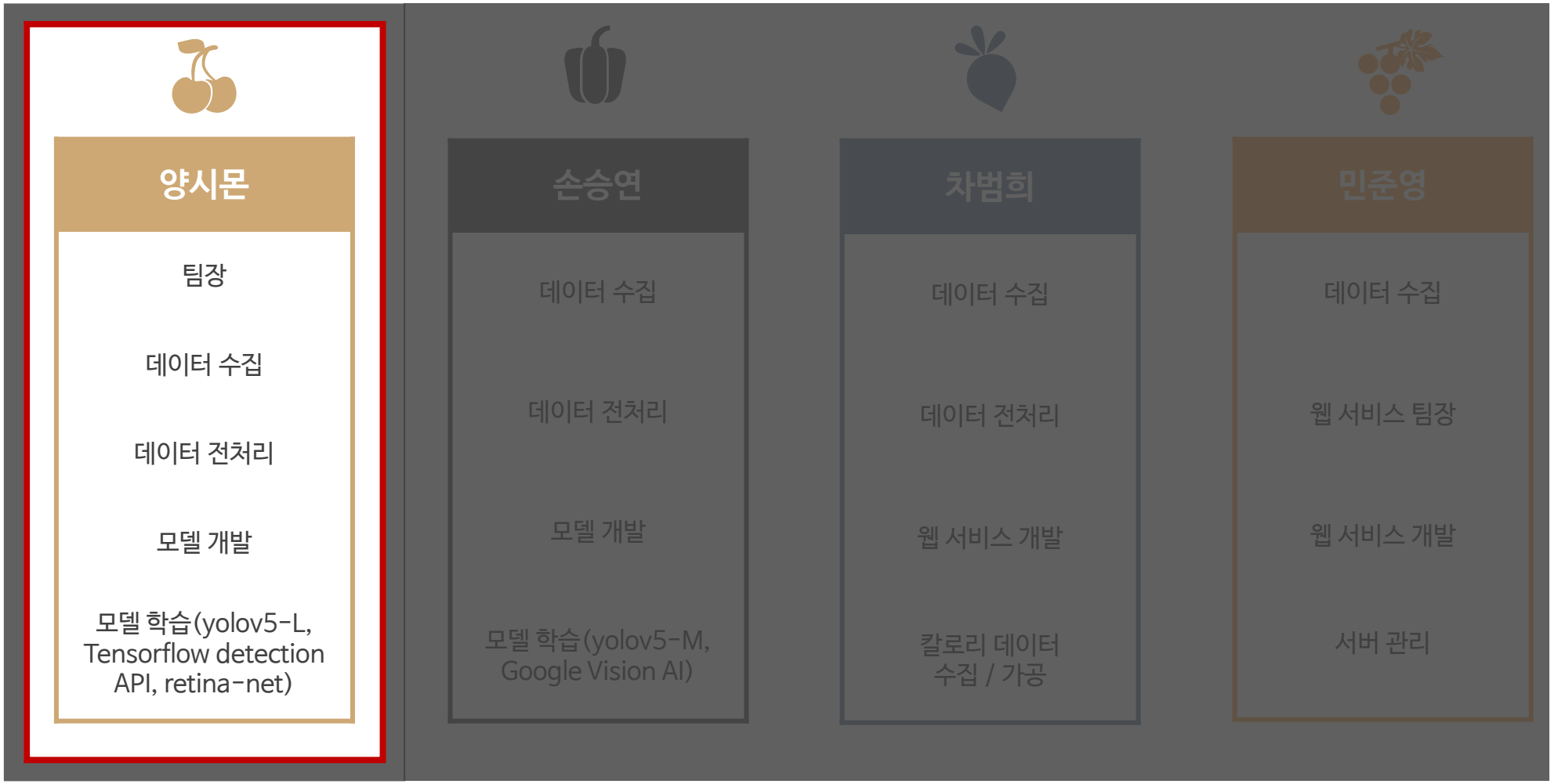
https://www.foodinca.tk

FOOD IN CAMERA

광주 인공지능 사관학교
나무 인텔리전스 기업 프로젝트
양시몬(팀장) 민준영 손승연 차범희

C O N T E N T S

1. 팀원역할



양시몬	손승연	차범희	민준영
팀장	데이터 수집	데이터 수집	데이터 수집
데이터 수집	데이터 전처리	데이터 전처리	웹 서비스 팀장
데이터 전처리	모델 개발	웹 서비스 개발	웹 서비스 개발
