

CNN 을 이용한 음식 판별시스템 개발



과 목 : 컴퓨터종합설계_01(COM407-02)

담당 교수 : 오승현 교수님

이 름 : 오아연, 조성은, 김지훈

학 번 : 2020213297, 2020213264, 2017211836

학 과 : 컴퓨터공학전공, 컴퓨터공학과

제 출 일 : 2023.07.31

[목차]

1. 서론

1.1 연구 배경 및 목적

2. 개발 환경 및 개발 내용

2.1 개발 환경

2.1.1 Google Colaboratory

2.1.2 Shutil

2.1.3 Matplotlib.pyplot, Matplotlib.image

2.1.4 Tensorflow

2.2 개발 내용

2.2.1 Food-101 dataset

2.2.2 Test set, Train set

2.2.3 CNN

2.2.4 Batch_size, Epoch

2.2.5 Inception v3

2.2.6 Data argument

2.2.7 Earlstopping

3. 실험 및 결과

3.1 실험

3.1.1 DataSet_Mini

3.2 결과

4. 결론

부록: 소스 코드

1. 서론

1.1 연구 배경 및 목적

최근 발생한 코로나 바이러스로 인해 배달 서비스 플랫폼을 이용하는 사람들이 많아졌다. 과학기술정보통신부는 신종 코로나바이러스 감염증(코로나 19)으로 상품배송과 음식 배달 수요가 급증한 것을 주요 원인으로 꼽았다. 취업 플랫폼인 잡코리아와 알바몬에서는 최근 20대 이상 성인 남녀 1791명을 대상으로 ‘코시국 건강 관리 현황’을 주제로 설문한 결과 65.5%가 ‘몸무게가 늘었다’고 답했다. ‘몸무게가 늘었다고 답한 이들은 그 이유를 무엇이라고 생각하고 있을까?’에 대한 질문으로는 ‘배달 음식과 레트로트 식품 등을 자주 먹어서’ 체중이 증가한 것 같다는 의견이 응답률 71.7%로 가장 많았다. 한편, 설문에 참여한 응답자 중 93.4%는 코시국에 확실히 살을 빼기 위해 현재 다이어트를 하고 있거나 곧 할 계획이라고 답했다. 그리고 이들이 체중 감량을 위해 선택한 다이어트 방법은 ‘식단 조절’이 응답률 84.1%로 가장 많았다.

위와 같이 다이어트의 수요가 증가하였다. 그에 따라 다이어트를 시작하는 사람들의 수요에 맞춰 다양한 콘텐츠들이 생겨났다. 특히, 다이어트와 관련된 운동과 식단을 알려주는 동영상 매체들과 일상에서 먹고 움직이는 것들을 기록하는 앱들이 생겨났다, 그 중 식단 어플에 관심을 두게 되었다. 관련 연구로는 첨단 IT 기술에 헬스케어를 접목한 kakaoVX에서 만든 스마트홈트가 있다.

스마트홈트와 같이 현재 출시되고 많이 알려진 어플들은 많이 알려진 어플들은 음식 사진만 카메라로 찍으면 AI 코치가 분석하여 음식명과 칼로리를 자동으로 등록해주는 방식이다. 즉 비접촉 스캐닝으로 실시간 음식의 종류를 분석하고 Food Digitalization을 통해 데이터를 생성해

사용자들에게 보여주는 방식이다. 그러나 이와 같은 어플들은 일반적인 1인분 기준의 양을 등록해주는데 이는 자신이 섭취한 양과 칼로리가 다를 수 있다. 이에 우리는 AI를 이용한 음식 판별 시스템을 개발해보고자 한다. 음식을 카메라로 촬영했을 시 음식의 종류와 양, 또한 본인이 섭취한 양의 칼로리를 식별할 수 있는 앱을 만들고자 한다.

2. 개발 환경 및 개발 내용

2.1 개발 환경

2.1.1 Google Colaboratory

구글 코랩은 웹에서 텍스트 기반의 코딩을 할 수 있도록 구글에서 제공하는 에디터이다. 브라우저에서 Python 을 작성하고 실행 가능하다. 코랩은 클라우드 기반으로 주피터 노트북 개발 환경이다. 코랩은 구글 드라이브, 도커, 리눅스, 구글 클라우드 등 기술로 이루어져 있다. 컴퓨터에 GPU 가 없거나 저사양이어도 사용이 가능하다.

