CNN 사용한

음식이미지 분류

2019100889 산업경영공학과 이수진

CONTENTS

01 프로젝트 설명

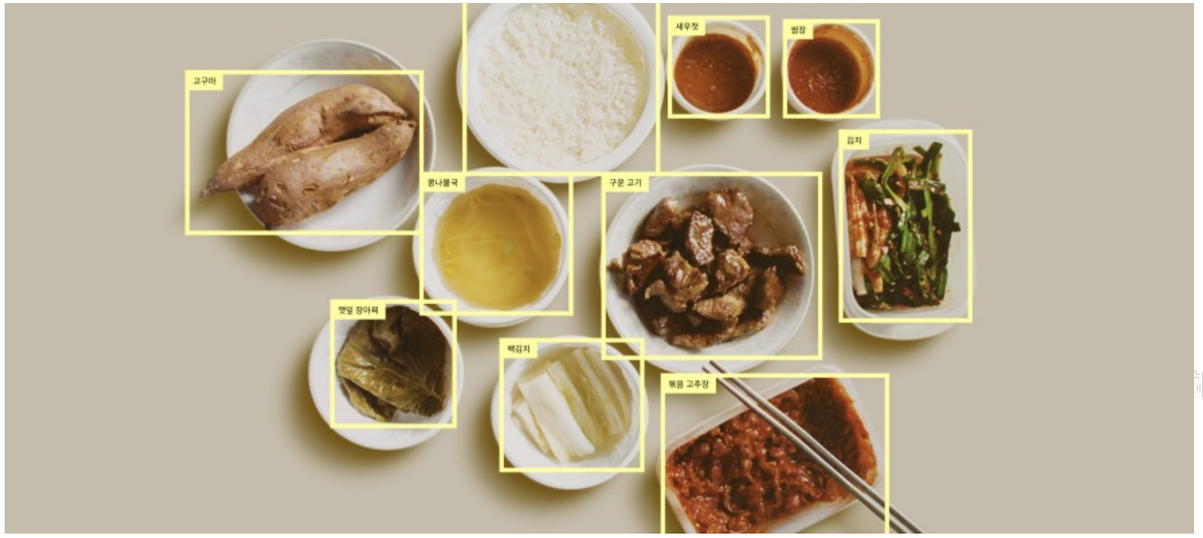
01 주제 및 기대결과

음식 이미지 분류

CNN 사용한 이미지 분류

음식 이미지를 사용해 음식을 분류한다.

한식메뉴 대표 6종을 선정하여 양질의 이미지데이터를 수집, 구축하고 이를 기반으로 음식의 종류를 추정할 수 있는 알고리즘을 개발하고자 한다



02 기술수준 및 동향

01 이미지 분류 기술

이미지 분류 기술 동향

딥러닝 기반 이미지 인식 기술 동향

적대적 예제

악의적 노이즈를 이미지에 주입해서 적대적 예제를 만들 수 있다.

이 적대적 예제는 사람이 판단하기에는 무리가 없으나 인공신경망의 판정을 교란시킨다. 이 적대적 예제를 이용해 사람이나 표지판을 인식하지 못하게 하는 사례가 발생했다. 이처럼 적대적 예제 처럼 인공의 노이즈를 응하는 기술의 연구가 활발하다.

따라서 이러한 각종 노이즈에 대응하는 연구, 인식기술이 계속해서 발전할 것으로 보인다.

학습하지 못한 패턴에 대한 처리

인공지능에 입력된 데이터가 학습된 분포의 데이터인지 아닌지 식별하는 것이 중요하다.

이것을 학습 외 분포 이미지 탐지라고 하는데, 이를 위해서는 확률값을 보정하거나, 학습 외 분포 데이터를 GAN으로 생성하고 학습하는 방법이 있다.

03 데이터 설명

01 음식 데이터 설명

음식 데이터

데이터 출처

AI -hub의 한국 음식 이미지 데이터 사용

데이터 구조

데이터: 한식 이미지 데이터베이스는 한식재단의 음식분류 및 한국인이 즐겨 먹는 음식통계를 참조하여 선정된 150종의 음식으로 구성되며, 한식메뉴외국어표기 길라잡이(한식재단, 200 International Korean Menu Guide, 2014)를 참고하여 음식의 종류를 대분류(밥, 면, 국 등) 및 소분류를 결정하고 ID를 부여하고 구조화

