전병준

농산물 분류 모델

데이터 구성

traindata

7개의 클래스, 토마토를 제외한 나머지클래스는
약 2000개의 데이터, 총 12500개

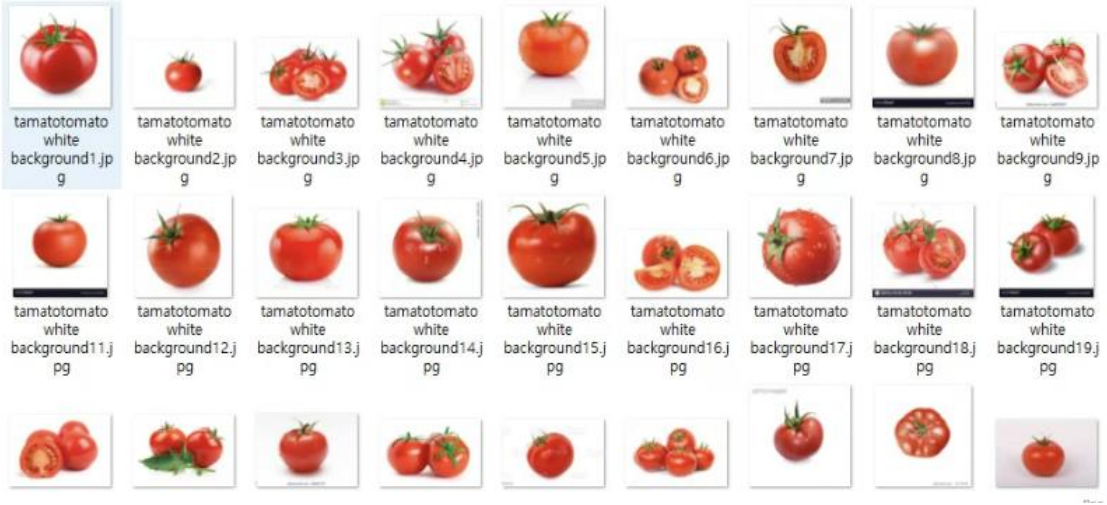
testdata

7개의 클래스, 토마토를 제외한 나머지클래스는
약 250개의 데이터, 총 1600개

predicdata

Predic data는 1개

토마토 데이터



웹 이미지 크롤링을 이용하여 train데이터 1000개 test데이터 100개 사용

데이터 전처리 과정

```
train_datagen = ImageDataGenerator(rescale=1./255,fill_mode='constant')
test_datagen = ImageDataGenerator(rescale=1./255,fill_mode='constant')

xy_train = train_datagen.flow_from_directory('../_data/farm/train',target_size=(150,150),batch_size=12554
,shuffle=False,class_mode='categorical')

xy_test = test_datagen.flow_from_directory('../_data/farm/test',target_size=(150,150),batch_size=1600
,shuffle=False,class_mode='categorical')
```

```
x_predic = test_datagen.flow_from_directory('../_data/farm/predict',target_size=(150,150),
shuffle=False,class_mode='categorical')
```

```
np.save('./_npy/x_train.npy',arr=xy_train[0][0])
np.save('./_npy/y_train.npy',arr=xy_train[0][1])
np.save('./_npy/x_test.npy',arr=xy_test[0][0])
np.save('./_npy/y_test.npy',arr=xy_test[0][1])
```

모델링 구성

```
model = Sequential()
model.add(Conv2D(64,(2,2),padding='same',activation='relu',input_shape=(150,150,3)))
model.add(MaxPool2D(2,2))
model.add(Conv2D(32,(2,2),padding='same',activation='relu'))
model.add(MaxPool2D(2,2))
model.add(Conv2D(16,(2,2),padding='same',activation='relu'))
model.add(MaxPool2D(2,2))
model.add(Flatten())
model.add(Dense(32,activation='relu'))
model.add(Dense(16,activation='relu'))
model.add(Dense(7,activation='softmax'))
```

훈련/컴파일

```
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
es = EarlyStopping(monitor='val_acc', patience=10, mode='max', verbose=1)

starttime = time.time()
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['acc'])
hist = model.fit(x_train,y_train,epochs=100,batch_size=128, validation_data=(x_test,y_test),callbacks=[es])
```