코랩은 별도의 파이썬 설치가 필요하지 않다. 데이터에 데이터에 분석 사용되는 Tensorflow, Keras, matplotlib, scikit-learn, pandas 와 같은 패키지가 기본적으로 설치되어있다. GPU 를 무료로 사용 가능하다. Jupyter 노트북과 비슷하지만 더 좋은 기능을 제공한다. 깃과 연동이 가능하여 사람들과 협업하여 코딩이 가능하다.

2.1.2 Shutil

파일과 파일 모음에 대한 여러 가지 고수준 연산을 제공, 파일이나 디렉터리 등에 대한 고수준 명령을 모아놓은 기본 모듈이다. 파일과 폴더를 이동하거나 복사할때 사용한다.

2.1.3 Matplotlib.pyplot, Matplotlib.image

Matplotlib은 다양한 데이터를 많은 방법으로 도식화 할 수 있도록 하는 파이썬 라이브러리, 데이터를 다양한 그래프로 만들어주는 파이썬 라이브러리이다. 그 중 matplotlib.pyplot 모듈은 데이터를 다양한 그래프로 만들어주는 파이썬 라이브러리이다. 그리고 matplotlib.image는 image 파일을 읽어서 그래프로 나타내주는 기능이다.

2.1.4 Tensorflow

구글이 오픈소스로 공개한 머신러닝 라이브러리로 딥러닝과 머신러닝 분야를 누구나 쉽게 사용할 수 있도록 다양한 기능을 제공한다.

2.2 개발 내용

2.2.1 Food-101 dataset

AI Hub 사이트의 데이터 중 한국 음식 이미지 데이터셋, 셀렉트스타의 음식 사진 데이터셋 등 여러 데이터 셋이 있지만 15GB 등 용량이 너무 크고 데이터 정제가 덜 되어져 있다. 한편, Food-101 데이터셋은 약 5GB로 용량이 비교적 작고 데이터 정제가 잘 되어 있어 이 데이터셋을 사용하였다.

최상위 폴더 food101 폴더 하위에 images 폴더와 meta 폴더가 있고 images 폴더 밑에는 101 개의 이미지 클래스 폴더들이 있고, 그 안엔 각각 1,000 개의 이미지들이 있다.



그림 1 Food 101 로고와 음식 사진들

2.2.2 Test set, Train set

알고리즘을 학습시켜 모델을 구성하고 또 그 모델을 통해 예측을 하기 위해선 학습을 위한 train 세트와 예측을 위한 test 세트가 필요하다.



그림 2 Test set 과 Train set

모델을 학습시키기 위해 보통 x_{train} , y_{train} , x_{test} , y_{test} 로 나누는데 여기서 y_{train} 과 y_{test} 는 위 그림의 Test 부분에 해당하고 x_{train} 은 Train 부분, x_{test} 는 Validation 에 해당한다.

Train 부분은 모델을 학습시키는 부분이고 Test 부분은 모델을 검증하는 부분이다. Validation 이란 모델의 성능을 평가하기 위해 사용되는 데이터로 Test 세트로만 모델을 평가한다면 과적합 문제가 발생할 수 있기에 이를 방지하기 위해 Validation 세트를 이용해 최적의 파라미터를 찾는데 이용한다

2.2.3 CNN

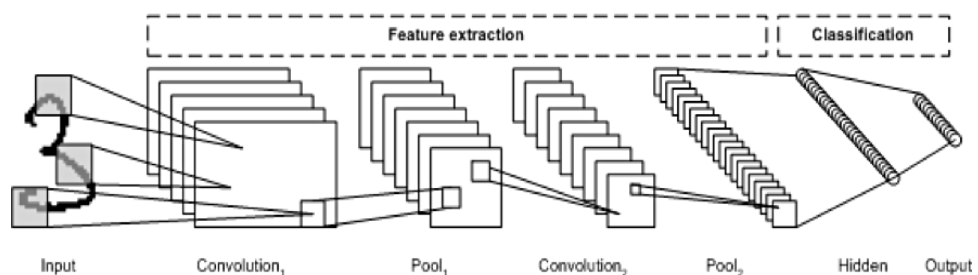


그림 3 CNN 을 설명하는 그림

CNN 은 Convolution + Neural Networks 의 약자로 컴퓨터 비전 분야
 중에서 이미지 인식 문제를 해결에 특화된 딥러닝 학습 모델 중 하나이다. CNN 의
 방식 중 첫 번째 방식은 합성곱 (Convolution) 이다. 학습의 대상은 필터
 파라미터이며 순회하며 채널별로 합성곱을 실행하고 모든 채널의 합성곱의
 합으로 Feature Map 을 제작한다.

