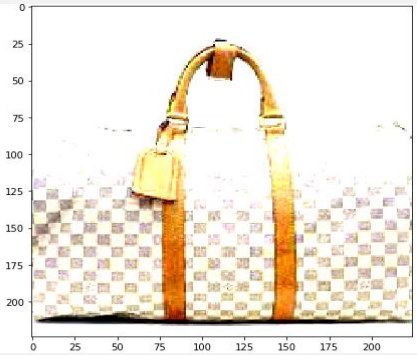
Classification 모델 적용

Truth

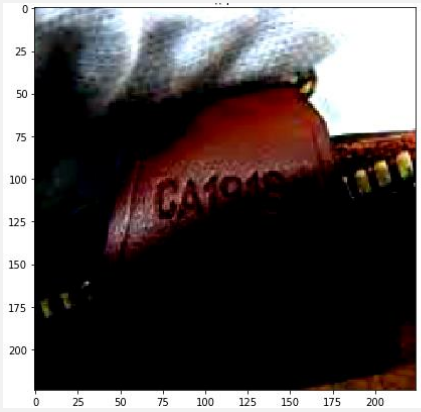


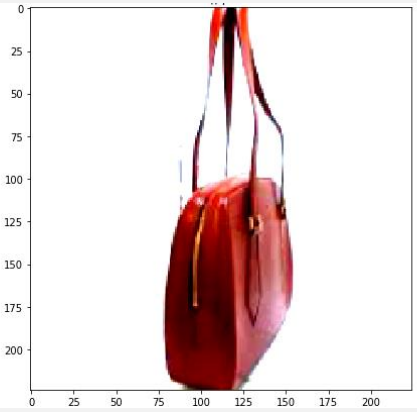
Truth



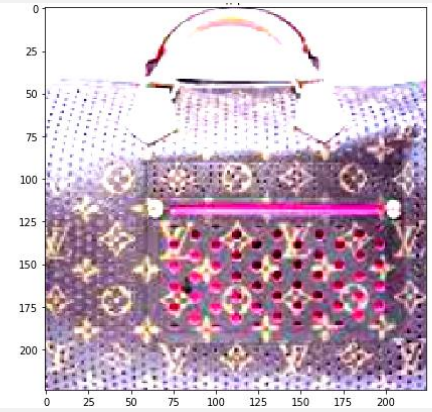


False





Truth



Metric Learning 모델 적용

1. Classification은 학습한 이미지에 대해서만, 인식이 가능하다.

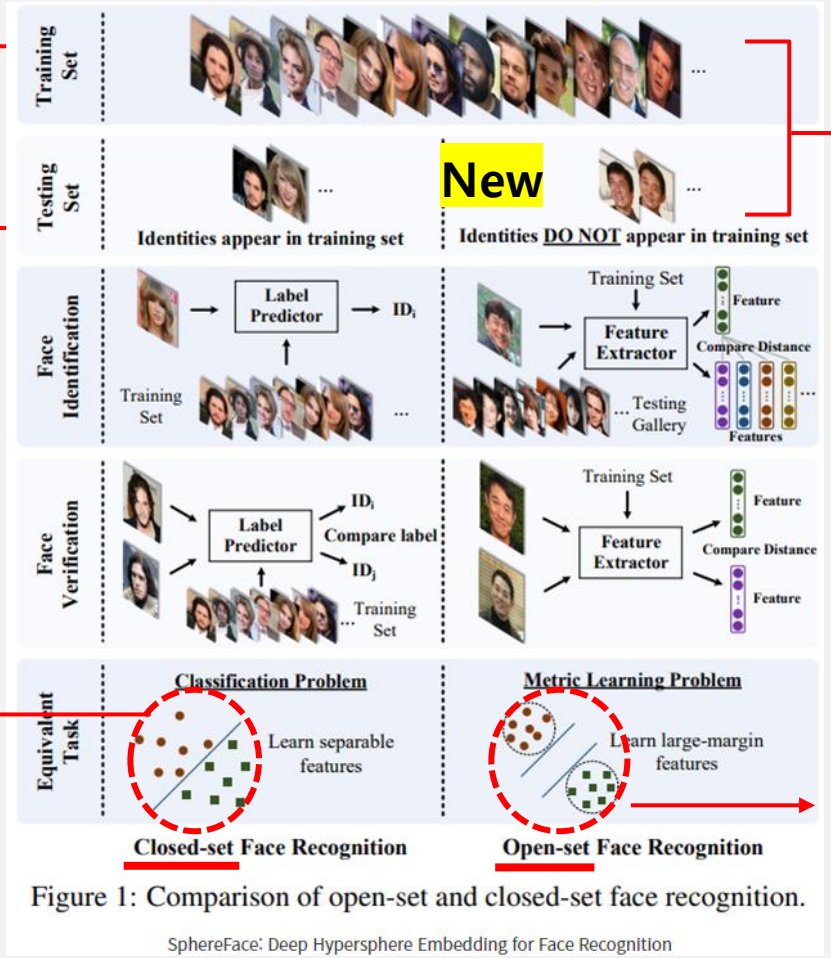


Figure 1: Comparison of open-set and closed-set face recognition.

