이름 : 강시온

학번 : 201802045

학과 : 컴퓨터공학과

코드 설명

```
model = Sequential([
  layers.experimental.preprocessing.RandomFlip("horizontal", input_shape=input_shape),
  layers.experimental.preprocessing.RandomRotation(0.1),
  layers.experimental.preprocessing.RandomZoom(0.1),  layers.experimental.preprocessing.Rescaling(1./255),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(128, 3, padding='same', activation='relu'),
  layers.Conv2D(128, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(256, 3, padding='same', activation='relu'),
  layers.Conv2D(256, 3, padding='same', activation='relu'),
  layers.Conv2D(256, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(512, 3, padding='same', activation='relu'),
  layers.Conv2D(512, 3, padding='same', activation='relu'),
  layers.Conv2D(512, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(512, 3, padding='same', activation='relu'),
  layers.Conv2D(512, 3, padding='same', activation='relu'),
  layers.Conv2D(512, 3, padding='same', activation='relu'),
  layers.Flatten(),
  layers.Dense(4096, activation='relu'),
  layers.Dense(4096, activation='relu'),
  layers.Dense(num_classes,activation='softmax')
])
```

실습자료에 첨부되어있는 vgg16 모델 그림을 참고하여 conv2D 레이어와 Pool레이어를 각 크기에 맞춰서 추가해주었다.

그리고 마지막에 분류할 num_classes의 개수에 맞춰 softmax로 확률값으로 만들어서 예측하는 모델을 만들었다.

결과 이미지

```
epochs = 15
history = model.fit(
  train_ds,
  validation_data=val_ds,
  epochs=epochs
)
```

```
Epoch 1/15
/usr/local/lib/python3.7/dist-packages/tensorflow/python/util/dispatch.py:1096: UserWarning: "`sparse_categorical_crossentropy` receive
  return dispatch_target(*args, **kwargs)
92/92 [==============================] - 132s 1s/step - loss: 1.6364 - accuracy: 0.2319 - val_loss: 1.6049 - val_accuracy: 0.2398
Epoch 2/15
92/92 [==============================] - 100s 1s/step - loss: 1.6021 - accuracy: 0.2459 - val_loss: 1.6026 - val_accuracy: 0.2398
Epoch 3/15
92/92 [==============================] - 99s 1s/step - loss: 1.6013 - accuracy: 0.2459 - val_loss: 1.6027 - val_accuracy: 0.2398
Epoch 4/15
92/92 [==============================] - 100s 1s/step - loss: 1.6016 - accuracy: 0.2459 - val_loss: 1.6031 - val_accuracy: 0.2398
Epoch 5/15
92/92 [==============================] - 100s 1s/step - loss: 1.6011 - accuracy: 0.2459 - val_loss: 1.6027 - val_accuracy: 0.2398
Epoch 6/15
92/92 [==============================] - 99s 1s/step - loss: 1.6009 - accuracy: 0.2459 - val_loss: 1.6025 - val_accuracy: 0.2398
Epoch 7/15
92/92 [==============================] - 99s 1s/step - loss: 1.6011 - accuracy: 0.2459 - val_loss: 1.6022 - val_accuracy: 0.2398
Epoch 8/15
92/92 [==============================] - 99s 1s/step - loss: 1.6012 - accuracy: 0.2459 - val_loss: 1.6026 - val_accuracy: 0.2398
Epoch 9/15
92/92 [==============================] - 99s 1s/step - loss: 1.6013 - accuracy: 0.2459 - val_loss: 1.6021 - val_accuracy: 0.2398
Epoch 10/15
92/92 [==============================] - 99s 1s/step - loss: 1.6010 - accuracy: 0.2459 - val_loss: 1.6027 - val_accuracy: 0.2398
Epoch 11/15
92/92 [==============================] - 99s 1s/step - loss: 1.6011 - accuracy: 0.2459 - val_loss: 1.6019 - val_accuracy: 0.2398
Epoch 12/15
92/92 [==============================] - 99s 1s/step - loss: 1.6009 - accuracy: 0.2459 - val_loss: 1.6022 - val_accuracy: 0.2398
Epoch 13/15
92/92 [==============================] - 100s 1s/step - loss: 1.6008 - accuracy: 0.2459 - val_loss: 1.6021 - val_accuracy: 0.2398
Epoch 14/15
92/92 [==============================] - 99s 1s/step - loss: 1.6013 - accuracy: 0.2459 - val_loss: 1.6027 - val_accuracy: 0.2398
Epoch 15/15
92/92 [==============================] - 100s 1s/step - loss: 1.6013 - accuracy: 0.2459 - val_loss: 1.6026 - val_accuracy: 0.2398
```

난이도

기계학습에서 배우는 내용들을 활용하여 해볼 수 있는 과제여서 더욱 좋은 과제였습니다.