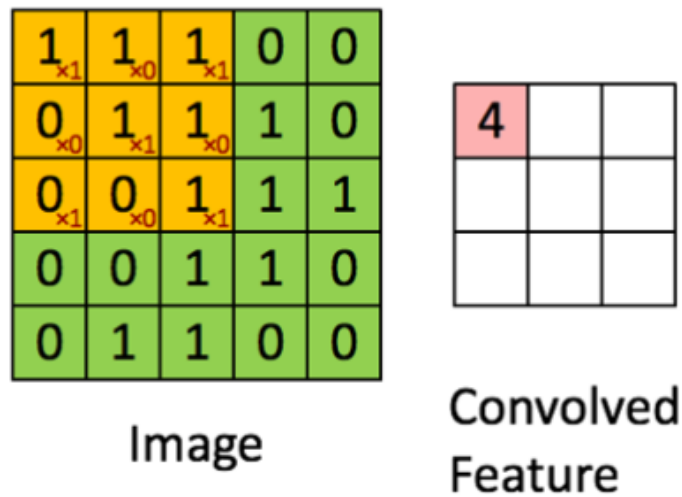


그림 4 합성곱 (Convolution)

두 번째 방식은 Pooling 이다. Pooling 은 정사각 행렬 영역 값의
 최댓값을 모으거나 특정영역의 평균을 구하는 방식으로 동작한다. 음식
 데이터셋을 학습 시킬 때는 주로 Max Pooling 방식을 이용하기에 이 방식을
 채택하였다.

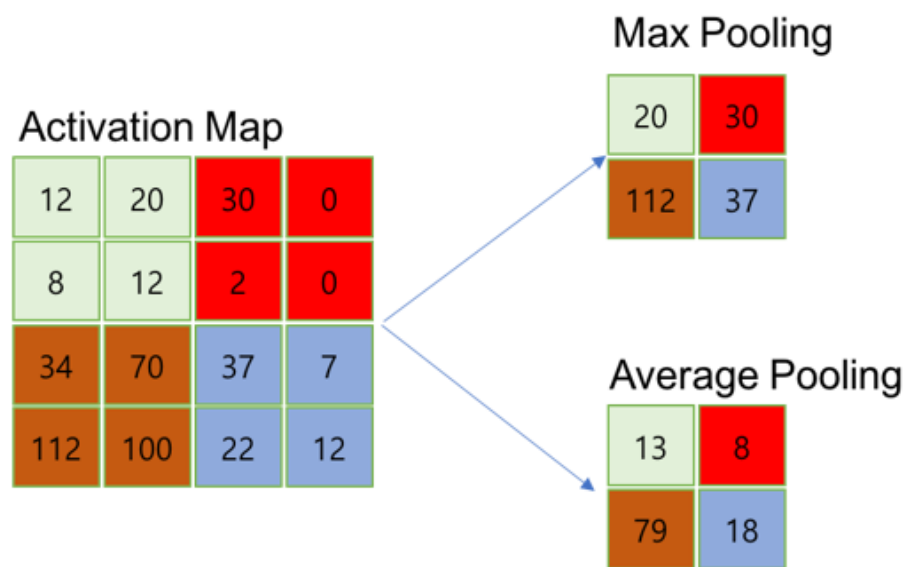


그림 5 Pooling 방식

2.2.4 Batch_size, Epoch

Batch_size 란 parameter 를 업데이트할 때마다 사용할 데이터의 개수 즉, 한번 학습할 때 얼마만큼의 데이터를 학습하는가를 의미한다. 배치사이즈가 클수록 많은 데이터를 저장해 두어야 하므로 용량이 커야한다. 반면, 배치사이즈가 작으면 학습은 촘촘하게 되겠지만 계속 레이블과 비교하고, 가중치를 업데이트하는 과정을 거치면서 시간이 오래 걸린다. Epoch 는 전체 데이터셋을 몇 번 학습하는가를 의미한다. Epoch 는 하나의 데이터셋을 몇 번 반복 학습할지 정하는 파라미터 같은 데이터셋이라 할지라도 가중치가 계속해서 업데이트되기 때문에 모델이 추가적으로 학습이 가능하다.

