

AI기반 농작물 질병 진단

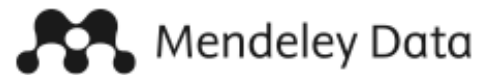
프로젝트 개요

- 위의 이미지로 위의 종류와 질병의 상태를 레이블한 데이터셋을 가지고 39종의 농작물 질병을 진단하는 모델을 학습해 봅니다.
- 농작물 질병 진단을 위해 이미지를 분류하는데 필요한 모델과 데이터 증식 방법을 알아봅니다.
- 이 분류 모델을 통해서 위의 종류에 따른 질병을 분류합니다.

01. 데이터셋

데이터셋 다운 받기
탐색적 데이터 분석
데이터 증식

데이터셋 다운 받기



Data for: Identification of Plant Leaf Diseases Using a 9-layer Deep Convolutional Neural Network

Published: 18 April 2019 | Version 1 | DOI: 10.17632/tywbtsjrjv.1

Contributors: ARUN PANDIAN J, GEETHARAMANI GOPAL

Description

In this data-set, 39 different classes of plant leaf and background images are available. The data-set containing 61,486 images. We used six different augmentation techniques for increasing the data-set size. The techniques are image flipping, Gamma correction, noise injection, PCA color augmentation, rotation, and Scaling.

The classes are,

- 1.Apple_scab
- 2.Apple_black_rot
- 3.Apple_cedar_apple_rust
- 4.Apple_healthy

데이터셋은 아래 링크를 통해 다운로드 받을 수 있습니다.
<https://data.Mendeley.com/datasets/tywbtsjrjv/1>

데이터셋 다운 받기

36.Tomato_spider_mites_two-spotted_spider_mite
37.Tomato_target_spot
38.Tomato_mosaic_virus
39.Tomato_yellow_leaf_curl_virus

Download All 1708 MB



Files



Plant_leaf_diseases_dataset_with_augmentation.zip

905 MB



Plant_leaf_diseases_dataset_without_augmentation.zip







828 MB

데이터 증식을 적용하지 않은 Plant_leaf_diseases_dataset_without_augmentation.zip파일을 사용합니다.

데이터셋은 아래 링크를 통해 다운로드 받을 수 있습니다.

<https://data.Mendeley.com/datasets/tywbtsjrjv/1>

탐색적 데이터 분석 (EDA)