SphereFace: Deep Hypersphere Embedding for Face Recognition

1. Metric Learning은 학습하지 않은 이미지도 DB로 구축만 하면 인식 가능하다.

2. Classification은 Feature 간의 Decision Boundary를 찾도록 학습한다.(Learn separable features)

2. Metric Learning은 비유사한 Feature들을 멀리 떨어지도록 학습한다.(Learn Large-Margin features)

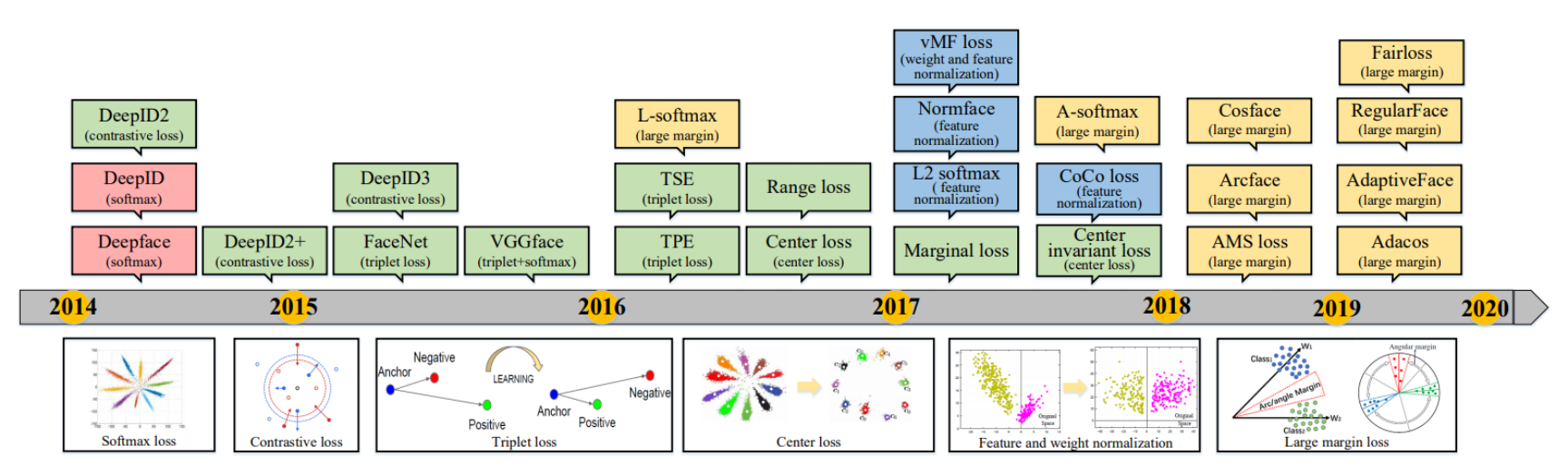
Metric Learning 모델 적용

Metric Learning의 핵심

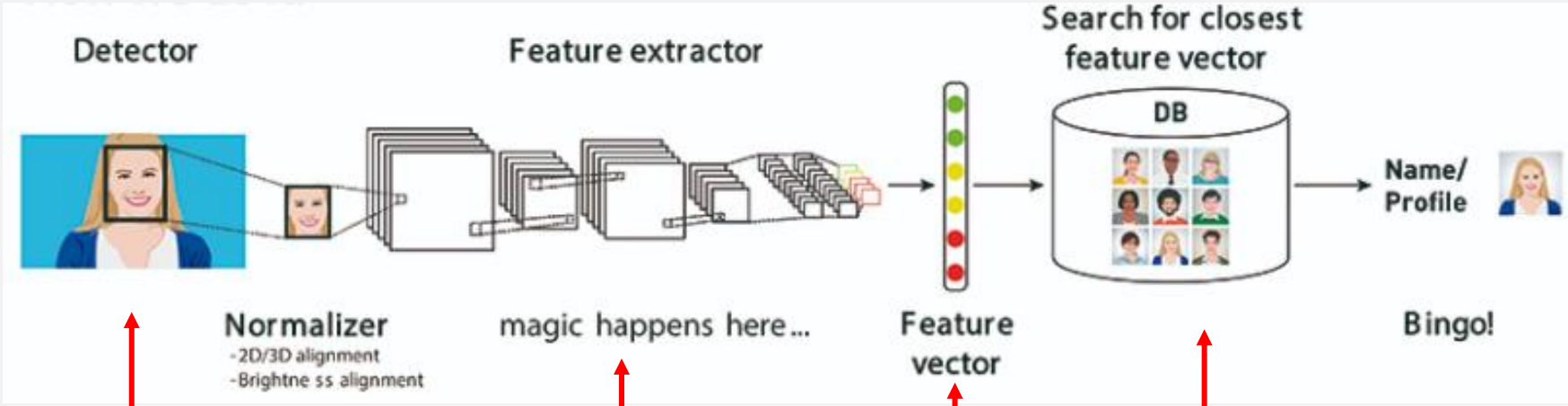
0. Embedding이란 무엇인가?
   - 문서, 자연어, 이미지, 그래프 등 데이터를 벡터 공간의 좌표로 표현
   - 차원이 큰 데이터를 본래의 성질을 최대한 유지한 상태에서 압축

1. 어떤 거리(유사도)를 사용할 것인가?
   - Euclidean, Covariance matrix(공분산 행렬), Cosine Similarity 등

