컴퓨터 비전

과제 #5

담당교수 : 김낙현 교수님

제출일자 : 21.05.21

학 과 : 전자공학과

학 년 : 4학년

이 름 : 배준성

학 번 : 201501487

한국외국어대학교
HANKUK UNIVERSITY OF FOREIGN STUDIES

1. Epoch=20으로 설정하여 정확도를 측정

실행 결과

```
Epoch 1/20
704/704 [==============================] - 80s 112ms/step - loss: 1.8193 - accuracy: 0.3310 - val_loss: 1.3253 - val_accuracy: 0.5120
Epoch 2/20
704/704 [==============================] - 75s 107ms/step - loss: 1.2270 - accuracy: 0.5639 - val_loss: 1.1095 - val_accuracy: 0.6200
Epoch 3/20
704/704 [==============================] - 76s 108ms/step - loss: 1.0258 - accuracy: 0.6410 - val_loss: 0.9404 - val_accuracy: 0.6732
Epoch 4/20
704/704 [==============================] - 77s 109ms/step - loss: 0.8935 - accuracy: 0.6877 - val_loss: 0.8773 - val_accuracy: 0.7004
Epoch 5/20
704/704 [==============================] - 76s 108ms/step - loss: 0.8306 - accuracy: 0.7143 - val_loss: 0.8443 - val_accuracy: 0.7160
Epoch 6/20
704/704 [==============================] - 78s 110ms/step - loss: 0.7546 - accuracy: 0.7396 - val_loss: 0.8401 - val_accuracy: 0.7150
Epoch 7/20
704/704 [==============================] - 77s 109ms/step - loss: 0.6928 - accuracy: 0.7585 - val_loss: 0.8027 - val_accuracy: 0.7306
Epoch 8/20
704/704 [==============================] - 79s 112ms/step - loss: 0.6564 - accuracy: 0.7712 - val_loss: 0.8084 - val_accuracy: 0.7236
Epoch 9/20
704/704 [==============================] - 75s 106ms/step - loss: 0.6173 - accuracy: 0.7865 - val_loss: 0.8117 - val_accuracy: 0.7238
Epoch 10/20
704/704 [==============================] - 75s 107ms/step - loss: 0.5701 - accuracy: 0.8030 - val_loss: 0.8204 - val_accuracy: 0.7250
Epoch 11/20
704/704 [==============================] - 76s 108ms/step - loss: 0.5345 - accuracy: 0.8138 - val_loss: 0.7476 - val_accuracy: 0.7540
Epoch 12/20
704/704 [==============================] - 75s 107ms/step - loss: 0.4981 - accuracy: 0.8286 - val_loss: 0.7648 - val_accuracy: 0.7512
Epoch 13/20
704/704 [==============================] - 74s 105ms/step - loss: 0.4680 - accuracy: 0.8362 - val_loss: 0.7826 - val_accuracy: 0.7502
Epoch 14/20
704/704 [==============================] - 75s 106ms/step - loss: 0.4443 - accuracy: 0.8428 - val_loss: 0.8215 - val_accuracy: 0.7480
Epoch 15/20
704/704 [==============================] - 75s 106ms/step - loss: 0.4251 - accuracy: 0.8507 - val_loss: 0.8437 - val_accuracy: 0.7496
Epoch 16/20
704/704 [==============================] - 75s 106ms/step - loss: 0.3803 - accuracy: 0.8688 - val_loss: 0.8166 - val_accuracy: 0.7508
Epoch 17/20
704/704 [==============================] - 74s 106ms/step - loss: 0.3601 - accuracy: 0.8743 - val_loss: 0.9096 - val_accuracy: 0.7336
Epoch 18/20
704/704 [==============================] - 75s 107ms/step - loss: 0.3477 - accuracy: 0.8815 - val_loss: 0.9135 - val_accuracy: 0.7414
Epoch 19/20
704/704 [==============================] - 76s 108ms/step - loss: 0.3127 - accuracy: 0.8914 - val_loss: 0.9275 - val_accuracy: 0.7472
Epoch 20/20
704/704 [==============================] - 83s 118ms/step - loss: 0.3040 - accuracy: 0.8941 - val_loss: 1.1337 - val_accuracy: 0.7270
313/313 - 4s - loss: 1.1943 - accuracy: 0.7086

테스트 정확도: 0.7085999846458435
```

20회의 학습이 끝난 이후 최 정확도는 약 70.86% 임을 확인할 수 있다.

