

SAVE US

비 전 있 는 비 전 공 자 팀

SAVE EARTH



서비스 산업 데이터를 활용한 빅데이터 8회차_H반 3조

INDEX

- Outline
- Process
- Expectation
- Review

One step

PROJECT OUTLINE

비전있는 비전공자

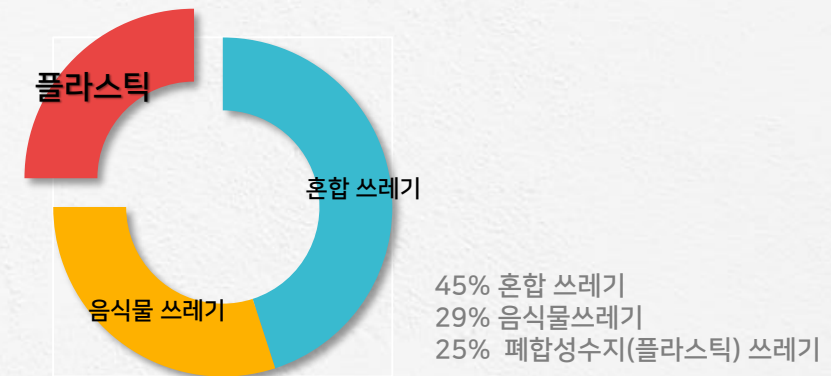
인류의 축복에서 재앙으로

플라스틱이 지구에 미치는 환경오염과 그에 따른 실태

전 세계 플라스틱 오염실태



- _쓰레기 문제는 환경에서 차지하는 비중이 매우 크고, 미래에는 쓰레기를 얼마나 줄이고 재활용 할 수 있느냐의 관점으로 이어질 것
- _최근 사회의 뜨거운 이슈인 ESG를 비롯한 탄소중립, Recycle등이 주목을 받고 있음
- _국내 쓰레기 배출 비중을 확인해보면 일 평균 45,000톤 정도의 쓰레기가 배출되고 있음



➡ 분리배출이 가능한 쓰레기의 종류에서 플라스틱의 비중이 높다는 것을 알 수 있음

인류의 축복에서 재앙으로

플라스틱이 지구에 미치는 환경오염과 그에 따른 실태

플라스틱, 한국의 위치는 지금 어디에?



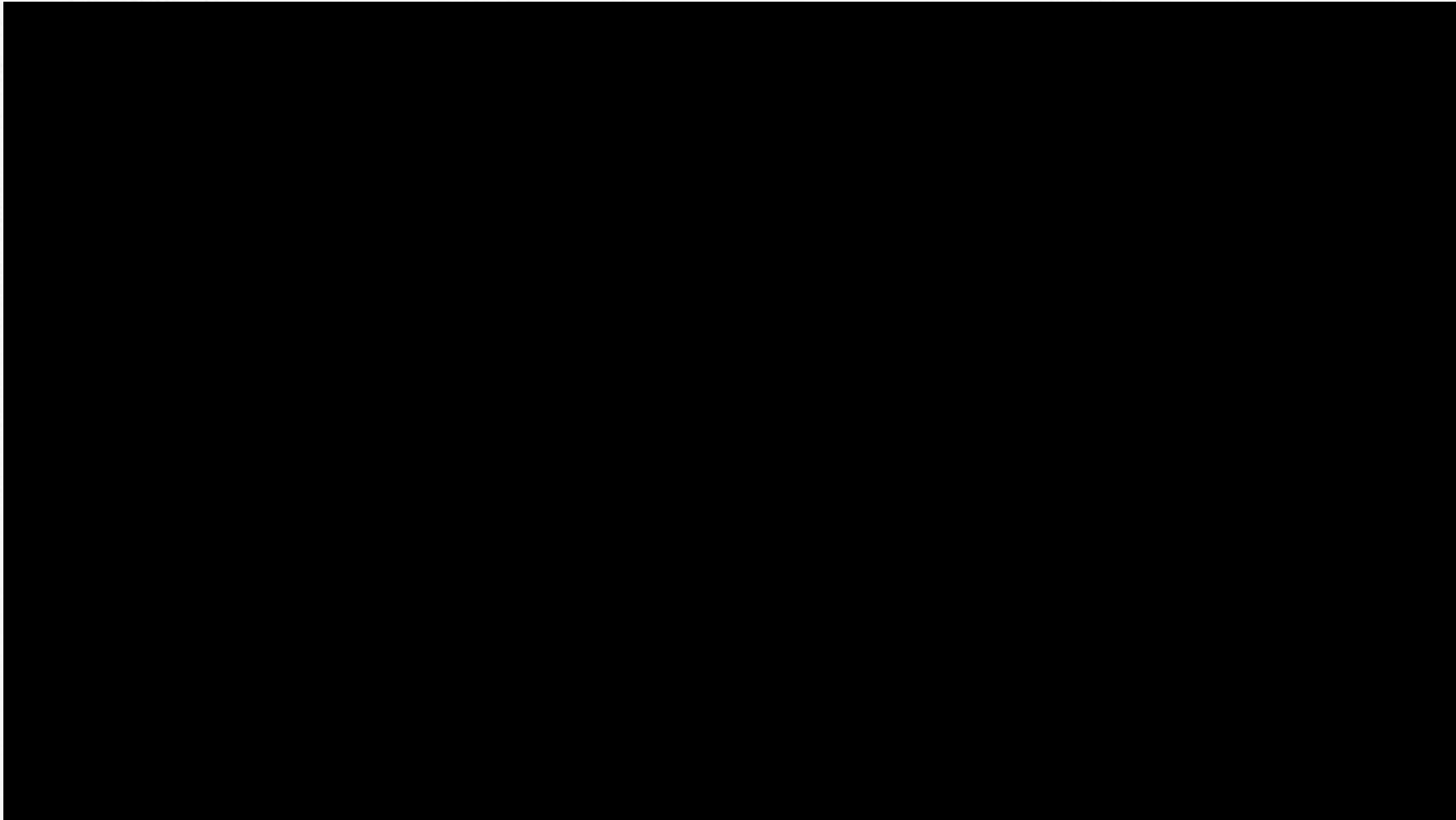
_한국 1인당 연간 플라스틱 소비량 88kg으로 '세계 3위'

_10년간 대한민국 플라스틱 발생량 70% 상승



플라스틱으로 가득 찬 세상

그린피스 제작



SAVE US, SAVE EARTH

재활용 가능한 페트병 이미지를 통한 분류 예측 모델 (다중 분류 모형)

플라스틱 재활용 생활화



_플라스틱 연간 생산량은 꾸준히 증가,
But 플라스틱 분리배출 및 분리수거에 대한 상식 부족

_재활용이 될 수 있는 비율이 가장 높은 자원은 플라스틱이며,
역설적으로 재활용이 가장 안되고 있는 자원 또한 플라스틱!