2. 유사도 학습을 위해서, 어떤 Loss로 학습할 것인가?
   - Contrastive Loss, Triplet Loss, Margin Loss 등



Metric Learning 모델 적용



Dataset

Deep Architecture

1. Custom model
2. Resnet, VGG
3. Pre-training

Feature Extraction (vector)

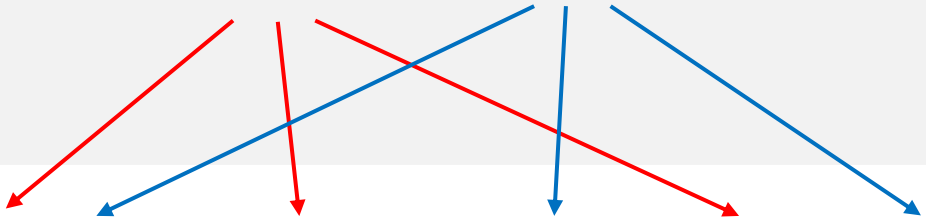
Embedding
+
Loss training

Metric Learning 모델 적용

Dataset

제품 수	Class_10	Class_81
제품 당 이미지 개수	110	50
총 개수	1,100	4,050
전처리 여부	O	X

- 제품(Class)의 개수가 증가할수록 성능의 저하
- 데이터가 일관성이 있을수록 성능이 향상되는 양상
- 적은 데이터를 활용 가능한 metric learning model의 경우에도 학습을 위한 일정량의 데이터가 필요
- 위의 조건을 충족 시킨 후 loss를 결정