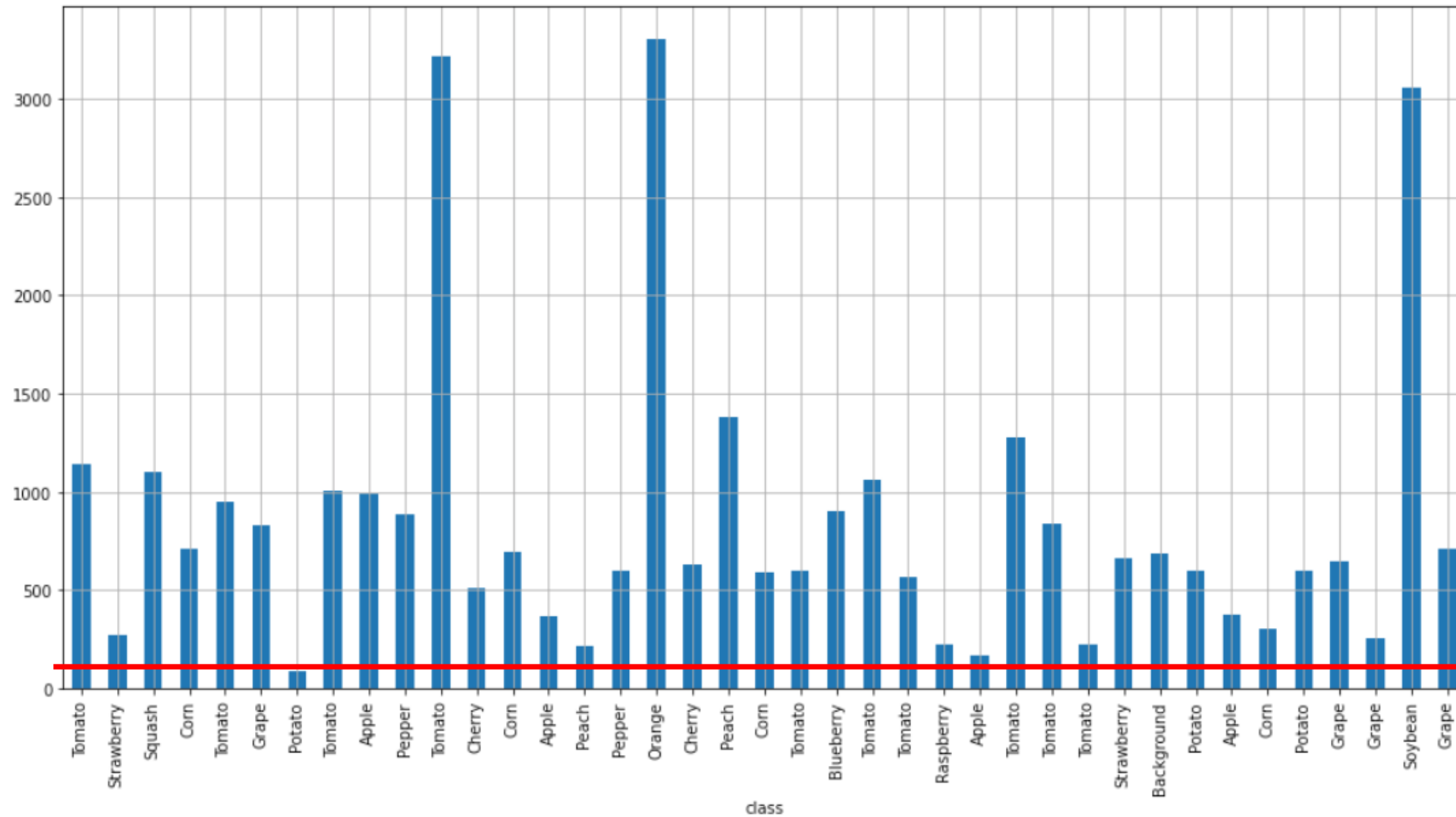
Tomato					
					
healthy	Bacterial_spot	Early_blight	Late_blight	Leaf Mold	Septoria_leaf_spot

이 데이터셋은 식물의 잎에서 발생하는 다양한 작물 질병에 대한 이미지를 클래스별로 분류하여 레이블링 하였습니다.

식물의 종류 : Apple(4), Blueberry(1), Cherry(2), Corn(4), Grape(4), Orange(1),
Peach(2), Pepper(2), Potato(3), Raspberry(1), Soybean(1),
squash(1), strawberry(2), Tomato(10), background(1) (총 14종)

잎의 상태수 : 39개

탐색적 데이터 분석 (EDA)