모델 개발	모델 학습(yolov5-M, Google Vision AI)	칼로리 데이터 수집 / 가공	서버 관리
모델 학습(yolov5-L, Tensorflow detection API, retina-net)			

2. 프로젝트 개요 및 목적

찰칵! 사진 한 장으로
칼로리와 영양을 찍는다

ex)

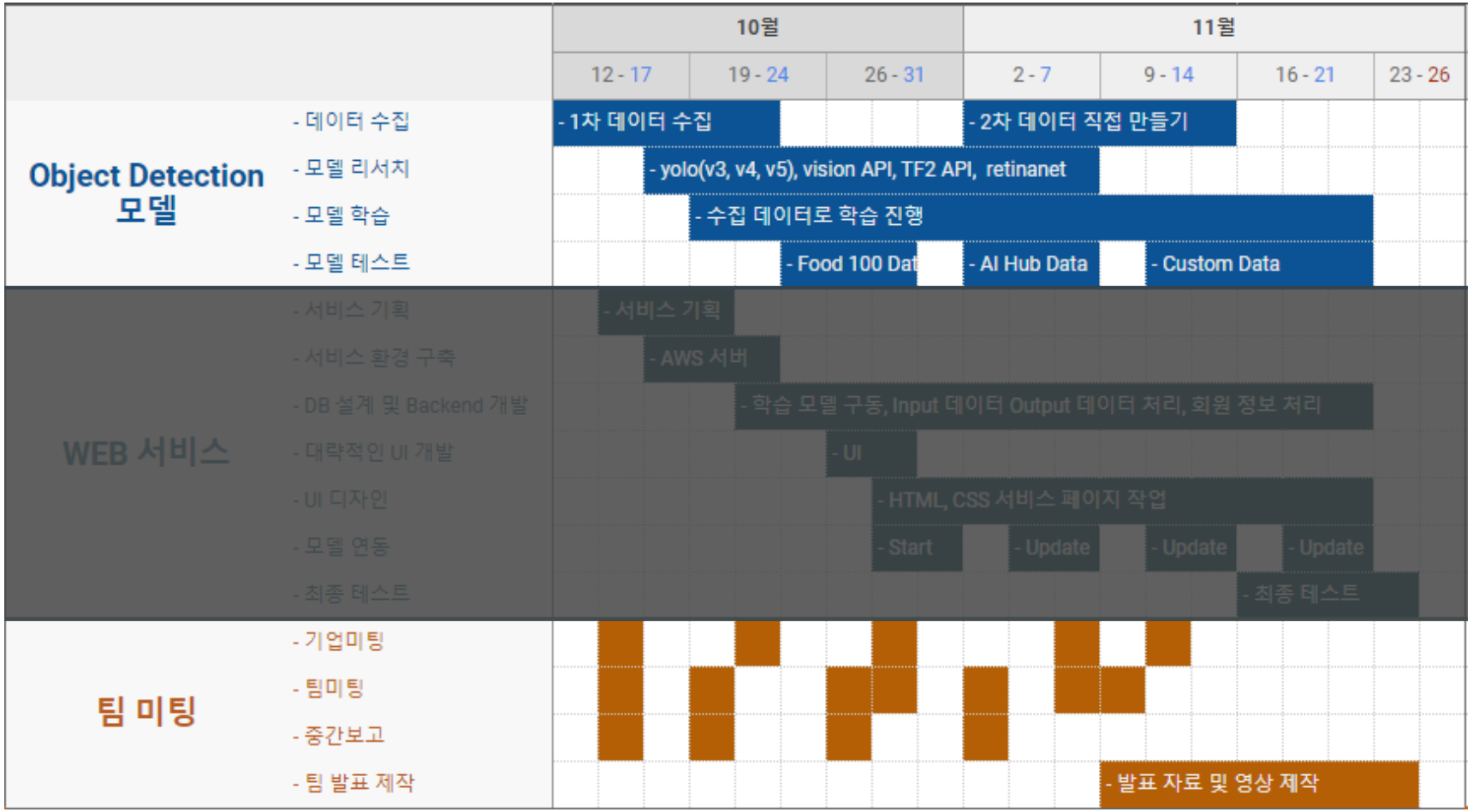
건강에 관심 많은 현대인
코로나의 여파로 건강에 대한 관심이
증가하면서 식단관리의 중요성 증가