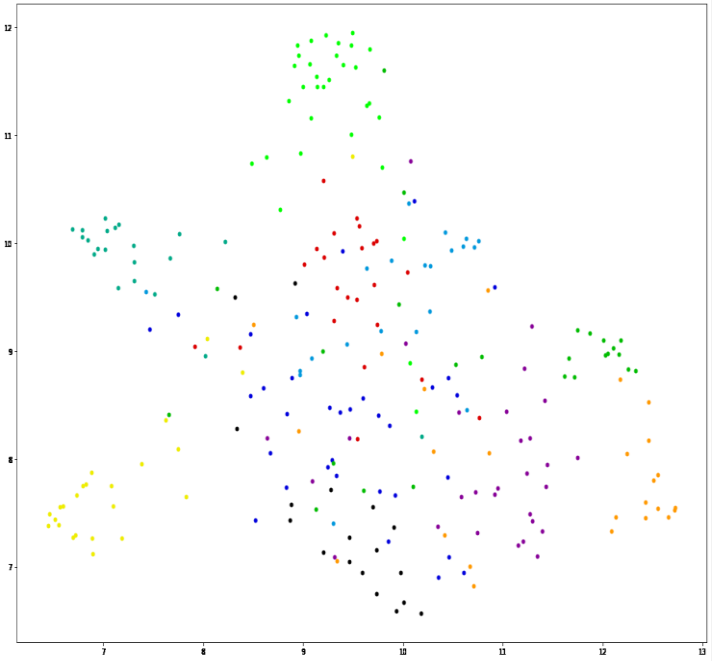


Accuracy

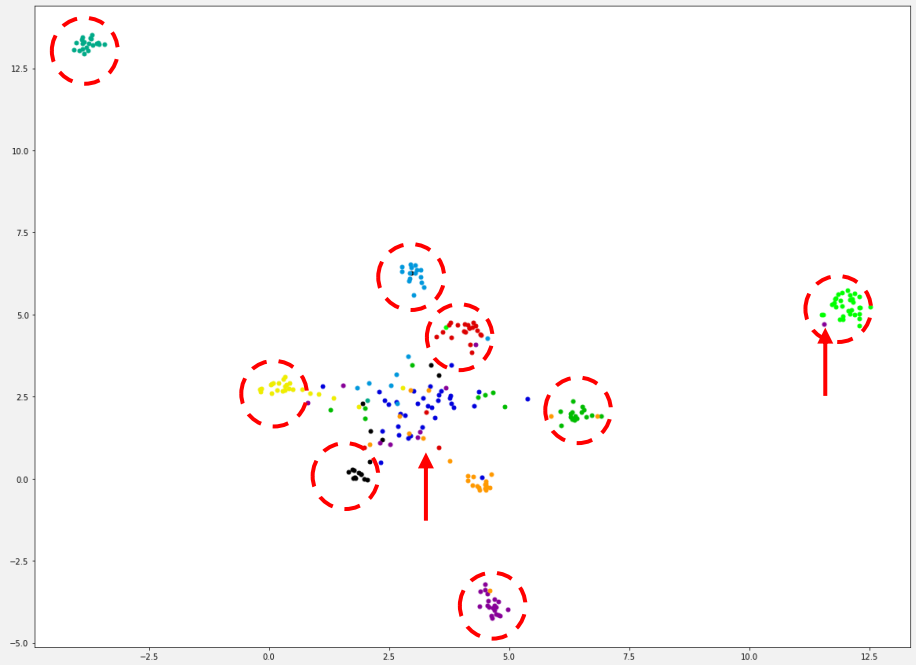
Triplet_10	Triplet_81	Center loss_10	Center loss_81	Cascaded embedding_10	Cascaded embedding_81
0.873	0.390	0.892	0.4107	0.698	0.109

Metric Learning 모델 적용

Plot Graph



Cascaded embedding_10 model (accuracy 0.698)



평가 및 느낀점

- **민규** -
  - 프로젝트에 맞는 형태로 정제된 데이터 셋을 수집하는 일이 가장 중요하다
  - 데이터의 중요성을 느낌

- **동원** -
  - 서비스 이용자의 입장에서 사고할 줄 알아야 한다고 느낌

- **형준** -
  - 데이터 상황에 따라 프로젝트의 성공여부가 달라질 만큼 중요하다고 생각
  - mmdetection을 활용하지 못한 점이 아쉬움

- **동환** -
  - 이미지 처리를 처음 접해보는 입장으로서 사전지식이 매우 중요하다고 느낌
  - 단순히 가방만 탐지하는 전처리 과정을 진행했지만 정확한 가방모델을 찾기 위해서는 그만큼의 데이터 수가 필요하다

Thank you