플라스틱은 생태계 오염 발생 원인의 1순위 자원이며,
플라스틱 1%만 줄여도 연간 640억 절감의 경제적 효과를 가져올 수 있음

SAVE US, SAVE EARTH

재활용 가능한 페트병 이미지를 통한 분류 예측 모델 (다중 분류 모형)

프로젝트 목표

- 1 재활용이 가장 많이 될 수 있는 플라스틱에 집중하여 재활용이 가능한 플라스틱과 재활용이 불가능한 플라스틱을 구분할 수 있는 이미지 다중 분류 모델 개발
- 2 6개의 카테고리(플라스틱, 재활용이 가능한 플라스틱, 캔, 유리, 종이, 기타)로 나누어 이미지 분류를 가능하게 함
- 3 합성곱 신경망 (Convolutional Neural Networks) 모델을 사용하여 다중 분류 모델 개발
- 4 다양한 모델 학습을 통해 가장 최적의 모델을 선택하여 새로운 쓰레기 이미지 데이터를 학습시켰을 때, 정확도(Accuracy)가 80-90% 이상이 나오도록 함
- 5 향후 우리의 모델이 사회에서 어떻게 활용될 수 있을지 기대 효과 제시

SAVE US, SAVE EARTH

재활용 가능한 페트병 이미지를 통한 분류 예측 모델 (다중 분류 모형)

Team. 비전있는 비전공자

Team 조직도

진세용(팀장)
전처리 & 모델 구축

김소연
전처리 & PPT

김성훈
모델 구축

김지찬
전처리 & PPT

문충현
자료 수집

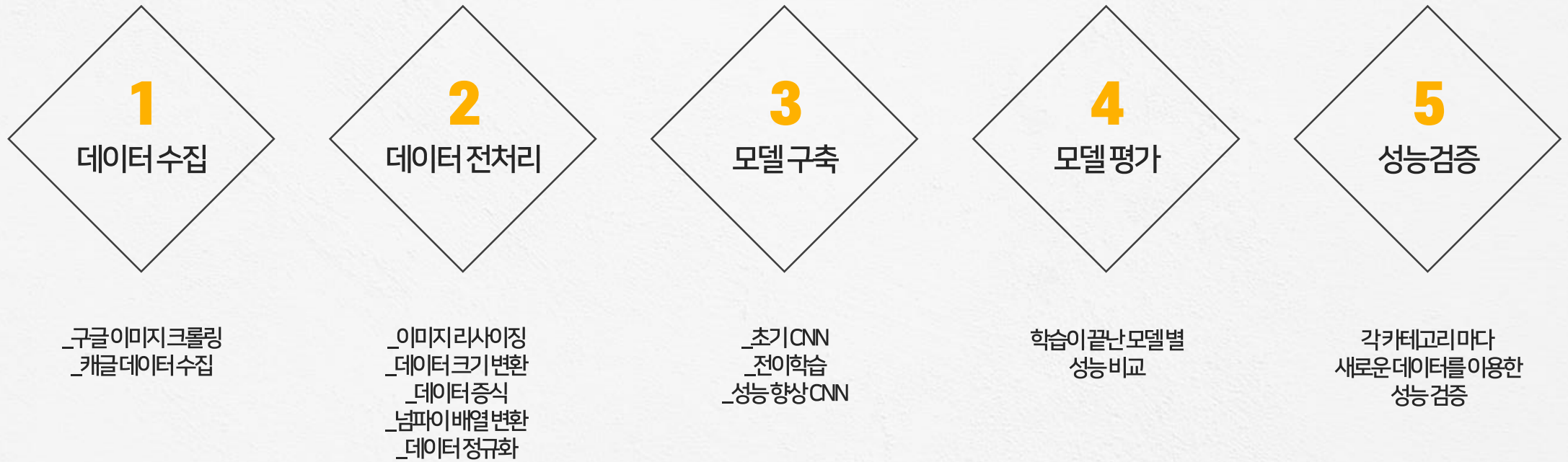
조영훈
모델 구축

Two step

PROJECT PROCESS

비전있는 비전공자

프로젝트 진행 프로세스



키워드 추출 및 데이터 수집

구글 이미지 크롤링 및 캐글 데이터 수집



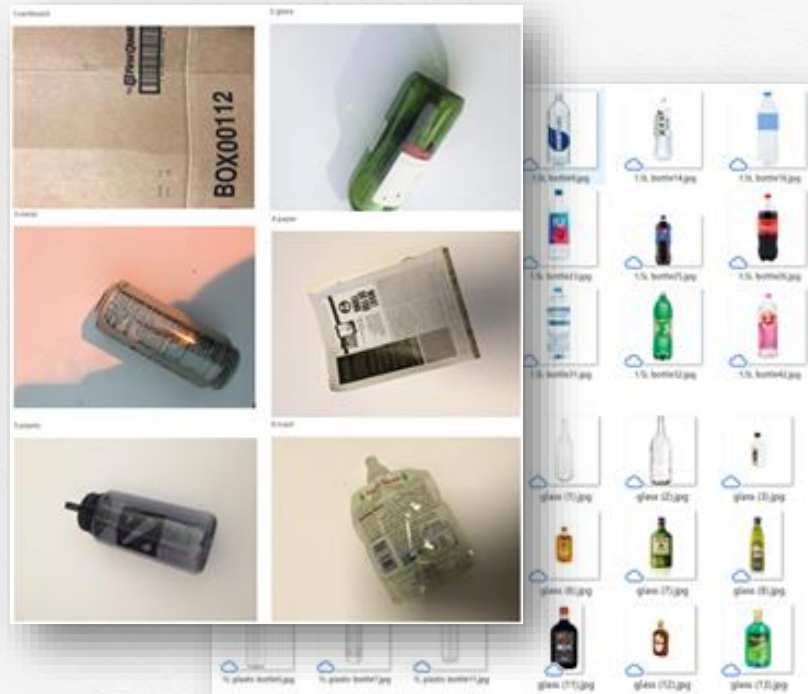
데이터 수집 요약

- 1 이미지 크롤링을 위한 키워드 100개 선정
- 2 한 키워드 당 대략 500장 정도의 이미지 크롤링
- 3 캐글 이미지 데이터 추가 확보
- 4 실제 가용 데이터로 사용 할 수 있는 이미지 분류 작업

데이터 분류

총 33,000개 이미지에서 6개의 카테고리 분류 작업

이미지 분류작업



실제 분류한 데이터 이미지

카테고리 설정

_6개의 카테고리 설정
{ '플라스틱' , '재활용이 가능한 플라스틱' , '캔' , '유리' , '종이' , '기타' }

_전체 수집 데이터 33,000개 中 가용 가능 데이터 2,720개

_추가 캐글 데이터 2,504개

>>> 총 학습 데이터 5,224개 확보
(전체 데이터 33,000개 에서 캐글 데이터 추가하여 사용 가능 데이터 15% 활용)

데이터 전처리

이미지 증식, 리사이징, 넘파이 배열 변환, 데이터 정규화



전처리 과정 요약

1 이미지 증식

2 분류 카테고리 지정

3 이미지 리사이징

4 넘파이 배열 변환

5 Train / Test Set 나누기

6 npy 파일 저장

7 데이터 정규화

이미지 증식

데이터 전처리

이미지 증식

```
datagen = ImageDataGenerator(  
    rotation_range=40,  
    width_shift_range=0.2,  
    height_shift_range=0.2,  
    shear_range=0.2,  
    zoom_range=0.2,  
    horizontal_flip=True,  
    fill_mode='nearest')
```