Food In Camera (푸딘카)
딥러닝 영상처리 기술을 이용해
음식사진 만으로 식단관리 끝

어려운 식단관리
섭취한 음식을 하나하나 기록하고
영양성분을 일일이 챙기는 번거로움

3. 프로젝트 프로세스

1. 일정표



일정표에 맞춰 프로젝트를 진행하면서 매주 주간보고서를 작성하였으며 아래 링크에서 확인할 수 있다.

- 1주차 : https://docs.google.com/document/d/1o3G8vSst8qDpIa4FJ0bSq473HpQqNRxVzA2huITpTls/edit
- 2주차 : https://docs.google.com/document/d/1nUY4R7fs37Q3F7PAhsumlXpUnCOWCXKKnlQseOACM8U/edit#heading=h.nrnw03t7conb
- 3주차 : https://docs.google.com/document/d/1fEJ81EfdlrKx57JVOYZ139BqIeYIBXNsEnwiw5uVweI/edit#heading=h.nrnw03t7conb

2. 시스템 구성도

작업 참여

4. 프로젝트 단계별 내용

A

모델 개발

1. 데이터셋 수집
2. Object detection 모델 선정
    a. YOLOv5
    b. Retinanet
    c. TF api
    d. Google Vision AI
    e. 선정 결과
3. 모델 학습
4. 데이터셋 재수집 및 학습