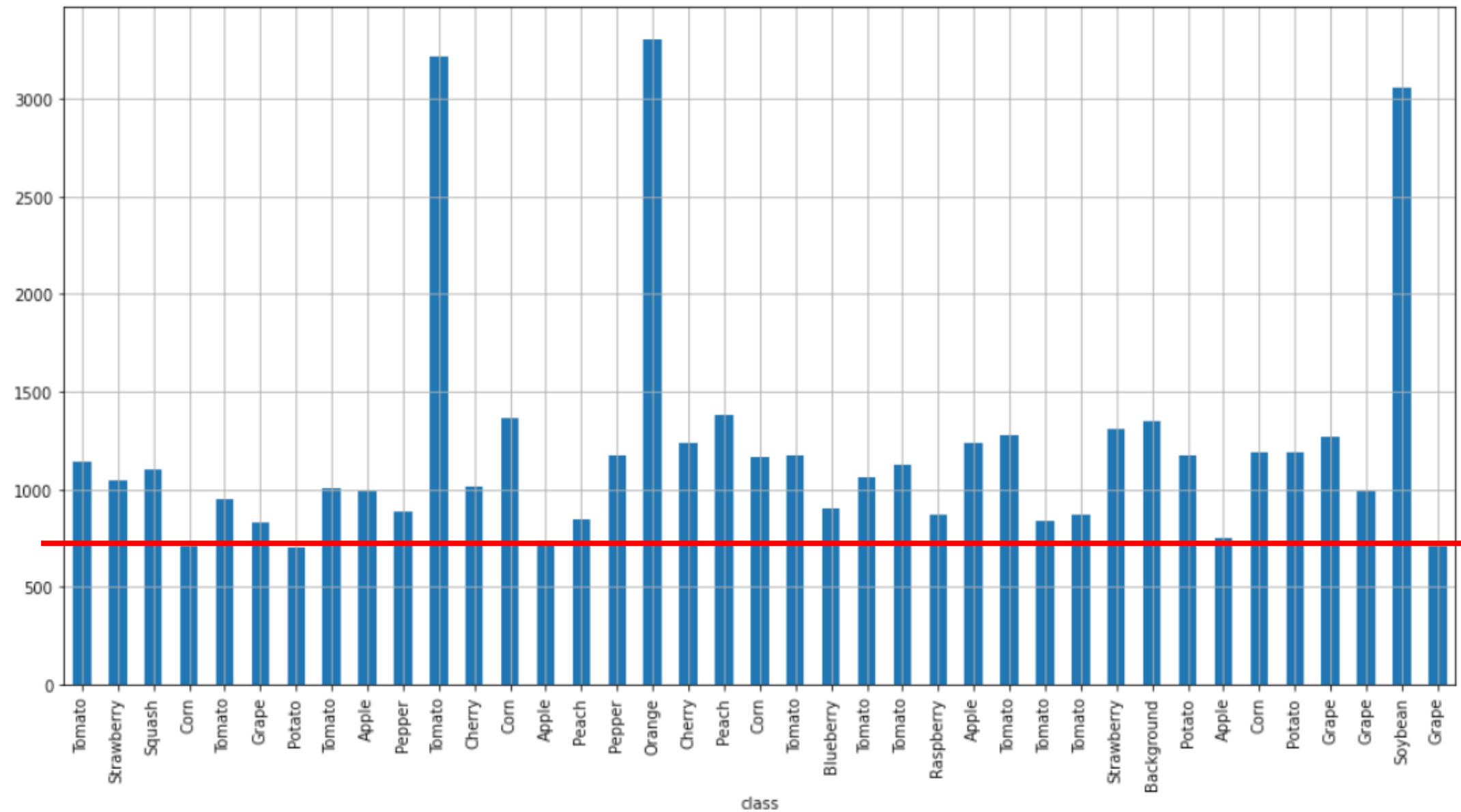
훈련 데이터의 수가 500개에도 미치지 못하는 클래스가 많습니다.

데이터 증식

```
[ ] train_datagen = ImageDataGenerator(rescale = 1./255,  
                                       horizontal_flip=True,  
                                       vertical_flip=True,  
                                       zoom_range=0.2)
```

```
[ ] trainPath = './dataset_split_raw/train/'  
   class_list = os.listdir(trainPath)  
  
   for className in class_list:  
       if len(os.listdir(trainPath + className))<600:  
           imgPath = os.path.join(trainPath, className)  
           images = os.listdir(imgPath)  
  
           for image in images:  
               image = load_img(os.path.join(imgPath, image))  
               input_arr = img_to_array(image)  
               input_arr = np.array([input_arr]) # Convert single image to a batch.  
  
               i = 0  
               for batch in train_datagen.flow(input_arr, batch_size=1, save_to_dir = imgPath, save_prefix = 'aug', save_format='jpg'):  
                   i+=1  
                   if i>0:  
                       break
```


탐색적 데이터 분석 (EDA)



훈련 데이터의 수가 최소 700개 이상 되도록 데이터를 증식하였습니다.