최종 데이터: 한 class당 100개의 이미지로 학습 진행

해당 분류에서 김치찌개, 된장찌개, 미역국, 배추김치, 백김치, 잡곡밥으로 총 6개의 클래스만 사용

대분류	소분류	대분류	소분류
구이	갈비구이,갈치구이,고등어구이,곱창구이,닭갈비, 더덕구이,떡갈비,불고기,삼겹살,장어구이,조개구이, 황태구이,훈제오리	국	계란국, 떡국/만두국, 무국, 미역국, 북엇국, 소고기무국, 시래기국, 육개장, 콩나물국
김치	갓김치, 깍두기, 나박김치, 무생채, 배추김치, 백김치, 부추김치, 열무김치, 오이소박이, 총각김치, 파김치	나물	가지볶음, 고사리나물, 미역줄기볶음, 숙주나물, 시금치나물, 애호박볶음
떡	경단	만두	만두
면	막국수, 물냉면, 비빔냉면, 수제비, 열무국수, 잔치국수, 쫄면, 칼국수, 콩국수, 라면, 자장면, 짬뽕	무침	고추된장무침, 파리고추무침, 도토리묵, 잡채, 도라지무침, 콩나물무침, 홍어무침
밥	김밥, 김치볶음밥, 비빔밥, 새우볶음밥, 알밥, 잡곡밥, 주먹밥, 유부초밥	볶음	건새우볶음, 오징어채볶음, 감자채볶음, 고추장진미채볶음, 두부김치, 떡볶이, 라볶이, 멸치볶음, 소세지볶음, 어묵볶음, 제육볶음, 쭈꾸미볶음
쌈	보쌈	음청류	수정과, 식혜

04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

음식 데이터 전처리

1. 음식 데이터를 불러와 256*256 사이즈로 변형하기

cv2의 imread, resize 사용

2. training set 과 test set으로 나누기

data 자체가 training과 test으로 나누어져 있지 않기 때문에 불러온 데이터를 random.shuffle로 섞은 뒤 80:20으로 나누어 리스트로 저장한다.

3. 이미지와 Label 분류하기

파일 이름을 숫자로 변형하여 해당 이미지에 label로 설정한다.
X를 이미지(픽셀 배열)로, y를 label로 하여 각각의 배열을 생성한 후 append한다.

4. 픽셀값 정규화 하기

255로 나누어 픽셀을 0과 1사이로 바꾸고 256*256 모양으로 변형한다.

04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

음식 분류기

음식의 종류를 검출하기 위한 모델

컨볼루션 신경망 모델 사용

padding = 'same"

출력 이미지 사이즈가 입력 이미지 사이즈와 동일

활성함수 softmax

다중 클래스 분류이기 때문에 softmax 함수를 활성함수로 사용

클래스가 6개이기 때문에 node 수를 6개로 설정

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 128)     3584

 max_pooling2d (MaxPooling2D  (None, 128, 128, 128)    0
 )

 conv2d_1 (Conv2D)           (None, 128, 128, 32)      36896

 max_pooling2d_1 (MaxPooling  (None, 64, 64, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 64, 64, 64)        18496

 max_pooling2d_2 (MaxPooling  (None, 32, 32, 64)       0
 2D)

 conv2d_3 (Conv2D)           (None, 32, 32, 64)        36928

 max_pooling2d_3 (MaxPooling  (None, 16, 16, 64)       0
 2D)

 dropout (Dropout)           (None, 16, 16, 64)        0

 flatten (Flatten)           (None, 16384)             0

 dense (Dense)               (None, 256)               4194560

 dense_1 (Dense)             (None, 6)                 1542

=================================================================
Total params: 4,292,006
Trainable params: 4,292,006
Non-trainable params: 0
_________________________________________________________________
```

04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

음식 분류기

모델 훈련

epoch =200

에포크 200으로 모델 훈련

monitor='val_loss', patience=15로 early stoppping 설정

```
#역전파로 모델 파라미터 업데이트 한다.
#200epoch로 모델 훈련한다.
callbacks = [keras.callbacks.EarlyStopping(monitor='val_loss', patience=15)]
opt = Adam(lr=0.0001)
model.compile(optimizer = opt , loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True) , metrics = ['accuracy'])
history = model.fit(x_train,y_train,epochs = 200, validation_data = (x_val, y_val), callbacks=callbacks)
```