2. Dropout을 적용

신경망 코드

```
#Dropout을 적용
model = models.Sequential()

model.add(layers.Conv2D(32, (3,3), padding="SAME", activation='relu', input_shape=(32, 32, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3,3), padding="SAME", activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3,3), padding="SAME", activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dropout(rate=0.2))
#20%의 노드를 0으로 만든다.
model.add(layers.Dense(10, activation='softmax'))

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# 파라미터 수 읽기
model.summary()
```

Dropout을 적용하여 20%의 노드를 제외해 주었다.

실행 결과

```
Epoch 1/20
704/704 [==============================] - 79s 112ms/step - loss: 1.1279 - accuracy: 0.6001 - val_loss: 1.0389 - val_accuracy: 0.6398
Epoch 2/20
704/704 [==============================] - 82s 117ms/step - loss: 1.0075 - accuracy: 0.6465 - val_loss: 0.9427 - val_accuracy: 0.6690
Epoch 3/20
704/704 [==============================] - 83s 118ms/step - loss: 0.9236 - accuracy: 0.6740 - val_loss: 0.8906 - val_accuracy: 0.6974
Epoch 4/20
704/704 [==============================] - 84s 120ms/step - loss: 0.8576 - accuracy: 0.6992 - val_loss: 0.8623 - val_accuracy: 0.7036
Epoch 5/20
704/704 [==============================] - 89s 127ms/step - loss: 0.8026 - accuracy: 0.7185 - val_loss: 0.8257 - val_accuracy: 0.7088
Epoch 6/20
704/704 [==============================] - 85s 121ms/step - loss: 0.7548 - accuracy: 0.7333 - val_loss: 0.7891 - val_accuracy: 0.7278
Epoch 7/20
704/704 [==============================] - 87s 124ms/step - loss: 0.7144 - accuracy: 0.7497 - val_loss: 0.8382 - val_accuracy: 0.7166
Epoch 8/20
704/704 [==============================] - 84s 120ms/step - loss: 0.6789 - accuracy: 0.7594 - val_loss: 0.7604 - val_accuracy: 0.7432
Epoch 9/20
704/704 [==============================] - 91s 129ms/step - loss: 0.6416 - accuracy: 0.7747 - val_loss: 0.7623 - val_accuracy: 0.7402
Epoch 10/20
704/704 [==============================] - 94s 133ms/step - loss: 0.6090 - accuracy: 0.7856 - val_loss: 0.7570 - val_accuracy: 0.7540
Epoch 11/20
704/704 [==============================] - 90s 127ms/step - loss: 0.5853 - accuracy: 0.7912 - val_loss: 0.7866 - val_accuracy: 0.7478
Epoch 12/20
704/704 [==============================] - 88s 125ms/step - loss: 0.5569 - accuracy: 0.8014 - val_loss: 0.7741 - val_accuracy: 0.7456
Epoch 13/20
704/704 [==============================] - 89s 127ms/step - loss: 0.5322 - accuracy: 0.8095 - val_loss: 0.7604 - val_accuracy: 0.7538
Epoch 14/20
704/704 [==============================] - 92s 131ms/step - loss: 0.5121 - accuracy: 0.8172 - val_loss: 0.7857 - val_accuracy: 0.7496
Epoch 15/20
704/704 [==============================] - 86s 123ms/step - loss: 0.4862 - accuracy: 0.8264 - val_loss: 0.8331 - val_accuracy: 0.7412
Epoch 16/20
704/704 [==============================] - 87s 124ms/step - loss: 0.4638 - accuracy: 0.8349 - val_loss: 0.8041 - val_accuracy: 0.7528
Epoch 17/20
704/704 [==============================] - 82s 116ms/step - loss: 0.4502 - accuracy: 0.8368 - val_loss: 0.8614 - val_accuracy: 0.7368
Epoch 18/20
704/704 [==============================] - 89s 127ms/step - loss: 0.4294 - accuracy: 0.8466 - val_loss: 0.8663 - val_accuracy: 0.7486
Epoch 19/20
704/704 [==============================] - 87s 124ms/step - loss: 0.4034 - accuracy: 0.8535 - val_loss: 0.8664 - val_accuracy: 0.7538
Epoch 20/20
704/704 [==============================] - 84s 119ms/step - loss: 0.3938 - accuracy: 0.8581 - val_loss: 0.9309 - val_accuracy: 0.7368
313/313 - 6s - loss: 0.9507 - accuracy: 0.7316

테스트 정확도: 0.7315999865531921
```