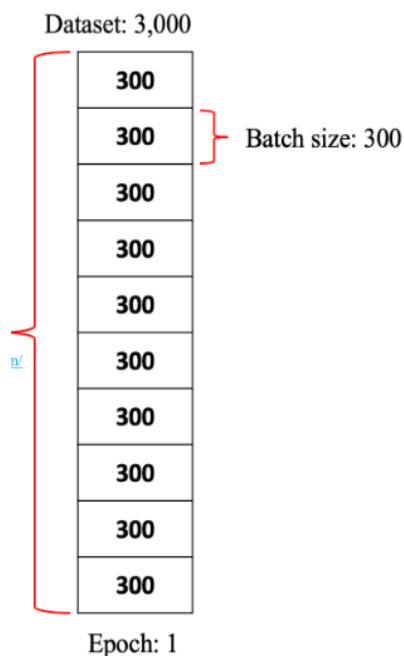


그림 6 Batch_size 와 Epoch

2.2.5 Inception v3

Inception-v2 구조에서 위에서 설명한 기법들을 하나하나 추가해 성능을 측정하고, 모든 기법들을 적용하여 최고 성능을 나타내는 모델이 Inception-v3 이다. 즉, Inception-v3 은 Inception-v2 에서 BN-auxiliary + RMSProp + Label Smoothing

+ Factorized 7x7 을 다 적용한 모델이다.

Inception-v2는 42층의 신경망이지만 연산량은 GoogLeNet 보다 2.5배 많고, VGGnet 과 비슷하다. Inception-v2 에는 3 종류의 inception module 이 사용된다.

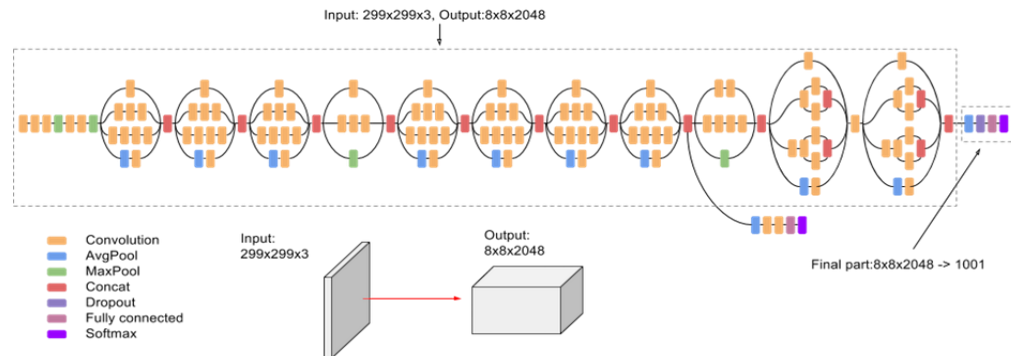


그림 7 Inception v3

2.2.6 Data argument

Data argument(데이터 증강)란 적은 양의 데이터를 바탕으로 다양한 알고리즘을 통해 데이터의 양을 늘리는 기술이다. 원본 데이터로부터 새로운 데이터를 만들어내는 것을 말하기도 한다. 데이터 증강은 사용하는 데이터의 종류에 따라 특성이 달라지는데, 새로운 데이터는 원본 데이터의 특성을 잘 반영하고 있어야 한다는 주의점이 있다. 그 중 우리는 Basic Image Manipulation 방법을 사용하였다. Basic Image Manipulation은 기존 이미지를 Crop, Rotate, Contrast, Invert, Flip 시켜 새로운 이미지를 만들어내는 방식이다.



그림 8 Data argument

2.2.7 Earlystopping

모델을 학습시키다 보면 과도하게 학습을 하여 train 데이터에 대한 성능은 늘어나지만 test 데이터에 대한 성능은 감소하는 시점이 온다. 10 train 데이터는 편향오차가 줄어들지만 test 데이터는 분산오차가 커지기 때문인데 이를 방지하기 위해 earlystopping 을 이용하였다. Earlystopping 은 정해진 학습 횟수(epochs)전에 검증 데이터셋의 손실이 감소하지 않고 증가하기 시작한다면 학습을 중단시키는 방법이다.