```
Epoch 20/200
3/3 [==============================] - 2s 769ms/step - loss: 0.4966 - accuracy: 0.8625 - val_loss: 1.6004 - val_accuracy: 0.3897
Epoch 21/200
3/3 [==============================] - 2s 817ms/step - loss: 0.3513 - accuracy: 0.9000 - val_loss: 1.8093 - val_accuracy: 0.3574
Epoch 22/200
3/3 [==============================] - 2s 857ms/step - loss: 0.3812 - accuracy: 0.8875 - val_loss: 1.6073 - val_accuracy: 0.4030
Epoch 23/200
3/3 [==============================] - 2s 841ms/step - loss: 0.2839 - accuracy: 0.9250 - val_loss: 1.7024 - val_accuracy: 0.3612
Epoch 24/200
3/3 [==============================] - 2s 779ms/step - loss: 0.2512 - accuracy: 0.9375 - val_loss: 1.7580 - val_accuracy: 0.3688
Epoch 25/200
3/3 [==============================] - 2s 819ms/step - loss: 0.2211 - accuracy: 0.9625 - val_loss: 1.6595 - val_accuracy: 0.4030
Epoch 26/200
3/3 [==============================] - 2s 787ms/step - loss: 0.1602 - accuracy: 0.9875 - val_loss: 1.7049 - val_accuracy: 0.3897
Epoch 27/200
3/3 [==============================] - 2s 775ms/step - loss: 0.1608 - accuracy: 0.9750 - val_loss: 1.7942 - val_accuracy: 0.3802
Epoch 28/200
3/3 [==============================] - 2s 781ms/step - loss: 0.1421 - accuracy: 0.9875 - val_loss: 1.8162 - val_accuracy: 0.3916
```

에포크 28에서 중단됨

04 적용기법 및 모델