B

서비스 개발

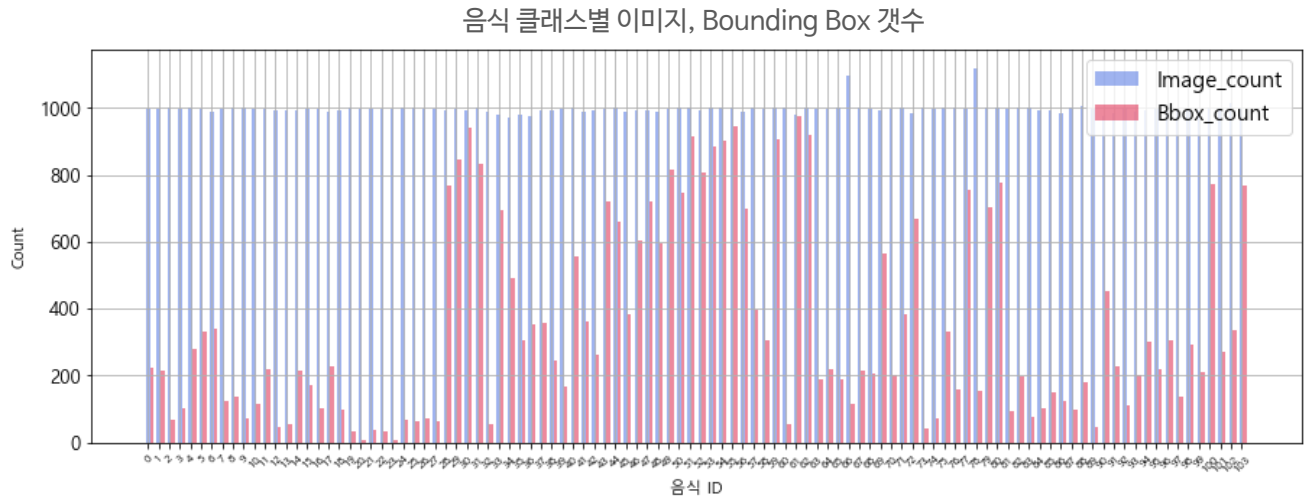
1. 웹 서비스 플로우 차트
2. 웹 서비스 기능

A. 모델개발

1. 데이터셋 수집(1/2)

AI HUB 한식 데이터

출처	AI Hub
음식 종류	105가지
각 클래스당 이미지 장수	1000장
Bounding box 정보	데이터의 35%에만 존재
특징	한 장의 사진에 한 종류의 음식만 존재



∴ bounding box가 없는 이미지는 우측 [그림3,4]처럼 bounding box를 이미지 전체 크기와 동일하게 설정

[그림3,4] AI HUB 한식 데이터 Bounding Box 정보를 포함한 예시 이미지

A-2. Object detection모델 선정 (1/5)

a. YOLOv5

YOLOv5는 4가지의 모델(YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x)로 소개되었으며
본 프로젝트에 적합한 모델을 찾기 위해 모델 종류와 image size를 달리해가며
학습시간 및 정확도를 비교해보았다.

	Case 1	Case 2	Case 3
모델	Yolov5s	Yolov5s	Yolov5m
Img-size	640	320	320
학습시간	약 4시간	2.214시간	3.428시간
mAP@0.5	0.8123	0.8047	0.8103
mAP@0.5:0.95	0.7136	0.6921	0.7064

학습시간 ↓ mAP ↑

∴ 이후 학습은 학습 시간을 줄이기 위해 img-size는 320 또는 416으로 설정하였으며
모델은 AP가 높은 Yolov5m과 Yolov5l을 사용하였다.

A-2. Object detection모델 선정 (2/5)

b. Retinanet

환경	- Fizyr 오픈 소스 API 사용 - v0.5.1 - 구글 클라우드 서버
선정 이유	- 오픈 소스 API : 내부 코드 수정 용이함 - 정확도가 높음 (Yolov3 보다)

- Yolov5 보다 학습 속도가 느림
- 구글 클라우드 불안정 (Kernel 꺼짐)

CODE

```
backbone_model = backbone('resnet50')
```

backbone 모델 생성 (바탕이 되는 모델)

```
train_gen, valid_gen = create_generators(args,backbone_model.preprocess_image)
```

API 학습 Train, Valid 데이터 생성

```
training_model.fit_generator(train_gen, steps_per_epoch=args.steps
                             ,epochs=args.epochs, verbose=1, callbacks=callbacks)
```

설정 값에 따라 학습 수행

Retinanet v0.5.1	
음식 종류	9가지
각 음식별 사진 장수	100~300장
img-resize	min-800, max-1333
batch / epoch	Batch 4, epoch 50
학습시간	약 6시간
loss	0.651
regression_loss	0.52
classification_loss	0.12

[그림 8,9] Retinanet 테스트 결과 이미지

A-2. Object detection모델 선정 (3/5)

c. TF api

colab

환경	- TensorFlow Object Detection API - Google Colab
선정 이유	- RetinaNet 보다 높은 정확도 - D1 ~ D6 다양한 모델 선택

- 학습 환경 세팅 어려움 (서버, 팀원 환경)
- YOLOv5 보다 학습 방법이 복잡함

CODE

```
example = tf.train.Example(
    features=tf.train.Features(feature=feature_dict))
```