→ 적은 데이터셋에서 최대한 많은 정보를 뽑아 내서 학습할 수 있도록 이미지 증식

→ Keras의 ImageDataGenerator 클래스를 사용

→ 이미지 회전, 평행 이동, 임의 확대/축소 등의 인자를 사용하여 이미지 증식

>>> 5,200개의 Raw Data에서 53,000개까지 데이터 증식

분류 카테고리 지정 & 이미지 리사이징

데이터 전처리

분류 카테고리 지정

```
categories = ['Can', 'Glass', 'Paper',  
             'Plastic', 'RecycledPlastic',  
             'Trash']
```

_이미지를 Numpy 배열로 변환하기 위해 첫번째로 증식한 이미지 데이터의 카테고리를 지정

이미지 리사이징

```
image_w = 64  
image_h = 64
```

_각기 다른 사이즈의 이미지를 한 사이즈 (64*64)로 통일

넘파이 배열 변환

데이터 전처리

이미지 증식

_모델에 학습시키기 위해 이미지를 Numpy 배열로 변환

_이미지는 RGB 형식으로 변환

```
from PIL import Image
import glob
import numpy as np

## 데이터 저장 리스트
X = [] # 이미지(numpy 배열) 리스트
Y = [] # label 리스트
print('===== 변환을 시작합니다. =====')
for idx, cat in enumerate(categories):
    ## label 지정
    label = [0 for i in range(nb_classes)]
    label[idx] = 1
    ## 이미지 경로 리스트
    image_dir = folder_dir + '/' + cat
    files = glob.glob(image_dir + '/' + '*.jpg')
    img = Image.open(f)
    img = img.convert("RGB")
    img = img.resize((image_w, image_h))
    data = np.asarray(img)
    X.append(data)
    Y.append(label)
    ## 저장 진행 상황 출력
    if i % (len(files)/10) == 0:
        print('.', end='')
        print(categories[idx], '이미지 >>> Numpy 배열[완료]')
print('===== 변환을 성공적으로 완료했습니다! =====')
X = np.array(X)
Y = np.array(Y)
```

Train/Test Set & 데이터 정규화

데이터 전처리

___ Train/Testset 나누기

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
train_test_set = (X_train, X_test, y_train, y_test)
```

_Sklearn의 train_test_split 모듈을 활용하여 test set의 사이즈를 0.2로 지정

검입이력

___ npy 파일 저장

_전처리가 끝난 train_test_set 데이터를 npy 파일로 저장

___ 데이터 정규화

```
X_train = X_train.astype("float") / 255
X_test = X_test.astype("float") / 255
```

_모델 학습 전에 데이터 값의 범위를 0~1 사이의 값으로 바꿈
_Scale이 큰 Feature의 영향이 비대해지는 것을 방지

모델 구축

CNN, 전이 학습



모델 구축 과정

1 초기 CNN

2 전이학습

- 1. VGG 16
- 2. ResNet50

3 성능향상 CNN

모델 구축 프로세스

CNN, 전이 학습

CNN -> 전이학습 -> 성능 향상 CNN

초기 CNN 모델을 구축했을 때,
현재 확보한 이미지 데이터의 양이 적어 모델의 성능이 낮은 것으로 판단

적은 데이터셋으로도 좋은 성능을 내는 전이 학습 모델을 사용
(VGG16, ResNet50)

초기 CNN 모델보다 긴 학습 시간이 필요하지만 그만큼의 성능 향상은 나오지 않음

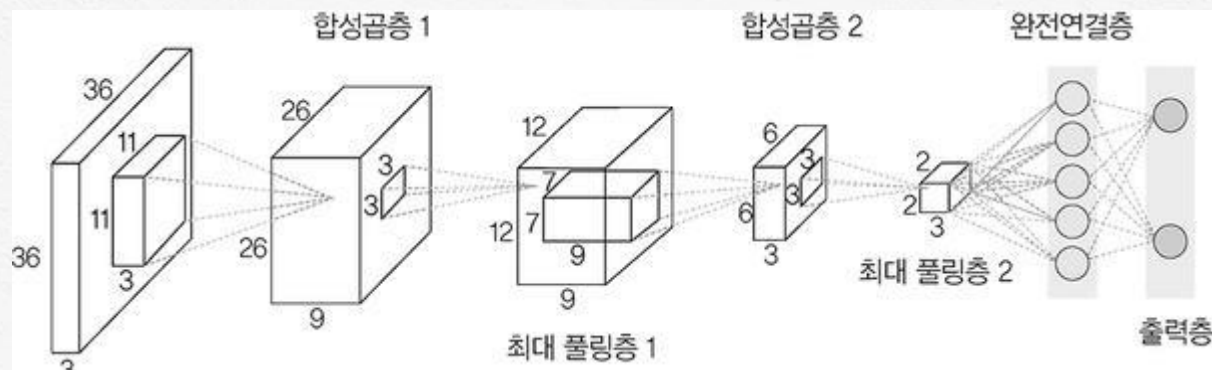
다시 초기 CNN 모델 돌아가 Conv2D, MaxPooling2D, Dropout를
적절히 사용하여 모델 성능을 향상 시킴

CNN

모델 구축

CNN?

_음성 인식이나 이미지/영상 인식에서 주로 사용되는 CNN 모델(합성곱 신경망)을 채택하여 사용
_CNN 모델은 다차원 배열 데이터를 처리하도록 구성되어 컬러 이미지 같은 다차원 배열 처리에 특화되어 있으며, 다음과 같이 다섯 개의 층으로 구성



_ (64, 64, 3) numpy 배열로 전처리된 이미지 데이터가 입력층을 지나
합성곱층과 풀링층을 거치면서 입력 이미지의 주요 특성 벡터(feature vector) 추출

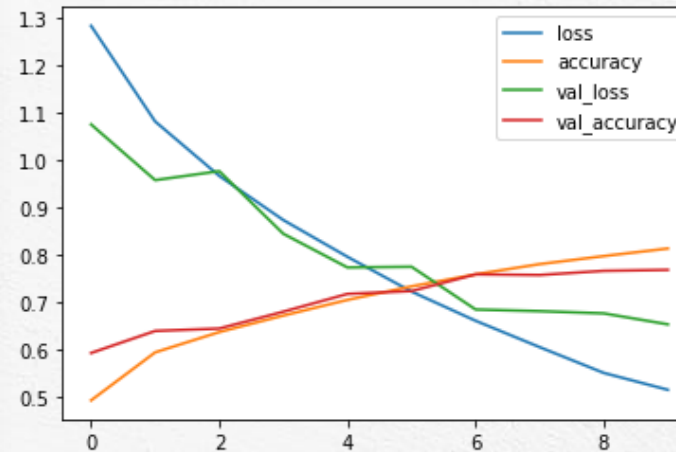
_ 그 후 추출된 주요 특성 벡터들은 완전연결층을 거치면서 1차원 벡터로 변환되며,
마지막으로 출력층에서 활성화 함수인 소프트맥스(softmax) 함수를 사용하여 최종 결과 출력

Custom CNN

모델 구축

초기 CNN 모델