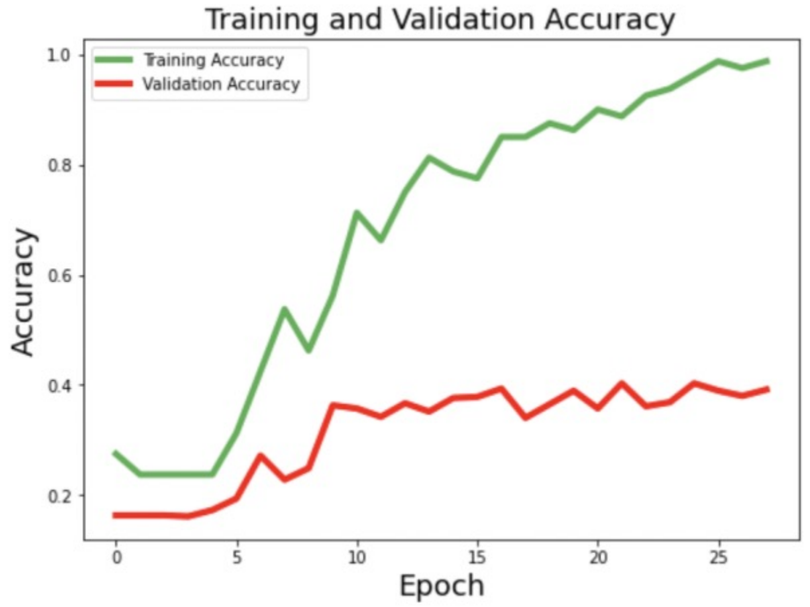
01. 음식 데이터 전처리

02. 음식 분류기

음식 분류기

training 및 validation loss plotting

epoch 에 대한 정확도 점수 분석



04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

음식 분류기

정밀도, 재현율 및 F1 점수의 형태로 성능 측정

CNN모델의 카테고리별 성능

	precision	recall	f1-score	support
된장찌개	0.29	0.35	0.31	86
잡곡밥	0.34	0.35	0.34	89
김치찌개	0.35	0.59	0.44	86
배추김치	0.51	0.25	0.34	88
미역국	0.88	0.51	0.64	89
백김치	0.30	0.31	0.30	88
accuracy			0.39	526
macro avg	0.44	0.39	0.40	526
weighted avg	0.45	0.39	0.40	526

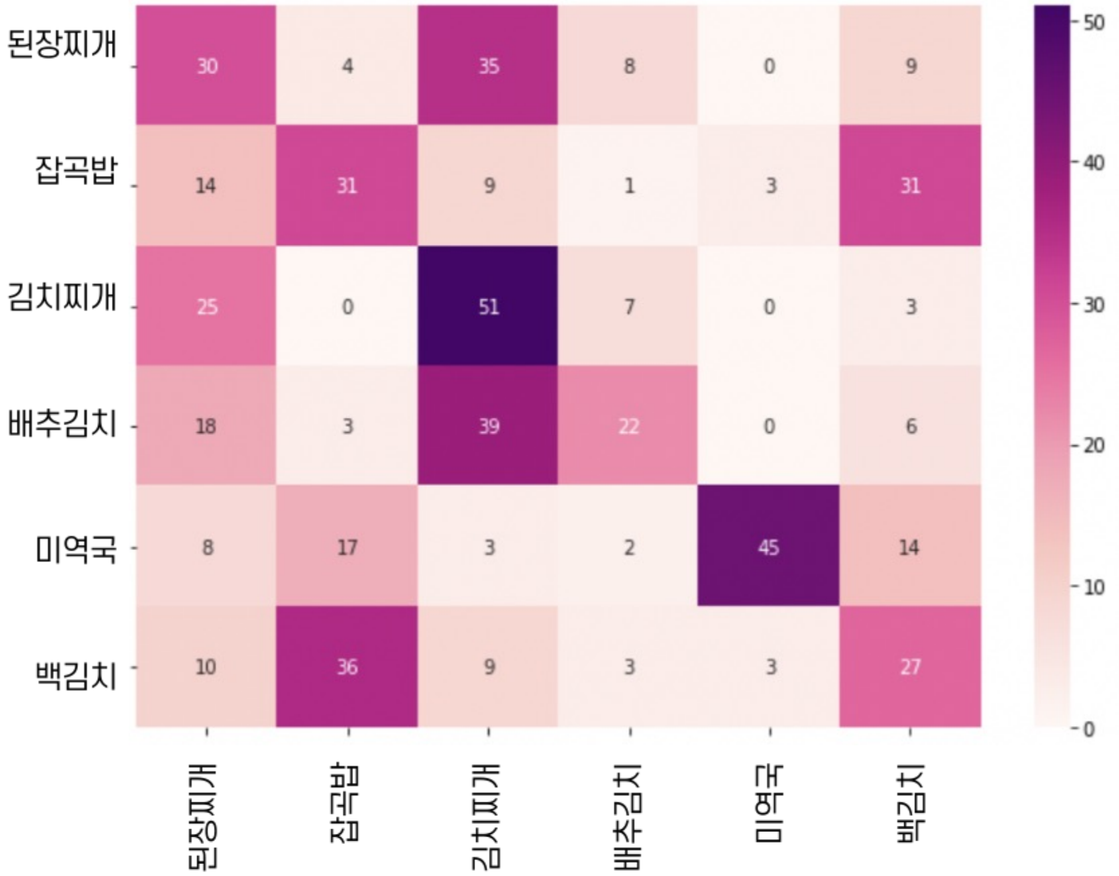
04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

음식 분류기

Confusion matrix



04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

03 전이학습

전이학습

전이학습

학습 데이터가 부족한 분야의 모델 구축을 위해 데이터가 풍부한 분야에서 훈련된 모델을 재사용하는 학습 기법

성능향상을 위해 전이학습 사용

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 mobilenetv2_1.00_224 (Funct  (None, 8, 8, 1280)       2257984
 ional)

 global_average_pooling2d (G  (None, 1280)             0
 lobalAveragePooling2D)

 dropout_2 (Dropout)         (None, 1280)              0

 dense_4 (Dense)             (None, 20)                25620

=================================================================
Total params: 2,283,604
Trainable params: 25,620
Non-trainable params: 2,257,984
_________________________________________________________________
```