데이터 정보를 TFRecord 형식에 맞게 바꿈

원본 학습 데이터

kfood_train.record kfood_valid.record

원본 데이터를 record파일로 변환

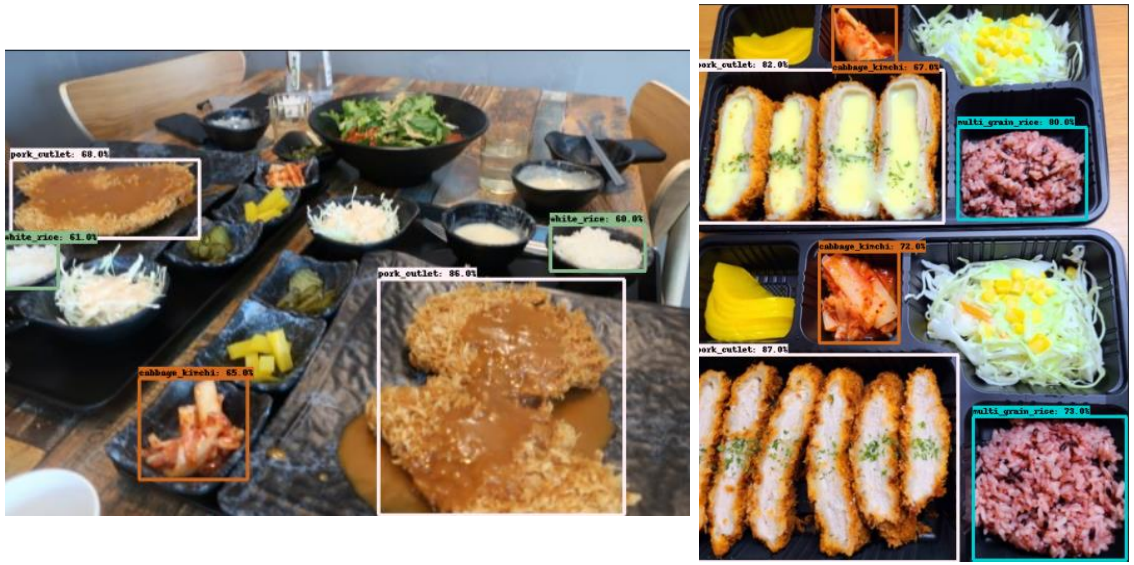
```
'efficientdet-d1': {
    'model_name': 'efficientdet_d1_coco17_tpu-32',
    'base_pipeline_file': 'ssd_efficientdet_d1_640x640_coco17_tpu-8.config',
    'pretrained_checkpoint': 'efficientdet_d1_coco17_tpu-32.tar.gz',
    'batch_size': batch
```

학습할 모델 정보 입력

```
!python models/research/object_detection/model_main_tf2.py #...학습 설정값 중략...
```

모델 학습 진행

Efficientdet_D1 모델	
학습 음식 종류	32가지
각 음식별 사진 장수	300장 이상
img-size	640 x 640
batch / epoch	Batch 8, epoch 50
학습시간	약 15시간
loss	0.412



[그림10, 11] TF api Efficientdet_D1모델 테스트 결과 이미지

A-2. Object detection모델 선정 (5/5)