Dropout을 적용하였을 경우 정확도가 73.16%로 소폭 상승하였음을 확인할 수 있었다.

3. 프로그램을 확장

신경망 코드

```
#프로그램을 확장
model = models.Sequential()

model.add(layers.Conv2D(32, (3,3), padding="SAME", activation='relu', input_shape=(32,
32, 3)))
model.add(layers.Conv2D(32, (3,3), padding="SAME", activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3,3), padding="SAME", activation='relu'))
model.add(layers.Conv2D(64, (3,3), padding="SAME", activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# 파라미터 수 읽기
model.summary()
```

실행 결과

```
Epoch 1/20
704/704 [==============================] - 191s 267ms/step - loss: 1.7095 - accuracy: 0.3719 - val_loss: 1.1424 - val_accuracy: 0.5992
Epoch 2/20
704/704 [==============================] - 187s 266ms/step - loss: 1.0435 - accuracy: 0.6316 - val_loss: 1.0168 - val_accuracy: 0.6520
Epoch 3/20
704/704 [==============================] - 206s 293ms/step - loss: 0.8211 - accuracy: 0.7116 - val_loss: 0.8047 - val_accuracy: 0.7240
Epoch 4/20
704/704 [==============================] - 180s 256ms/step - loss: 0.6806 - accuracy: 0.7624 - val_loss: 0.7303 - val_accuracy: 0.7504
Epoch 5/20
704/704 [==============================] - 179s 254ms/step - loss: 0.5824 - accuracy: 0.7962 - val_loss: 0.8198 - val_accuracy: 0.7186
Epoch 6/20
704/704 [==============================] - 178s 253ms/step - loss: 0.5024 - accuracy: 0.8231 - val_loss: 0.7482 - val_accuracy: 0.7502
Epoch 7/20
704/704 [==============================] - 185s 262ms/step - loss: 0.4273 - accuracy: 0.8494 - val_loss: 0.7898 - val_accuracy: 0.7516
Epoch 8/20
704/704 [==============================] - 181s 257ms/step - loss: 0.3520 - accuracy: 0.8773 - val_loss: 0.7743 - val_accuracy: 0.7564
Epoch 9/20
704/704 [==============================] - 176s 250ms/step - loss: 0.2862 - accuracy: 0.8991 - val_loss: 0.8657 - val_accuracy: 0.7460
Epoch 10/20
704/704 [==============================] - 174s 247ms/step - loss: 0.2555 - accuracy: 0.9104 - val_loss: 0.9012 - val_accuracy: 0.7574
Epoch 11/20
704/704 [==============================] - 176s 249ms/step - loss: 0.2057 - accuracy: 0.9279 - val_loss: 1.0603 - val_accuracy: 0.7406
Epoch 12/20
704/704 [==============================] - 179s 255ms/step - loss: 0.1758 - accuracy: 0.9385 - val_loss: 1.1312 - val_accuracy: 0.7424
Epoch 13/20
704/704 [==============================] - 186s 263ms/step - loss: 0.1490 - accuracy: 0.9461 - val_loss: 1.1926 - val_accuracy: 0.7358
Epoch 14/20
704/704 [==============================] - 191s 271ms/step - loss: 0.1278 - accuracy: 0.9543 - val_loss: 1.3430 - val_accuracy: 0.7486
313/313 - 10s - loss: 1.4035 - accuracy: 0.7379

테스트 정확도: 0.7379000186920166
```

정확도는 73.79%가 나왔으며, EarlyStopping 콜백 함수가 작동하여 Epoch이 14에서 정지된 것으로 보인다. Val_loss가 감소하지 않고 증가함.