ImageNet 데이터 셋에서 훈련된 모델의 가중치를 가지고 진행

MobileNetV2 모델 사용

사전 훈련 된 모델의 입력 및 출력 레이어를 입력 및 출력 레이어로 대체

04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

03 전이학습

전이학습

전이학습

에포크 800으로 학습 진행했을 때 285에서 early stop

validation loss 0.756

```
3/3 [==============================] - 2s 917ms/step - loss: 0.0457 - accuracy: 1.0000 - val_loss: 0.6958 - val_accuracy: 0.7567
Epoch 616/800
3/3 [==============================] - 2s 905ms/step - loss: 0.0438 - accuracy: 1.0000 - val_loss: 0.6957 - val_accuracy: 0.7567
Epoch 617/800
3/3 [==============================] - 2s 916ms/step - loss: 0.0403 - accuracy: 1.0000 - val_loss: 0.6956 - val_accuracy: 0.7567
Epoch 618/800
3/3 [==============================] - 2s 913ms/step - loss: 0.0470 - accuracy: 1.0000 - val_loss: 0.6952 - val_accuracy: 0.7567
Epoch 619/800
3/3 [==============================] - 2s 921ms/step - loss: 0.0431 - accuracy: 1.0000 - val_loss: 0.6951 - val_accuracy: 0.7567
Epoch 620/800
3/3 [==============================] - 2s 914ms/step - loss: 0.0443 - accuracy: 1.0000 - val_loss: 0.6951 - val_accuracy: 0.7567
Epoch 621/800
3/3 [==============================] - 2s 901ms/step - loss: 0.0374 - accuracy: 1.0000 - val_loss: 0.6953 - val_accuracy: 0.7567
Epoch 622/800
3/3 [==============================] - 2s 914ms/step - loss: 0.0415 - accuracy: 1.0000 - val_loss: 0.6953 - val_accuracy: 0.7567
Epoch 623/800
3/3 [==============================] - 2s 942ms/step - loss: 0.0479 - accuracy: 1.0000 - val_loss: 0.6952 - val_accuracy: 0.7567
Epoch 624/800
3/3 [==============================] - 2s 909ms/step - loss: 0.0348 - accuracy: 1.0000 - val_loss: 0.6950 - val_accuracy: 0.7567
```