데이터 시각화

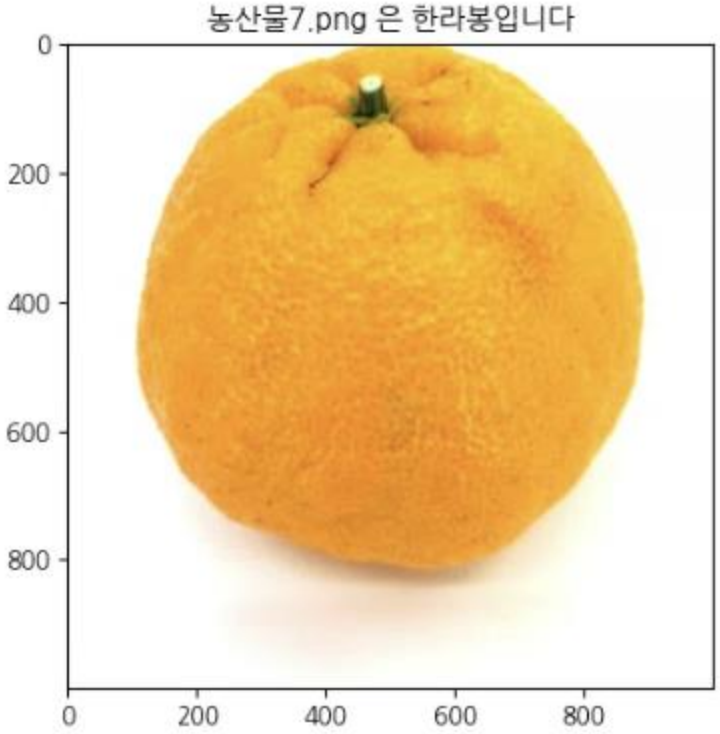
```
class_pred = ''
for i in y_predic:
        max_temp = i.argmax()
        if max_temp == 0 : class_pred = '사과'
        elif max_temp == 1 : class_pred = '한라봉'
        elif max_temp == 2 : class_pred = '감귤'
        elif max_temp == 3 : class_pred = '적양파'
        elif max_temp == 4 : class_pred = '양파'
        elif max_temp == 5 : class_pred = '감자'
        elif max_temp == 6 : class_pred = '토마토'
```

가장 큰 배열의 위치에 따라서 예측되는 클래스 값을 넣어준다.

데이터 시각화

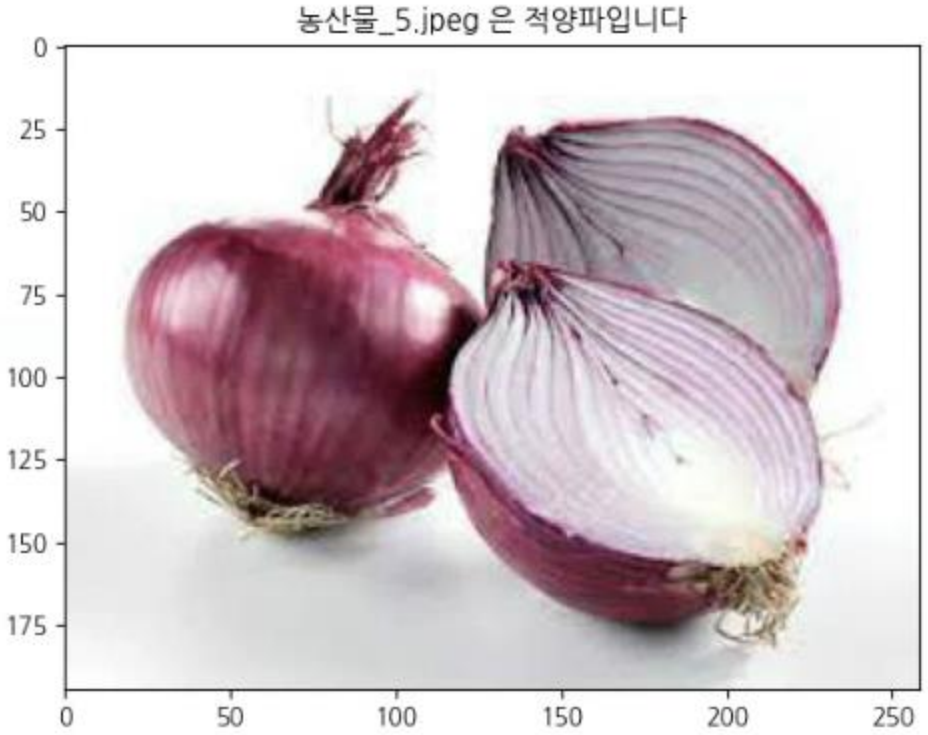
```
filename = []
predic_dir = '../_data/farm/predict'
files = glob.glob(predic_dir + "/*/*.*")
print(files)
#['../_data/farm/predict\\x_predict\\농산물6.jpeg']
for i,f in enumerate(files):
    filename.append(f)

name = filename[0].split('\\')[2] # 문자를 나누는 기호
path = '../_data/farm/predict/x_predict/'+name
predic_img = img.imread(path,1) # color

print("걸린시간", end)
print('acc는 : ',acc[-1],'입니다')
print(name,'는 : ',round(val_acc[-1],3) * 100,'%의 확률로 ',class_pred,'입니다')
plt.imshow(predic_img)
plt.title(name+' 은 '+class_pred+'입니다')
plt.show()
```

최종 결과 한 개




```
acc는 :  1.0 입니다
농산물7.png 는 :  90.0 %의 확률로  한라봉 입니다
```

최종 결과 여러 개




```
농산물_5.jpeg 는 :  95.0 %의 확률로  적양파 입니다
```

부족한 점

1. 기존의 목표였던 품질의 따른 분류를 하지 못한 점

2. 사과와 토마토의 분류를 명확하게 하지 못하였던 점

3. 예측값들을 여러 개를 처리하지 못하고 하나씩 예측한 점

4. 단순하게 수치에만 집중해 버린것

Q&A

감사합니다!!