```
1 ## 모델 구조 정의
2 model = Sequential()
3
4 ## 1층
5 model.add(Conv2D(32, (3, 3), input_shape=X_train.shape[1:], padding='same'))
6 model.add(Activation('relu'))
7 model.add(MaxPooling2D(pool_size=(2, 2)))
8 # model.add(Dropout(0.25))
9
10 ## 2층
11 model.add(Conv2D(64, (3, 3), padding='same')) # 합성곱층 2
12 model.add(Activation('relu')) # 활성화 함수
13
14 ## 3층
15 model.add(MaxPooling2D(pool_size=(2, 2))) # 풀링층 2
16 model.add(Activation('relu')) # 활성화 함수
17
18 ## 4층
19 model.add(Conv2D(64, (3, 3))) # 합성곱층 3
20 model.add(MaxPooling2D(pool_size=(2, 2))) # 풀링층 2
21 model.add(Dropout(0.25)) # 과적합 방지
22
23 ## 완전연결층
24 model.add(Flatten()) # 1차원 벡터 형태로 reshape
25 model.add(Dense(512)) # 출력
26 model.add(Activation('relu'))
27 model.add(Dropout(0.5))
28
29 ## 출력층
30 model.add(Dense(nb_classes))
31 model.add(Activation('softmax'))
```



loss_0.51
accuracy_0.81
val_loss_0.65
val_accuracy_0.76

_총 4개의 층으로 구성하였고, 활성화 함수로는 relu와 softmax 함수를 사용
_Dropout을 사용하여 과적합을 방지

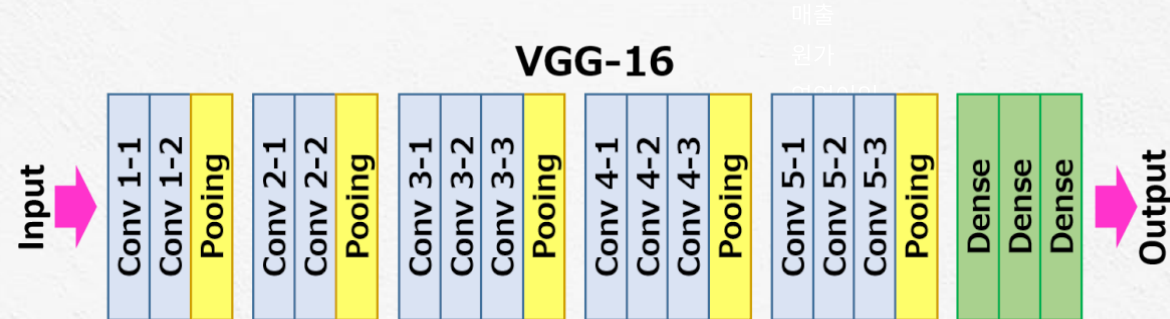
전이학습

전이학습 및 사용한 전이학습 모델 소개

전이학습?

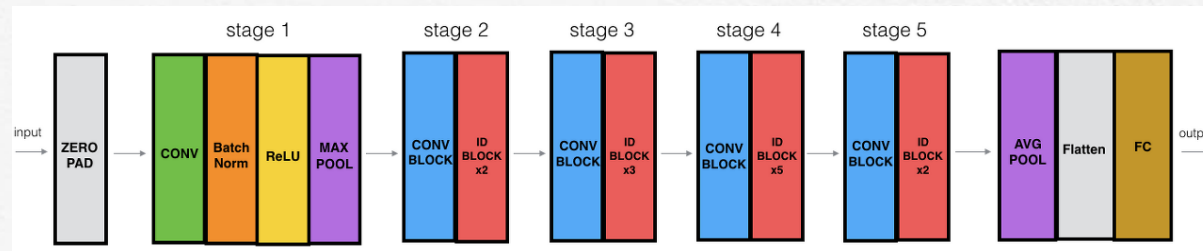
- _이미지넷과 같이 아주 큰 데이터셋에 훈련된 모델을 가지고 와서 해결하고자 하는 과제에 맞게 보정하여 사용하는 것
- _전이학습을 수행하지 않은 모델들보다 빠르고 정확하게 정확도를 달성할 수 있음 (전이 학습은 바닥부터 훈련시킬 필요가 없음)

VGG-16



- _Oxford에서 개발한 모델로, Convolution과 Pooling을 반복하여 완전연결층으로 구성되어 있는 모델

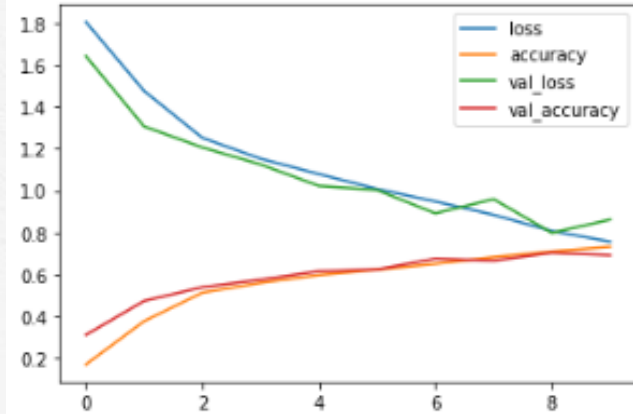
ResNet50



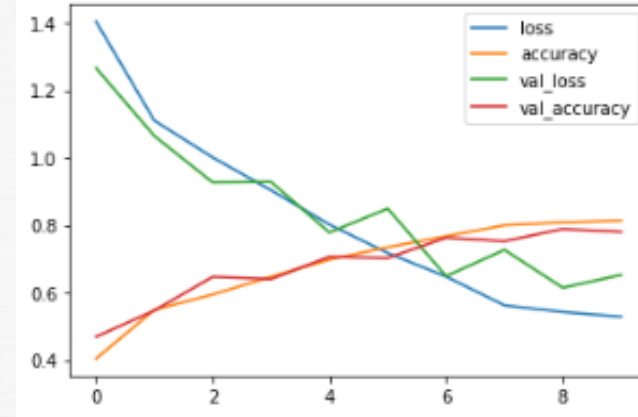
- _Microsoft에서 개발한 모델 VGG-16과 같은 구성이지만, 이전 layer와 다음 layer를 이어주는 연결선이 추가된 모델

사용한 전이 학습 모델

VGG-16



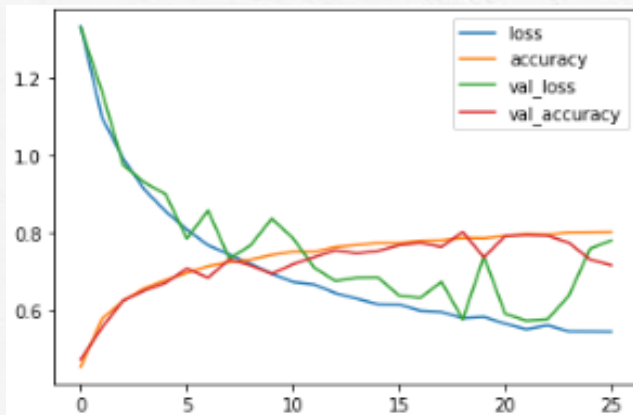
loss_0.76
accuracy_0.73
val_loss_0.86
val_accuracy_0.69



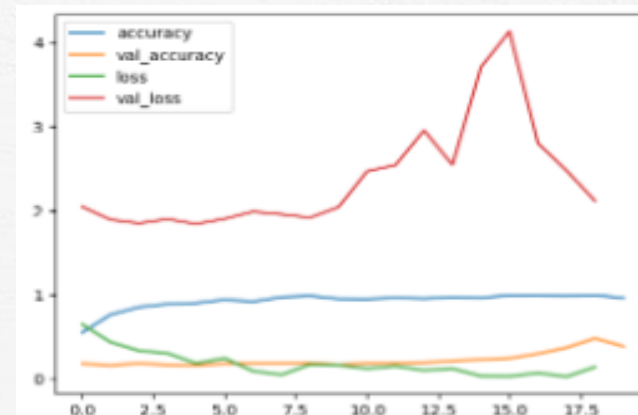
loss_0.53
accuracy_0.81
val_loss_0.65
val_accuracy_0.78

>>> VGG-16을 사용했을 때, accuracy와 val_accuracy가 어느정도 유사하게 나왔으나, CustomCNN을 사용했을 때보다 성능이 낮음

ResNet50



loss_0.13
accuracy_0.95
val_loss_0.65
val_accuracy_0.78

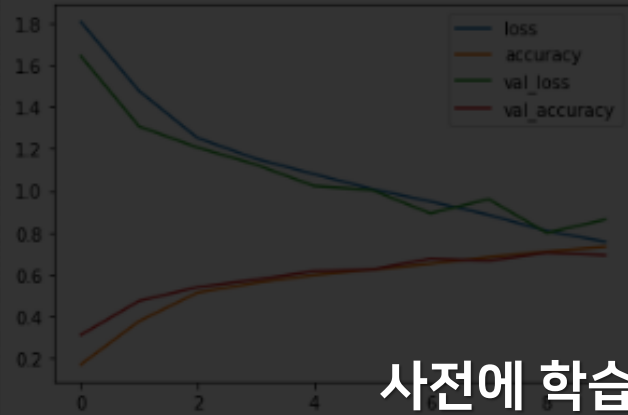


loss_0.14
accuracy_0.96
val_loss_2.11
val_accuracy_0.38

>>> ResNet50을 사용했을 때, accuracy는 높게 나왔으나, 상대적으로 val_accuracy가 적게 나왔으며, 심한경우 과적합(overfitting)이 발생

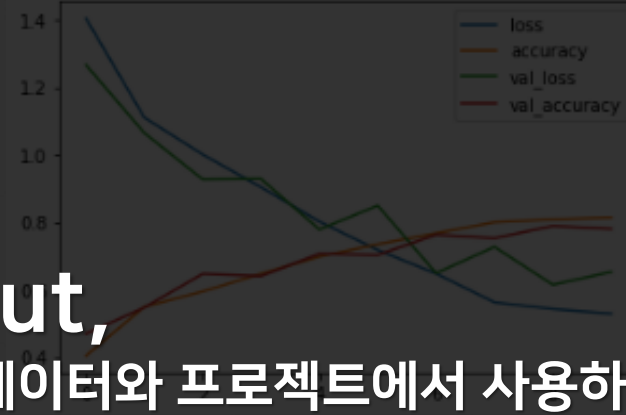
사용한 전이 학습 모델

VGG-16



loss_0.76
accuracy_0.73
val_loss_0.86
val_accuracy_0.65

But,



loss_0.53
accuracy_0.81
val_loss_0.65
val_accuracy_0.78

사전에 학습할 때의 이미지 데이터와 프로젝트에서 사용하는

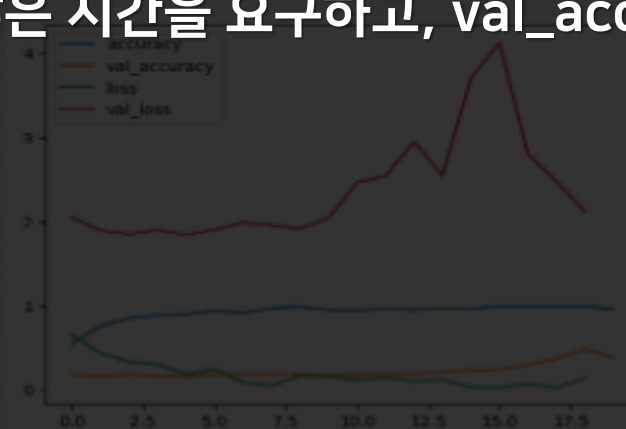