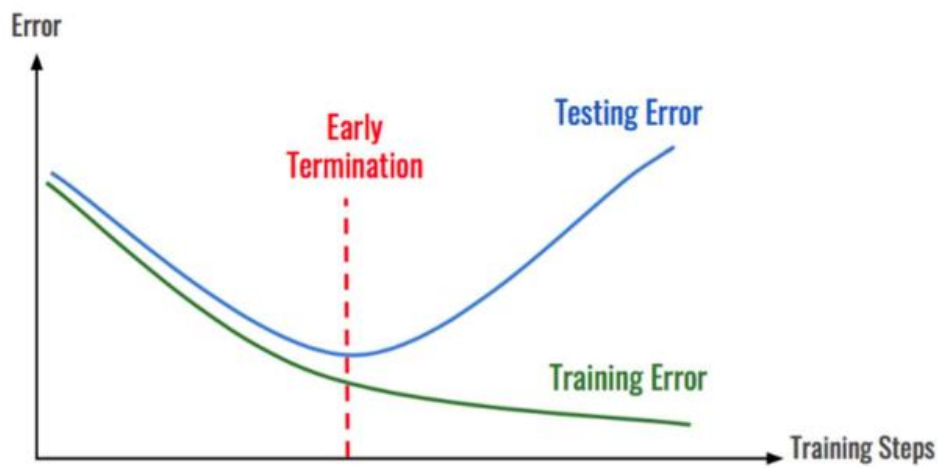


그림 9 Earlystopping

3. 실험 및 결과

3.1 실험

3.1.1 DataSet_Mini

```
Epoch 1/10 .....  
3/1515 [.....] - ETA: 22:15:34 - loss: 5.2506 - accuracy: 0.0133
```

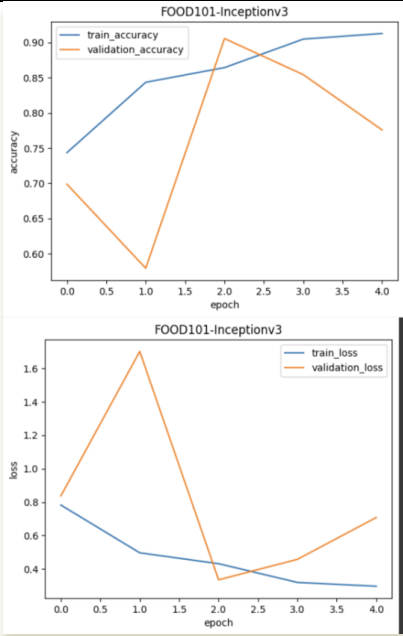
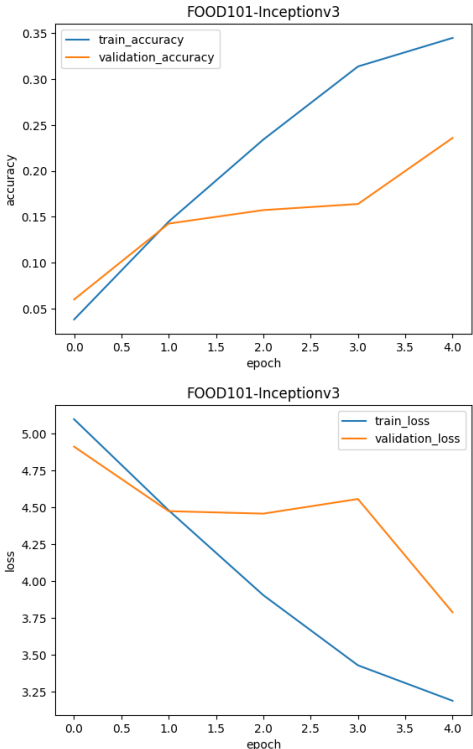
그림 10 과도한 피팅 시간

dataset 의 전체를 이용하여 학습을 시켰을 경우 22 시간이 걸리는 것을 볼 수 있다. 모델 피팅에 과도한 시간이 걸린다. 따라서 본 연구는 dataset 을 dataset_mini 로 세분화하여 시도한 결과 이전보다 적은 시간이 걸리는 것을 알 수 있다.

```
Epoch 1/5  
16/234 [=>.....] - ETA: 1:02:49 - loss: 1.1899 - accuracy: 0.5820
```

