20장. 전이학습

#전이학습 : 기존의 이미지에서 학습한 정보를 가져와 내 프로젝트에 활용하는 것.

=기존의 값을 가져옴

```
train_datagen = ImageDataGenerator(rescale=1./255,
                                   horizontal_flip=True,    #수평 대칭 이미지를 50% 확률로 만들어 줍니다.
                                   width_shift_range=0.1,  #전체 크기의 10% 범위에서 좌우로 이동합니다.
                                   height_shift_range=0.1, #마찬가지로 위, 아래로 이동
                                   #rotation_range=5,
                                   #shear_range=0.7,
                                   #zoom_range=[0.9, 2.2],
                                   #vertical_flip=True,
                                   fill_mode='nearest')
```

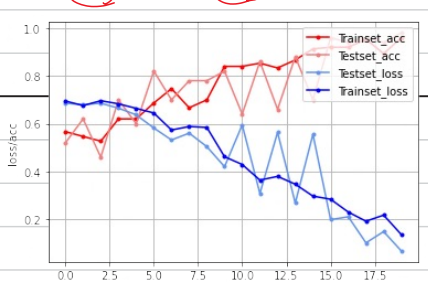
```
train_generator = train_datagen.flow_from_directory(
        '/content/drive/MyDrive/Colab Notebooks/train',
        target_size=(150, 150),
        batch_size=5,
        class_mode='binary')

#테스트 셋은 이미지 부풀리기 과정을 진행하지 않습니다.
test_datagen = ImageDataGenerator(rescale=1./255)

test_generator = test_datagen.flow_from_directory(
        '/content/drive/MyDrive/Colab Notebooks/test',
        target_size=(150, 150),
        batch_size=5,
        class_mode='binary')
```

```
#모델을 실행합니다
history = model.fit_generator(
        train_generator,
        steps_per_epoch=30,  #적절한 값으로 조절합니다.
        epochs=20,    #적절한 값으로 조절합니다.
        validation_data=test_generator,
        validation_steps=10)
```

```
Epoch 17/20
30/30 [==============================] - 2s 51ms/step - loss: 0.2843 -
accuracy: 0.8931 - val_loss: 0.2102 - val_accuracy: 0.9400
Epoch 18/20
30/30 [==============================] - 2s 52ms/step - loss: 0.2112 -
accuracy: 0.9509 - val_loss: 0.1012 - val_accuracy: 0.9600
Epoch 19/20
30/30 [==============================] - 2s 52ms/step - loss: 0.2434 -
accuracy: 0.8862 - val_loss: 0.1484 - val_accuracy: 0.9400
Epoch 20/20
30/30 [==============================] - 2s 51ms/step - loss: 0.1357 -
accuracy: 0.9680 - val_loss: 0.0672 - val_accuracy: 0.9800
```



- 모델 성능 극대화하기

기존모델사용 ⇒ '형태'를 구분하는 기본적인 학습이 되어있음. ⇒ VGG16 사용

```
transfer_model = VGG16(weights='imagenet', include_top=False, input_shape=(150, 150, 3))
transfer_model.trainable = False
transfer_model.summary()
```

그이름의 인식모델

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 150, 150, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 150, 150, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 150, 150, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 75, 75, 64)        0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 75, 75, 128)       73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 75, 75, 128)       147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 37, 37, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 37, 37, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 37, 37, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 37, 37, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 18, 18, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 18, 18, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 18, 18, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 18, 18, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 9, 9, 512)         0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 9, 9, 512)         2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 9, 9, 512)         2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 9, 9, 512)         2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 4, 4, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 0
Non-trainable params: 14,714,688
```

학습가능한 파라미터가 없음

-모델 생성

```
finetune_model = models.Sequential()
finetune_model.add(transfer_model)
finetune_model.add(Flatten())
finetune_model.add(Dense(64, activation='relu'))
finetune_model.add(Dense(2, activation='softmax'))
finetune_model.summary()
```

위 모델

레이어 층 추가

```
Model: "sequential_10"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 4, 4, 512)         14714688
_________________________________________________________________
flatten_10 (Flatten)         (None, 8192)              0
_________________________________________________________________
dense_20 (Dense)             (None, 64)                524352
_________________________________________________________________
dense_21 (Dense)             (None, 2)                 130
=================================================================
Total params: 15,239,170
Trainable params: 524,482
Non-trainable params: 14,714,688
```

(결과)

```
15/20
30/30 [==============================] - 3s 88ms/step - loss: 0.1066 -
accuracy: 0.9653 - val_loss: 0.1640 - val_accuracy: 0.9400
Epoch 16/20
30/30 [==============================] - 3s 89ms/step - loss: 0.0892 -
accuracy: 0.9907 - val_loss: 0.0857 - val_accuracy: 0.9600
Epoch 17/20
30/30 [==============================] - 3s 88ms/step - loss: 0.1057 -
accuracy: 0.9510 - val_loss: 0.1231 - val_accuracy: 0.9600
Epoch 18/20
30/30 [==============================] - 3s 88ms/step - loss: 0.0846 -
accuracy: 0.9783 - val_loss: 0.0943 - val_accuracy: 0.9600
Epoch 19/20
30/30 [==============================] - 3s 88ms/step - loss: 0.1164 -
accuracy: 0.9396 - val_loss: 0.1559 - val_accuracy: 0.9400
Epoch 20/20
30/30 [==============================] - 3s 88ms/step - loss: 0.1039 -
accuracy: 0.9422 - val_loss: 0.1491 - val_accuracy: 0.9400
```

Q. 평균정확도 구하는 법... ?