02.

딥러닝 모델

CNN을 직접 구성하여 사용
전이 학습 모델 사용

CNN을 직접 구성하여 학습

```
# baseline 모델 구조 정의
model1 = Sequential()
model1.add(Conv2D(32, (3, 3), input_shape=(imgSize,imgSize,channel), padding='same', activation= 'relu'))
model1.add(MaxPooling2D(pool_size=(2, 2)))
model1.add(Dropout(0.25))

model1.add(Conv2D(64, (3, 3), padding='same', activation= 'relu'))
model1.add(MaxPooling2D(pool_size=(2, 2)))
model1.add(Dropout(0.25))

model1.add(Conv2D(64, (3, 3), padding = 'same', activation= 'relu'))
model1.add(MaxPooling2D(pool_size=(2, 2)))
model1.add(Dropout(0.25))

# 전결합층
model1.add(Flatten()) # 벡터형태로 reshape
model1.add(Dense(512, activation='relu'))
model1.add(Dropout(0.5))
model1.add(Dense(classNum, activation='softmax'))

# 모델 구축하기
optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=0.1, decay=0.0) # 최적화 함수 지정

model1.compile(loss='categorical_crossentropy',
               optimizer=optimizer,
               metrics=['acc'])
```

CNN을 직접 구성하여 학습

데이터 증식

```
train_datagen = ImageDataGenerator(rescale = 1./255,  
                                   |             horizontal_flip=True,  
                                   |             vertical_flip=True,  
                                   |             zoom_range=0.2)  
test_datagen = ImageDataGenerator(rescale=1./255)  
  
train_generator = train_datagen.flow_from_directory(train_dir,  
                                                    target_size=(imgSize, imgSize),  
                                                    batch_size=batch_size,  
                                                    class_mode='categorical')  
  
validation_generator = test_datagen.flow_from_directory(validation_dir,  
                                                        target_size=(imgSize, imgSize),  
                                                        batch_size=batch_size,  
                                                        class_mode='categorical')
```