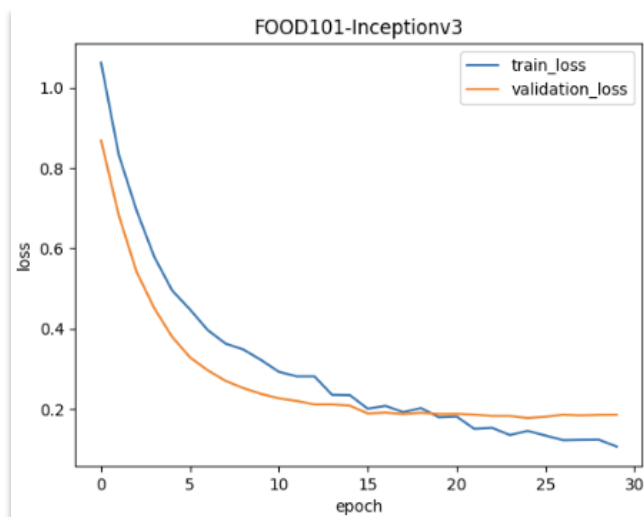
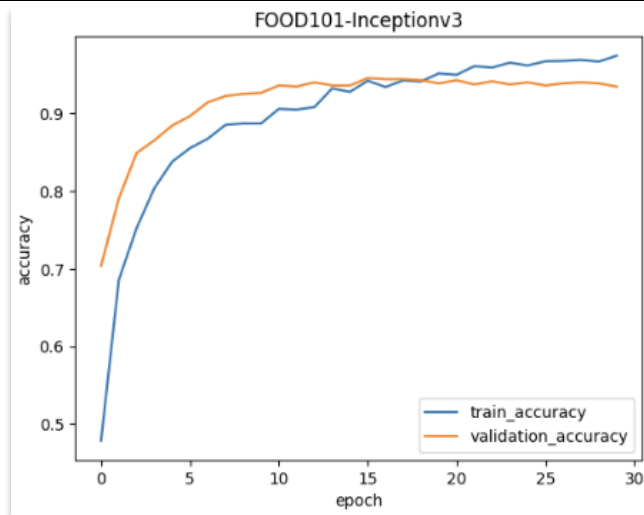
그림 11 많이 줄어든 피팅 시간

3.2 결과

실험	결과
Data Argument 생략 시	 <p>The top graph, titled 'FOOD101-Inceptionv3', plots accuracy (y-axis, 0.60 to 0.90) against epoch (x-axis, 0.0 to 4.0). The blue line (train_accuracy) starts at ~0.74, rises to ~0.84 at epoch 1, and continues to ~0.91 at epoch 4. The orange line (validation_accuracy) starts at ~0.70, drops to ~0.58 at epoch 1, peaks at ~0.90 at epoch 2, and then declines to ~0.78 at epoch 4.</p> <p>The bottom graph, also titled 'FOOD101-Inceptionv3', plots loss (y-axis, 0.4 to 1.6) against epoch (x-axis, 0.0 to 4.0). The blue line (train_loss) starts at ~0.8, drops to ~0.5 at epoch 1, and continues to ~0.35 at epoch 4. The orange line (validation_loss) starts at ~0.85, peaks at ~1.65 at epoch 1, drops to ~0.35 at epoch 2, and then rises to ~0.7 at epoch 4.</p>
Batch_size = 100 Epoch = 10	 <p>The top graph, titled 'FOOD101-Inceptionv3', plots accuracy (y-axis, 0.05 to 0.35) against epoch (x-axis, 0.0 to 4.0). The blue line (train_accuracy) starts at ~0.04 and increases steadily to ~0.34 at epoch 4. The orange line (validation_accuracy) starts at ~0.06, increases to ~0.16 at epoch 3, and then rises to ~0.24 at epoch 4.</p> <p>The bottom graph, also titled 'FOOD101-Inceptionv3', plots loss (y-axis, 3.25 to 5.00) against epoch (x-axis, 0.0 to 4.0). The blue line (train_loss) starts at ~5.1 and decreases steadily to ~3.2 at epoch 4. The orange line (validation_loss) starts at ~4.9, decreases to ~4.45 at epoch 2, rises slightly to ~4.55 at epoch 3, and then drops to ~3.8 at epoch 4.</p>