이미지 데이터의 크기가 달라 원하는 성능을 얻어내지 못하였고 심한 경우 과적합이 발생

>>> VGG-16을 사용했을 때, accuracy와 val_accuracy가 어느정도 유사하게 나왔으나, CustomCNN을 사용했을 때보다 성능이 낮음

ResNet50



loss_0.13
accuracy_0.95
val_loss_0.65
val_accuracy_0.78



loss_0.14
accuracy_0.96
val_loss_2.11
val_accuracy_0.38

사용하는 데이터에 비하여 층이 과다하여 오히려 많은 시간을 요구하고, val_accuracy가 개선되지 않음

>>> ResNet50을 사용했을 때, accuracy는 높게 나왔으나, 상대적으로 val_accuracy가 적게 나왔으며, 심한 경우 과적합(overfitting)이 발생

모델 평가

가장 성능 좋은 모델 찾기

4 모델 평가

모델 찾기

학습이 끝난 모델은 엑셀 시트에 기록하여 각 모델 별 성능 비교

>>> 가장 성능 좋은 모델 찾기

모델 평가

가장 성능 좋은 모델 찾기

학습이 끝난 모델 평가

모델 성능표							
순번	모델 책임자	이미지 크기	카테고리 별 이미지 수(전체)	모델 구조	Batch_size	Epochs	accuracy & loss
1	영훈	64 * 64	4500(27000)	Custom CNN	32	10	loss: 0.5146 - accuracy: 0.8124 - val_loss: 0.6525 - val_accuracy: 0.7674
2	영훈	위와 동일	위와 동일	Custom CNN	32	20	loss: 0.2600 - accuracy: 0.9087 - val_loss: 0.7181 - val_accuracy: 0.7650
3	영훈	위와 동일	위와 동일	위와 동일+1층 dropout 추가	32	20	loss: 0.3717 - accuracy: 0.8679 - val_loss: 0.7204 - val_accuracy: 0.7700
4	영훈	위와 동일	위와 동일	전이학습(VGG16)	32	10	loss: 0.7589 - accuracy: 0.7323 - val_loss: 0.8617 - val_accuracy: 0.6921
5	영훈	위와 동일	4500(22500)	전이학습(VGG16)	32	10	loss: 0.5280 - accuracy: 0.8134 - val_loss: 0.6519 - val_accuracy: 0.7750
6	세훈	64 * 64	4500(27000)	Res_Net50	64	30	train test loss: 0.1261 - accuracy: 0.9513 - val_loss: 0.6542 - val_accuracy: 0.7801
7	영훈	128 * 128	4500(27000)	Res_Net50	500	20	loss: 0.1391 - accuracy: 0.9576 - val_loss: 2.1139 - val_accuracy: 0.3638
8	소민	192 * 192	5000(30000)	초기 학습 모델	32	10	loss: 0.4117 - accuracy: 0.8598 - val_loss: 1.1766 - val_accuracy: 0.6560

약 30회에 걸친 구성 모델 중 23번이 가장 높은 수준의 validation accuracy (약 88%)를 보임

Best Model !

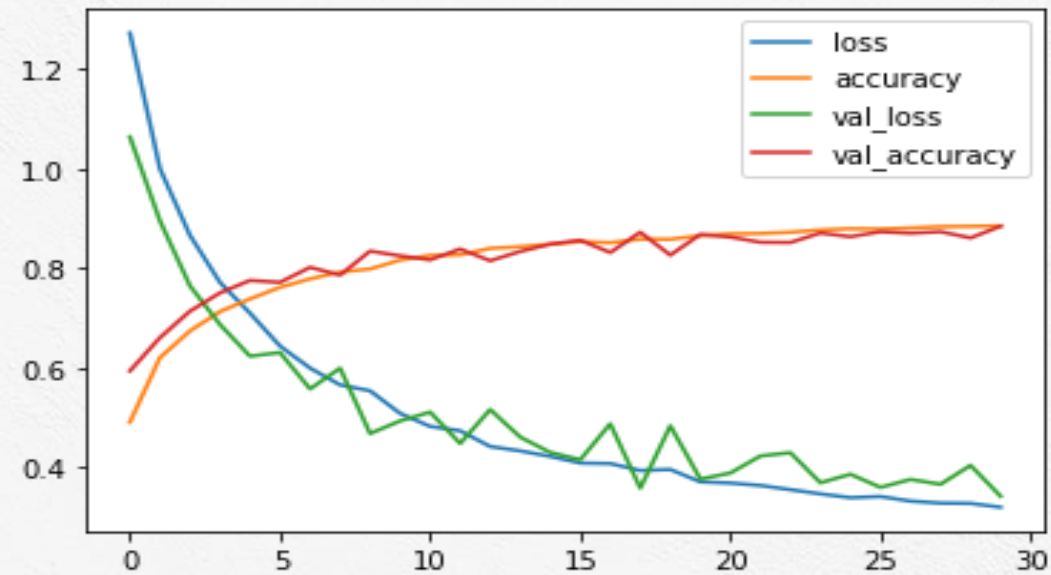
매출
원가
영업이익

순번	모델 책임자	이미지 크기	카테고리 별 이미지 수	모델 구조	Batch_size	Epochs	Accuracy & loss
23	영훈	64*64	53061	Custom CNN (4층)	64	30	loss_0.32 accuracy_0.89 val_loss_0.34 val_accuracy_0.88

모델 평가

가장 성능 좋은 모델 찾기

가장 성능 좋은 모델



위 그래프를 보면, loss, val_loss는 epochs가 늘어날수록 감소하고, accuracy, val_accuracy의 경우에는 epochs가 늘어날수록 증가한다는 점으로 보아 과적합없이 안정적인 수준을 보인다고 할 수 있음

성능 검증

각 카테고리 별 데이터 검증



성능 검증

각 카테고리마다 10개의 새로운 데이터를 이용하여 성능을 검증

성능 검증

Plastic

Plastic Predict



Plastic1.jpg



Plastic2.jpg



Plastic3.jpg



Plastic4.jpg



Plastic5.jpg



Plastic6.jpg



Plastic7.jpg



Plastic8.jpg



Plastic9.jpg



Plastic10.jpg