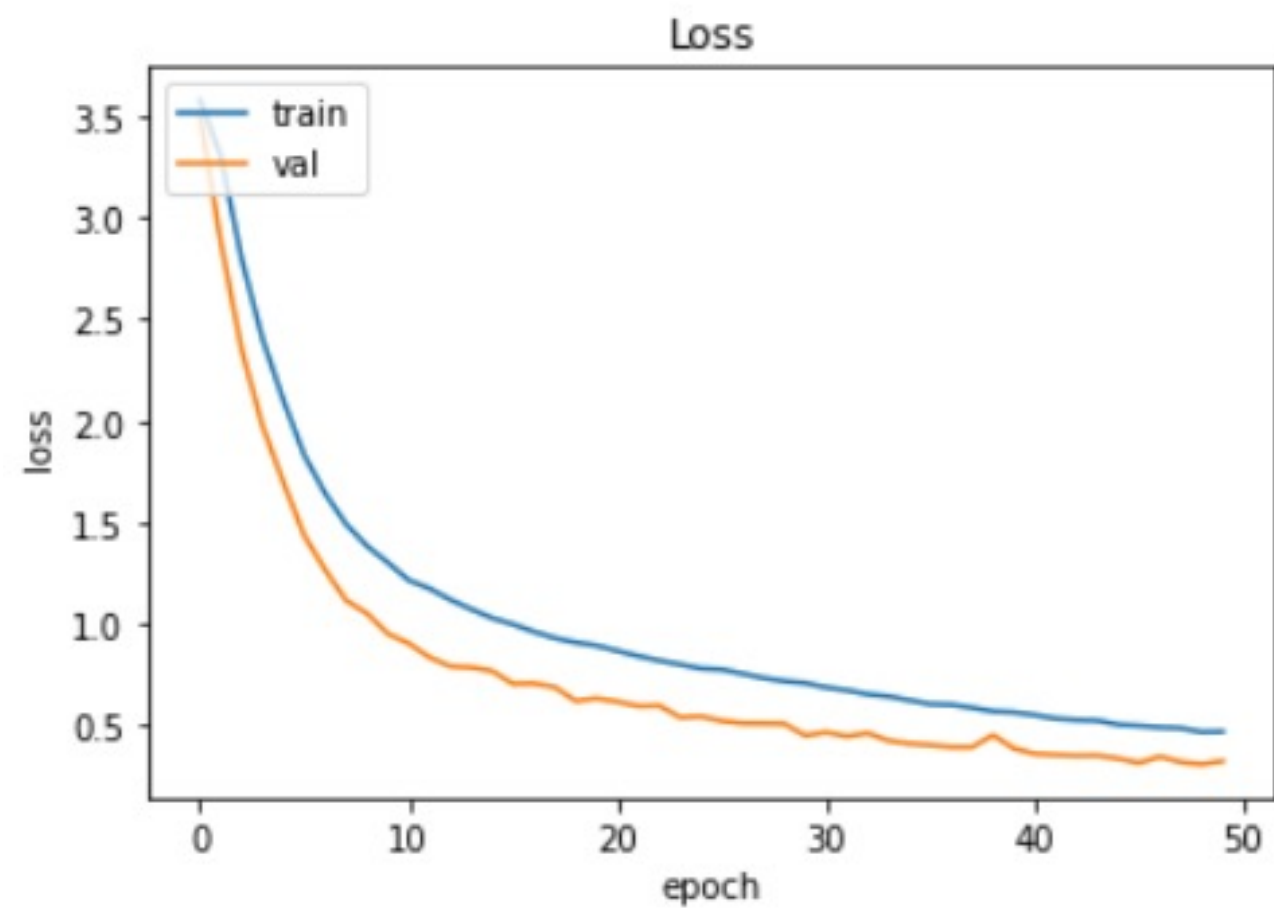
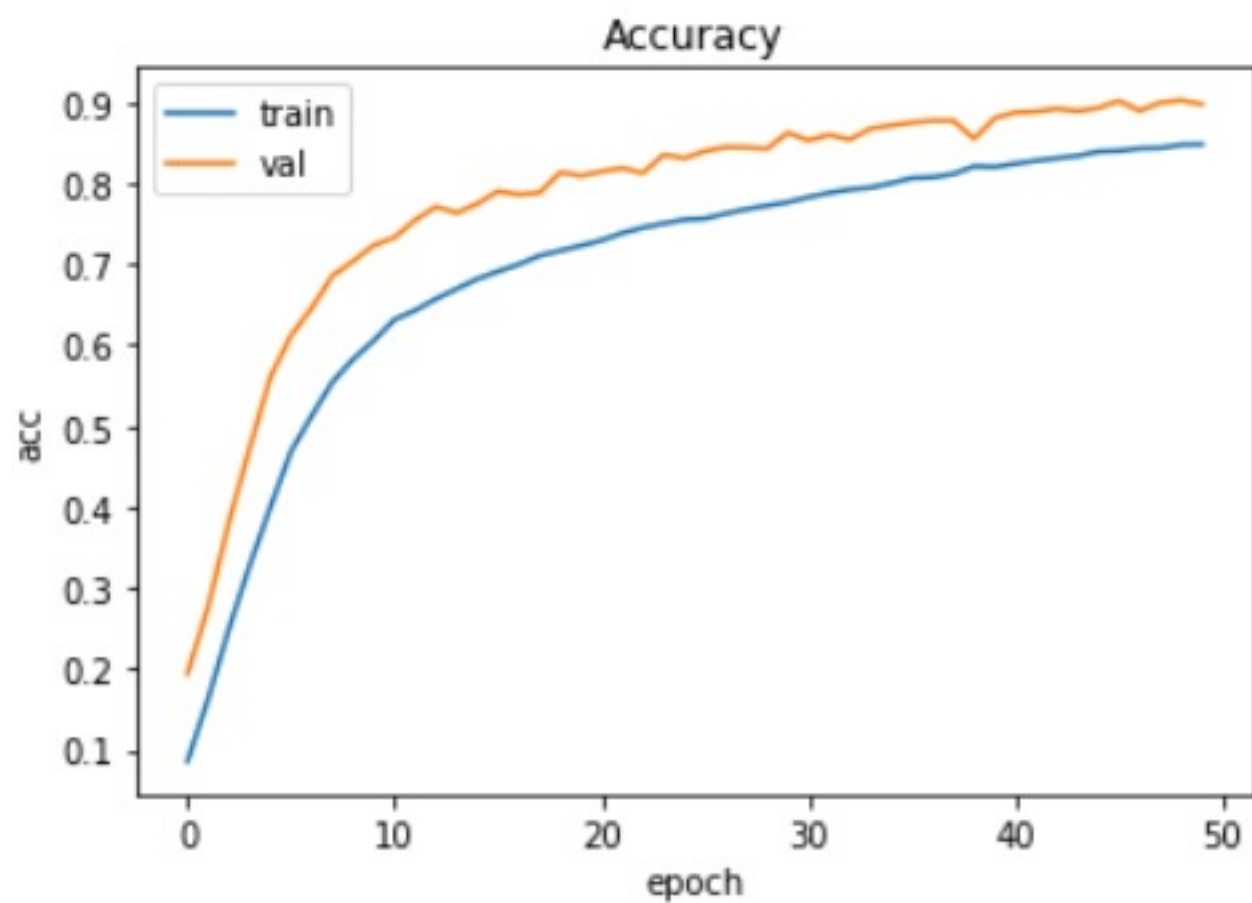
훈련 데이터에만 데이터 증식 조건을 설정합니다.