Batch_size = 16

Epoch = 50



Apple pie 사진 입력

후 예측

```
images.append('upload.jpg')
```

```
predict_class(model_best, images, True)
```

```
1/1 [=====] - 3s 3s/step
```

```
WARNING:matplotlib.image:Clipping input data to the valid range
```

apple_pie



표 1 실험 및 결과

4. 결론

본 연구는 FOOD-101 을 이용한 음식 데이터 수집과 DATA Argument 를 이용한 데이터 최적화 그리고 CNN 의 일종인 InceptionV3 를 이용한 음식 판별 알고리즘을 개발하고 실험한 결과를 분석했다. 실험 결과, 알고리즘이 성공적으로 개발되어 각종 음식 사진에 대한 예측에 성공하였다. 그러나 모든 음식과 사진에 대하여 항상 정확한 음식에 대한 판별을 하지는 못하였고 음식 판별의 종류 또한 대한민국 사람들이 흔히 섭취하는 음식보다는 서양사람들이 선호하는 음식에 대한 판별에 보다 효과적인 모습을 보여주었다.

이러한 결론은 빨간 양념의 음식이 많아 판별이 어려운 한국음식이 판별에 어려움을 있음을 보여준다. 물론 본 연구에서 이용된 데이터 세트가 보다 서양음식에 편향된 데이터 세트임을 감안해야 할 것이다. 또한 다른 음식일 지라도 나라별로 비슷한 음식들이 있기 때문에 이에 대한 정확한 판별은 하지 못하였다.

또 본 연구는 결과적으로 다이어트 등 식단 조절을 위해 음식 사진을 분별하여 각 음식의 칼로리를 계산하는 것에 의의를 뒀지만 음식마다 레시피가 각각 다르기 때문에 이에 대한 정확한 칼로리 분석이 힘들어 대략적인 그리고 평균적인 레시피에 의한 칼로리만 제공이 되었다. 그리고 음식의 양 역시 칼로리에 지대한 영향을 끼치는 요소이지만 음식의 담는 법, 그릇, 재료의 종류에 따라 천차만별로 그 모습이 달라지기 때문에 본 연구의 알고리즘은 음식의 양까지 분별하지는 못했기 때문에 대략적인 1인분 기준으로의 칼로리만 제공이 가능하였다.

따라서 이후 더 정확한 칼로리 예측 서비스를 제공하기 위하여 본 알고리즘을 개선하여 보다 좋은 예측 확률을 만들어내고 각 레시피에 따른 칼로리 예측 기능 등을 후에 추가하고 보다 한국에 맞는 한식 데이터세트를 추가하여 최종적으로 실사용자에게 보다 도움이 되는 식단 조절 도움 기능을 제공하기 위해 연구할 예정이다.

부 록 : 소스코드

깃허브 주소

- <https://github.com/SeongeunJo360/CAPTCHA>

-

소스코드

```
def get_data_extract():
    if "food-101" in os.listdir():
        print("Dataset already exists")
    else:
        tf.keras.utils.get_file(
            'food-101.tar.gz',
            'http://data.vision.ee.ethz.ch/cvl/food-101.tar.gz',
            cache_subdir='/content',
            extract=True,
            archive_format='tar',
            cache_dir=None
        )
        print("Dataset downloaded and extracted!")
```

```
def dataset_mini(food_list, src, dest):
    if os.path.exists(dest):
        rmtree(dest)
    os.makedirs(dest)
    for food_item in food_list :
        print("Copying images into", food_item)
        copytree(os.path.join(src, food_item), os.path.join(dest, food_item))
```

```
food_list = ['apple_pie', 'pizza', 'omelette']
src_train = 'food-101/train'
dest_train = 'food-101/train_mini'
src_test = 'food-101/test'
dest_test = 'food-101/test_mini'
```

```

def train_model(n_classes, num_epochs, nb_train_samples, nb_validation_samples):
    K.clear_session()

    img_width, img_height =
    train_data_dir = 'food-101/train_mini'
    validation_data_dir = 'food-101/test_mini'
    batch_size = 16

    bestmodel_path = 'bestmodel_'+str(n_classes)+'class.hdf5'
    trainedmodel_path = 'trainedmodel_'+str(n_classes)+'class.hdf5'
    history_path = 'history_'+str(n_classes)+'.log'

    train_datagen = ImageDataGenerator(
        preprocessing_function=preprocess_input,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)

```

```

n_classes = 3
epochs = 50
nb_train_samples = train_files
nb_validation_samples = test_files

history, class_map_3 = train_model(n_classes, epochs, nb_train_samples, nb_validation_samples)
print(class_map_3)

```