```
1/1 [=====] - 0s 19ms/step
Plastic1.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
Plastic10.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 18ms/step
Plastic2.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 20ms/step
Plastic3.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 20ms/step
Plastic4.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 20ms/step
Plastic5.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 19ms/step
Plastic6.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 20ms/step
Plastic7.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 20ms/step
Plastic8.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 19ms/step
Plastic9.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 18ms/step
```

Plastic Prediction rate = 90%


성능 검증

Recycle Plastic

Recycle Plastic Predict




RecycledPlastic1.jpg



RecycledPlastic2.jpg



RecycledPlastic3.png




RecycledPlastic4.jpg




RecyclePlastic5.jpg




RecyclePlastic6.jpg




RecyclePlastic7.jpg



RecyclePlastic8.jpg



RecyclePlastic9.jpg



RecyclePlastic10.jpg

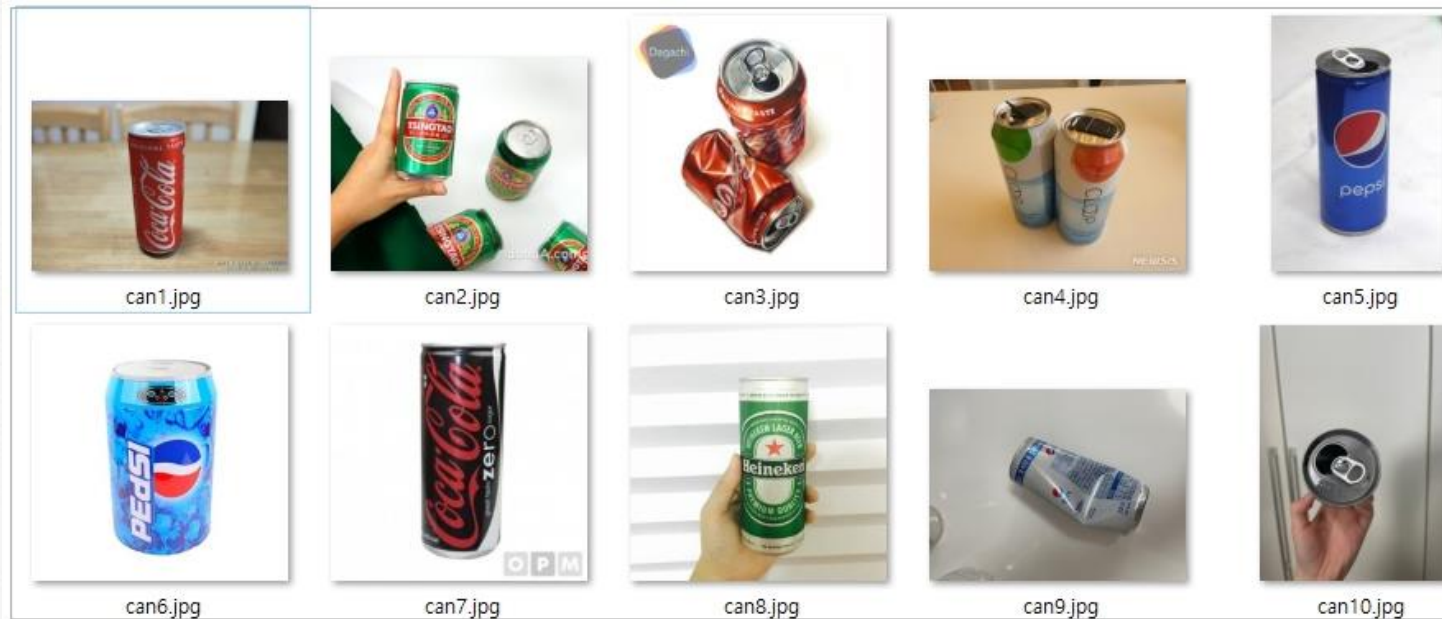
```
1/1 [=====] - 0s 18ms/step
RecycledPlastic1.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 19ms/step
RecycledPlastic2.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 19ms/step
RecycledPlastic3.png 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
RecycledPlastic4.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
RecyclePlastic10.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
RecyclePlastic5.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
RecyclePlastic6.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
RecyclePlastic7.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
RecyclePlastic8.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 21ms/step
RecyclePlastic9.jpg 사진은 RecycledPlastic 입니다.
```

Recycle Plastic Prediction rate = 100%

성능 검증

Can

Can Predict