CNN을 직접 구성하여 학습

콜백함수 설정

```
# val_acc를 기준으로 val_acc가 가장 높을때 모델을 저장한다.  
mc = ModelCheckpoint('simpleCNN_best.h5', monitor='val_acc', mode='auto', verbose=1, save_best_only=True)  
  
# val_acc를 기준으로 10 epoch동안 더 높은 val_acc가 나타나지 않을 경우 학습을 종료한다.  
es = EarlyStopping(monitor='val_acc', patience=10)  
  
# 'val_loss'가 5epoch동안 감소하지 않으면 learningRate를 0.5(절반)로 줄인다. 그런데 최소값은 min_lr만큼까지 줄인다.  
rlr = ReduceLROnPlateau(monitor='val_acc', patience=5, factor=0.5, min_lr=0.0001)  
  
callBack = [mc, es, rlr]
```

CNN을 직접 구성하여 학습



전이 학습 모델을 사용

```
from tensorflow.keras.applications import ResNet50

input_tensor = Input(shape=(imgSize, imgSize, channel))
conv_base = ResNet50(include_top=False, weights='imagenet', input_tensor=input_tensor, pooling='max')
conv_base.trainable = True
```

conv5_block3_add (Add)	(None, 4, 4, 2048)	0	['conv5_block2_out[0][0]', 'conv5_block3_3_bn[0][0]']
conv5_block3_out (Activation)	(None, 4, 4, 2048)	0	['conv5_block3_add[0][0]']
max_pool (GlobalMaxPooling2D)	(None, 2048)	0	['conv5_block3_out[0][0]']

```
=====
Total params: 23,587,712
Trainable params: 23,534,592
Non-trainable params: 53,120
=====
```


전이 학습 모델을 사용

```
x = conv_base.output
output = Dense(classNum, activation='softmax')(x)
model = Model(input_tensor, output)
optimizer = Adam(learning_rate=LR, beta_1=0.9, beta_2=0.999, epsilon=0.1, decay=0.0) # 최적화 함수 지정
model.compile(loss='categorical_crossentropy', optimizer=optimizer, metrics=['acc'])
```