04 적용기법 및 모델

01. 음식 데이터 전처리

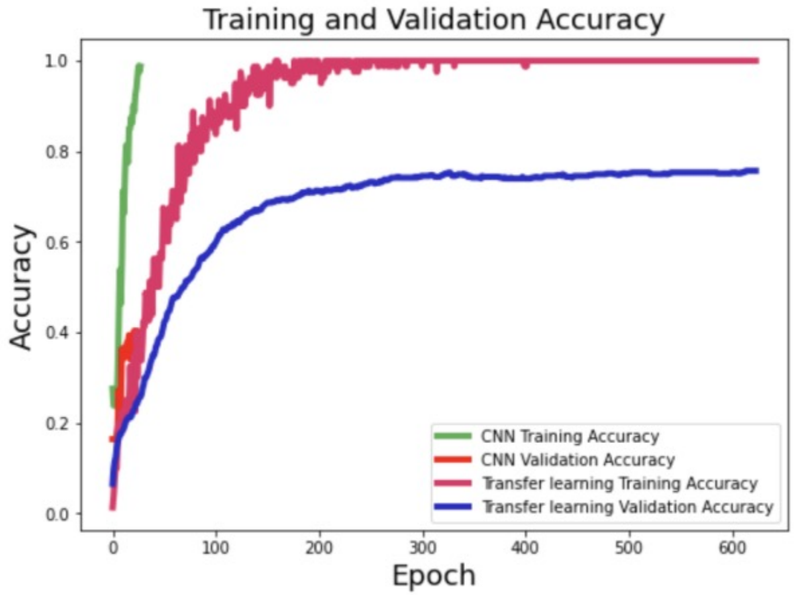
02. 음식 분류기

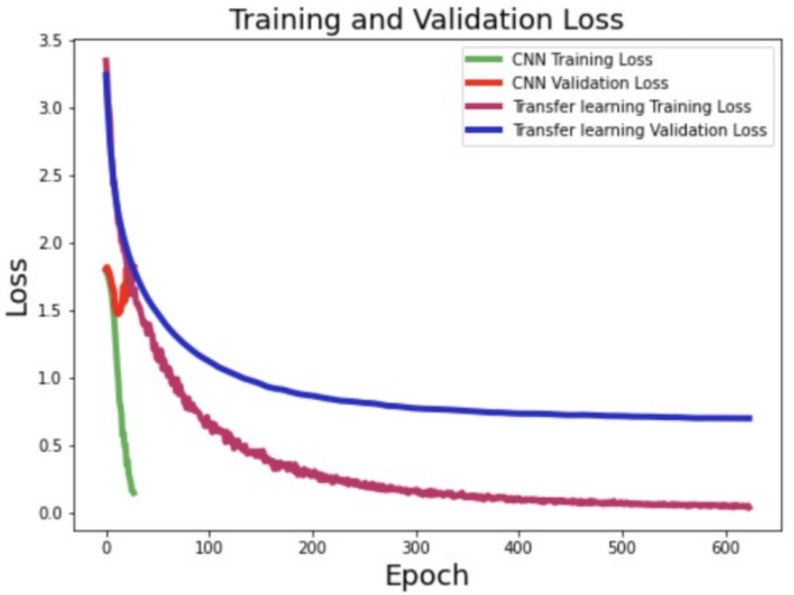
03 전이학습

전이학습

training 및 validation loss plotting

epoch 에 대한 정확도 점수 분석





04 적용기법 및 모델

01. 음식 데이터 전처리

02. 음식 분류기

03 전이학습

전이학습

정밀도, 재현율 및 F1 점수의 형태로 성능 측정

CNN모델의 카테고리별 성능

	precision	recall	f1-score	support
된장찌개	0.68	0.60	0.64	86
잡곡밥	0.89	0.82	0.85	89
김치찌개	0.57	0.72	0.64	86
배추김치	0.73	0.91	0.81	88
미역국	0.88	0.65	0.75	89
백김치	0.88	0.83	0.85	88
accuracy			0.76	526
macro avg	0.77	0.76	0.76	526
weighted avg	0.77	0.76	0.76	526

04 적용기법 및 모델

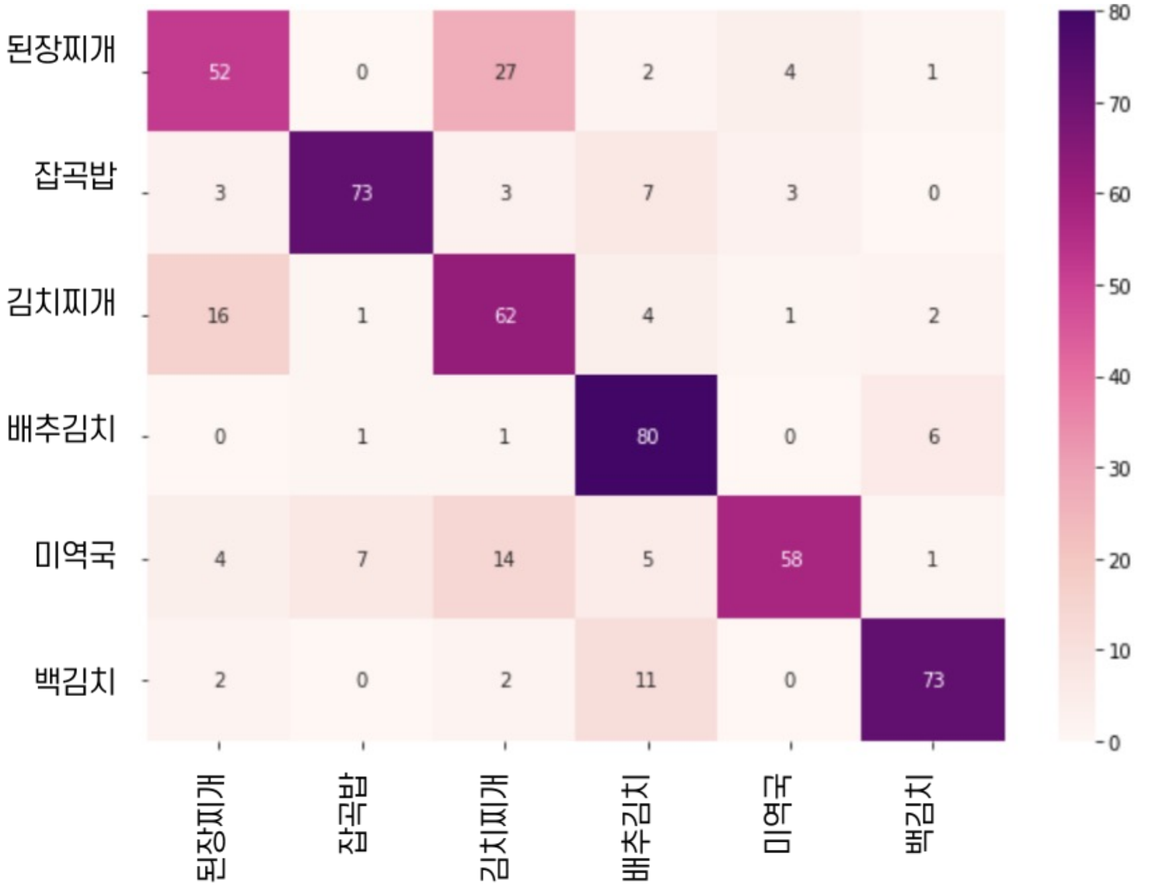
01. 음식 데이터 전처리

02. 음식 분류기

03 전이학습

전이학습

Confusion matrix



05 결론

결론

Validation Accuracy = 0,75의 정확도

여전히 잘못 예측하는 부분은 있으나 전이학습을 쓰기 전보다 성능이 좋아졌음을 알 수 있음