```
1/1 [=====] - 0s 19ms/step  
can1.jpg 사진은 trash 입니다.  
1/1 [=====] - 0s 18ms/step  
can10.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 18ms/step  
can2.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 19ms/step  
can3.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 19ms/step  
can4.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 18ms/step  
can5.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 18ms/step  
can6.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 20ms/step  
can7.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 19ms/step  
can8.jpg 사진은 Plastic 입니다.  
1/1 [=====] - 0s 20ms/step  
can9.jpg 사진은 RecycledPlastic 입니다.
```

Can Prediction rate = 70%

성능 검증

Glass

Glass Predict

				
glass1.jpg	glass2.jpg	glass3.jpg	glass4.jpg	glass5.jpg
				
glass6.jpg	glass7.jpg	glass8.jpg	glass9.jpg	glass10.jpg


```
1/1 [=====] - Us 19ms/step  
glass1.jpg 사진은 RecycledPlastic 입니다.  
1/1 [=====] - 0s 20ms/step  
glass10.jpg 사진은 Glass 입니다.  
1/1 [=====] - 0s 21ms/step  
glass2.jpg 사진은 RecycledPlastic 입니다.  
1/1 [=====] - 0s 22ms/step  
glass3.jpg 사진은 Glass 입니다.  
1/1 [=====] - 0s 21ms/step  
glass4.jpg 사진은 RecycledPlastic 입니다.  
1/1 [=====] - 0s 20ms/step  
glass5.jpg 사진은 Glass 입니다.  
1/1 [=====] - 0s 20ms/step  
glass6.jpg 사진은 Glass 입니다.  
1/1 [=====] - 0s 19ms/step  
glass7.jpg 사진은 Glass 입니다.  
1/1 [=====] - 0s 19ms/step  
glass8.jpg 사진은 Glass 입니다.  
1/1 [=====] - 0s 21ms/step  
glass9.jpg 사진은 Glass 입니다.
```

glass Prediction rate = 70%

성능 검증

Paper

Paper Predict

				
Paper1.jpg	Paper2.png	Paper3.jpg	paper4.jpg	paper5.jpg
				
paper6.jpg	paper7.jpg	paper8.jpg	paper9.jpg	paper10.jpg

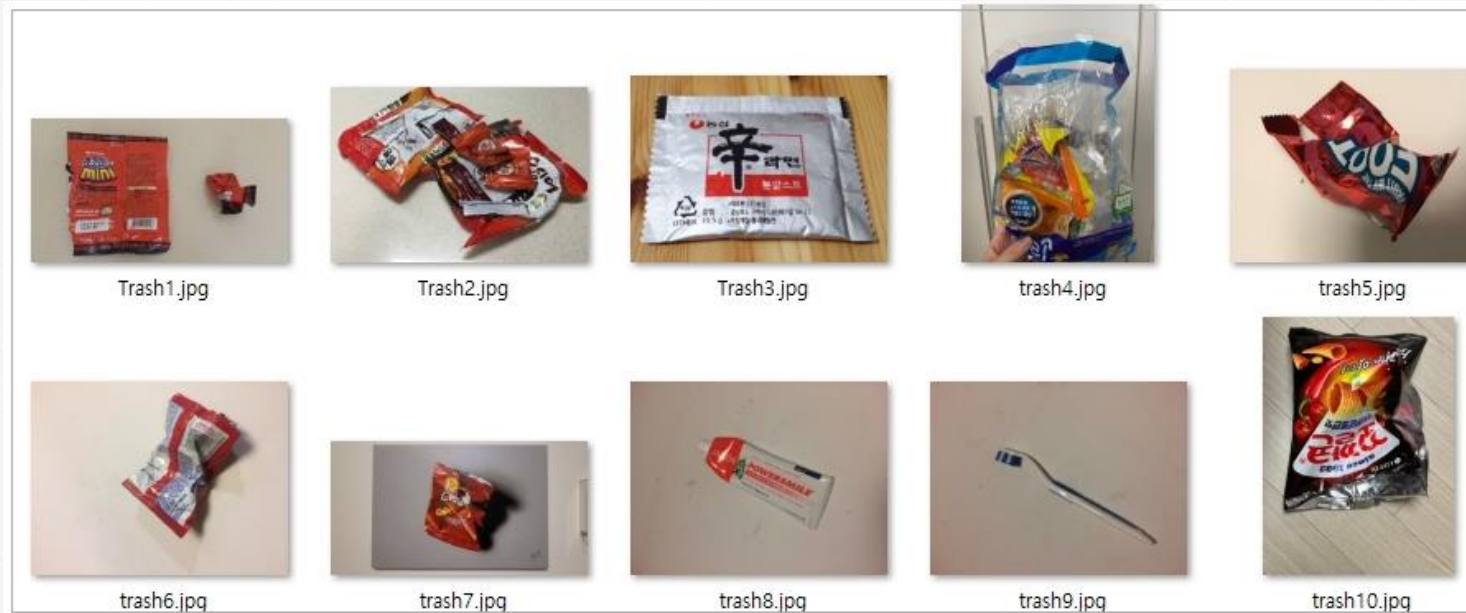
```
1/1 [=====] - 0s 22ms/step
Paper1.jpg 사진은 Paper 입니다.
1/1 [=====] - 0s 21ms/step
paper10.jpg 사진은 RecycledPlastic 입니다.
1/1 [=====] - 0s 20ms/step
Paper2.png 사진은 Paper 입니다.
1/1 [=====] - 0s 20ms/step
Paper3.jpg 사진은 Paper 입니다.
1/1 [=====] - 0s 20ms/step
paper4.jpg 사진은 Paper 입니다.
1/1 [=====] - 0s 20ms/step
paper5.jpg 사진은 Paper 입니다.
1/1 [=====] - 0s 20ms/step
paper6.jpg 사진은 Paper 입니다.
1/1 [=====] - 0s 20ms/step
paper7.jpg 사진은 Plastic 입니다.
1/1 [=====] - 0s 19ms/step
paper8.jpg 사진은 Paper 입니다.
1/1 [=====] - 0s 19ms/step
paper9.jpg 사진은 Paper 입니다.
1/1 [=====] - 0s 19ms/step
```

Paper Prediction rate = 80%

성능 검증

Trash

Trash Predict