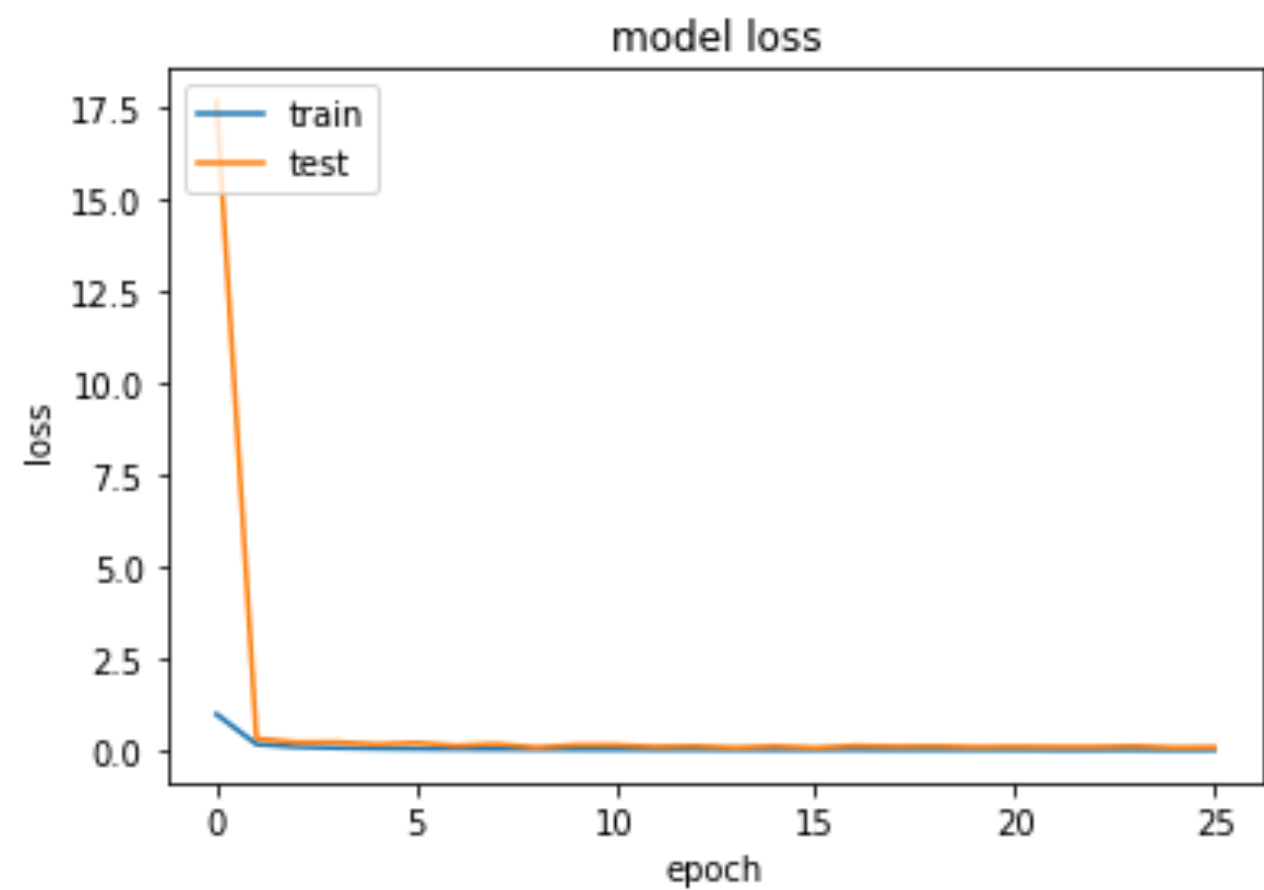
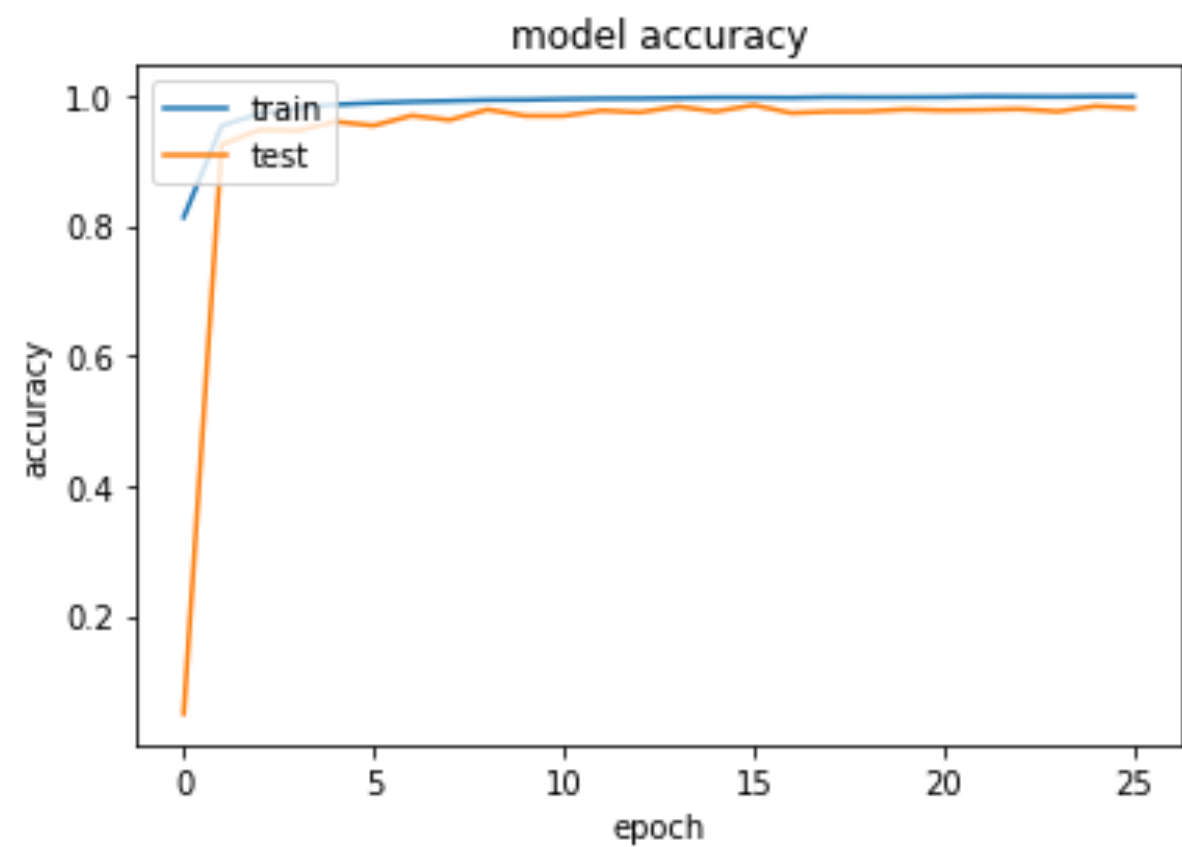
CNN을 직접 구성하여 난수에서 부터 파라미터를 학습하는 것보다 학습에 소요되는 시간이 적게 듭니다. 또한 커스텀 데이터셋의 크기가 상대적으로 적더라도 더 좋은 성능을 보입니다.

전이 학습 모델을 사용

```
imgSize = 128  
channel = 3  
batch_size = 64  
classNum = 39  
epochs = 50  
classNum = 39  
LR = 0.001
```

원본 이미지의 크기는 256x256이다.
CNN모델(ResNet50)에 입력시
128x128로 이미지를 조금 크게 해서
학습시 성능을 높이는데 도움이 됩니다.
대신 학습에 많은 시간이 소요됩니다.

전이 학습 모델을 사용



지금까지 AI기반 농작물 질병 진단 프로젝트
구현 방법에 대해 알아보았습니다.

감사합니다.