```
1/1 [=====] - 0s 19ms/step  
Trash1.jpg 사진은 Paper 입니다.  
1/1 [=====] - 0s 19ms/step  
trash10.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 21ms/step  
Trash2.jpg 사진은 Paper 입니다.  
1/1 [=====] - 0s 24ms/step  
Trash3.jpg 사진은 Plastic 입니다.  
1/1 [=====] - 0s 20ms/step  
trash4.jpg 사진은 Plastic 입니다.  
1/1 [=====] - 0s 20ms/step  
trash5.jpg 사진은 trash 입니다.  
1/1 [=====] - 0s 18ms/step  
trash6.jpg 사진은 trash 입니다.  
1/1 [=====] - 0s 21ms/step  
trash7.jpg 사진은 Can 입니다.  
1/1 [=====] - 0s 19ms/step  
trash8.jpg 사진은 trash 입니다.  
1/1 [=====] - 0s 19ms/step  
trash9.jpg 사진은 trash 입니다.
```

Trash Prediction rate = 40%

성능 검증

성능 검증 결과

_Recycle Plastic과 Plastic의 경우에는 모델의 성능에 따라 좋은 결과 도출

_Can, Glass, Paper 의 경우에는 무난한 결과를 도출

_Trash의 경우 다양한 종류의 데이터가 학습되었기에 다른 카테고리 대비 예측이 어려웠음으로 예상

Three step

PROJECT EXPECTATION

비전있는 비전공자

활용 기대 방안

1

1인 가구가 증가함에 따라 정확한 재활용 분류 방법을 알지 못해도 카메라만 있다면, 재활용 분리기 모델을 통해서 재활용 분류가 가능

2

일반 종량제로 분류되어 버려진 혼합쓰레기들을 지금까지는 수작업으로 분류했지만, 향후에는 수직으로 확인하는 카메라 센서 인식기만 통과하면 자동으로 분류가 가능

3

다세대 주택 등 여러 가구가 사는 곳에서 활용될 수 있는 공동 재활용 분리기 모델을 통해 쉽게 분류가 가능하고 재활용률의 상승 기대

4

기업이나 민영 기관 혹은 공공기관 등에서 선별 분류 과정 간 인건비 감소로 재정 감축 기대 가능 & 분류 공장의 가동률을 증가시켜 전반적인 생산성의 효율 증대

개선 방안

1 다양성 관점 - class 추가

- _다양한 쓰레기를 분류할 수 있도록 새로운 Feature를 추가함
- _비닐, 스티로폼 등

2 정확도 관점 - data 수 증가

- _훈련데이터를 대폭 증가시켜 모델이 구분해내지 못한 사진들에 대한 데이터를 보강
- _(흰 종이 → 흰색 때문에 플라스틱으로 인식, 흰 유리컵 → 투명한 색 때문에 플라스틱으로 인식)

3 범용적 관점 - 다양한 모델을 사용

- _Custom CNN뿐 아니라 전이학습모델(VGG16 ,ResNet50 etc..)로 더욱 정교하고 다양한 관점에서 사용 가능

Four step

PROJECT REVIEW

비전있는 비전공자

진세용 (팀장)



저희 모델의 성능은 약 88%정도의 수준을 보입니다.
90%의 수치가 안되는 것으로 보았을 때 낮은 게 아니냐고 할 수 있겠지만,
구글 이미지를 통해 크롤링 데이터만을 기반으로 모델을 구성하고
실생활에서 사용하는 것들을 사진으로 구분해 내는 수준이
88%라는 것을 생각하였을 때 충분한 활용가치가 있다고 생각합니다.
또한 실생활에서 사용하는 주요 Class에 대한 데이터 보강이 이루어진다면
90%이상인 수준의 판별능력이 있을 것으로 예상됩니다.



김성훈 (팀원)

머신러닝에 대해 많이 기대하고 어느 정도의 분류를 해낼 지 호기심을 가지고 시작한 프로젝트지만
생각보다 한계가 명확함을 느꼈고, 더 좋은 성능을 이끌어내는 것이 얼마나 어려운 것인지 체감하였습니다.
이번 프로젝트를 진행하면서 이러한 부분을 알아간다는 것이 큰 도움이 되었고,
더 공부해서 좋은 성능을 이끌어 낼 수 있도록 노력하겠습니다.

김소연 (팀원)

인공지능이나 머신러닝이라는 개념을 이번 교육과정에 참여하면서 처음 접해보았는데,
3개월간 배운 것들을 활용하여 2주간 짧은 시간이었지만 무언가 결과를 도출했다는 것이 뿌듯했습니다.
모델을 구축하면서 다양한 변수들을 복합적으로 생각해야 한다는 것이 어려웠지만
최종적으로 최적의 모델을 선택하고 모델에 새로운 데이터를 넣어보았을 때
예측률이 약간 낮더라도 분류가 된다는 점이 신기했습니다.

문충헌 (팀원)

저희 조에서 구상한 재활용분류모델이 현재 시점에서 얼마나 정밀한지를 떠나,
카메라 센서를 통해 시각적이고 직관적으로 손쉽게,
누구나 분류가 가능하다는 것이 타분류모델과는 차별점이라고 생각하며,
만약 정밀성, 접근성 등등이 더 보완되고 개선되어 상용화가 된다면
해당 모델을 통한 서비스를 제공하는 입장이든
직접 서비스를 제공받는 입장이든 서로 만족스러울 것으로 생각합니다.

조영훈 (팀원)



머신러닝을 통해 이미지를 정확히 분류하는 일이 생각보다 어렵다는 사실을 알게 되었습니다.
상대적으로 데이터가 적은 Category의 예측률이 낮게 나왔는데,
데이터의 양이 더 많았으면 어땠을까 하는 아쉬움이 있었습니다.
개선방안을 적용하여 다음에는 더욱 예측률이 높은 모델을 만들어보고 싶습니다.



김지찬 (팀원)

지난 3개월 동안 가장 크게 배웠던 건 머신러닝, 인공지능에 대한 이해도 보다는
어떻게 이 분야를 공부하고 도전하는지에 대해서 배운 것 같습니다.

이번 세미 프로젝트에서는 주제선정부터 모델을 구축하는 것 까지
많은 시행착오 끝에 작동하는 것을 보고 조금 더 좋은 모델을 구상을 해보고 싶다는 생각이 들었습니다.

세미 프로젝트에서 큰 결과물을 내놓기에는 어려운 점이 있지만,
이 경험이 토대가 되어서 우리 팀원들 뿐만 아니라 H조 모두가 큰 결과를 만들어 낼 수 있는 사람이 되면 좋겠습니다.

THANK YOU

끝 까 지 보 주 셔 서 감 사 합 니 다



비전있는 비전공자

QUESTION