e.선정 결과

YOLOv5

- 학습 속도 빠름
- 노트북 GPU를 이용해서 학습 가능
- 오픈 소스로 내부 코드 열람/수정 가능
- 환경 설정 간단

Retinanet

- 노트북GPU를 이용하기엔 무거움
- 오픈 소스로 내부 코드 열람/수정 가능
- YOLOv3보다 정확도 높음

TF API

- 환경 설정 번잡
- 제공되는 여러 모델을 사용할 수 있음
- Tfrecord 학습 데이터를 따로 만들어줘야 하는 번거로움

Google Vision AI - AutoML Vision

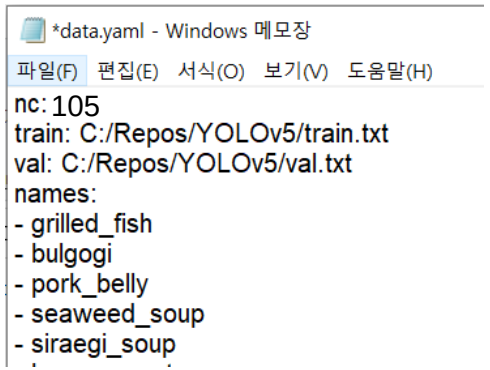
- GCP(Google Cloud Platform) 이용
- API 이용 간단
- 내부 코드 열람 불가
- 무료 크레딧 소진 후 비용 지불(유료)

최종 선택 모델

A-3. 모델 학습(1/2)

'A-2.object detection 모델 선정' 단계를 바탕으로 본 프로젝트에 가장 적합한 모델은 학습 속도가 빠르며 코드 수정이 가능한 YOLOv5라고 판단하여 AI HUB 한식 데이터를 YOLOv5m과 YOLOv5l모델에서 각각 학습시켜보았다.

1 학습과정



train data경로를 담고 있는 train.txt, validation data 경로를 담고 있는 val.txt, 클래스 개수, 각 클래스의 이름, train.txt와 val.txt의 경로가 담겨있는 data.yaml을 생성한 후 아래 명령어를 통해 학습을 시작한다.

CODE

```
!python train.py --batch 8 --epochs 500 --data ../data.yaml
--cfg ./models/yolov5m.yaml --weights yolov5m.pt  - img 416
```

A-3. 모델 학습(2/2)

2 학습결과 및 피드백

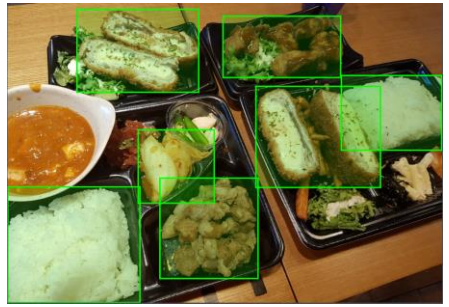
각 클래스별로 테스트 이미지를 5개씩 확보한 후 아래와 같이 테스트 결과를 정리하였다.

소분류	영어이름	0이미지	1이미지	2이미지	3이미지	4이미지
갈비구이	grilled_ribs	O 0.46	O 0.35	X	X 불고기	X 장어구이
고등어구이	grilled_mackerel	X	O 0.68	O 0.73	O 0.24 0.40	O 0.62
곱창구이	grilled_giblets	△ 0.2 채소까지	△ 0.84 채소까지	X 삼겹살로인식	O 0.77	X

문제점	- Bounding box 맞지 않음(그림12) - 여러 음식을 동시에 인식 못함(그림13)
원인	학습 데이터셋에 문제가 있음 (과도한 확대샷, 부정확한 bbox 정보)
개선방안	학습 데이터 직접 제작 - 50가지 음식, 각 300장씩 - 여러 음식이 나온 이미지 위주로 크롤링(그림14,15) - Bounding Box 제작

[그림12] bbox부정확

[그림13] 여러 음식 동시에 인식 실패



[그림14,15] 한 장의 이미지에 여러가지 음식 존재

A-4. 데이터셋 재수집 및 학습(1/3)

	AI HUB 한식 데이터	Custom Data 50 (푸딘카 제작 데이터)
음식 클래스 가짓수 * 각 클래스당 이미지 장수	105가지 * 1000장	50가지 * 300장
Bounding Box 존재여부	△ (bounding box 정보가 없는 이미지도 다수 존재)	O
특징	하나의 이미지에 하나의 음식만 존재	하나의 이미지에 여러가지 음식 존재

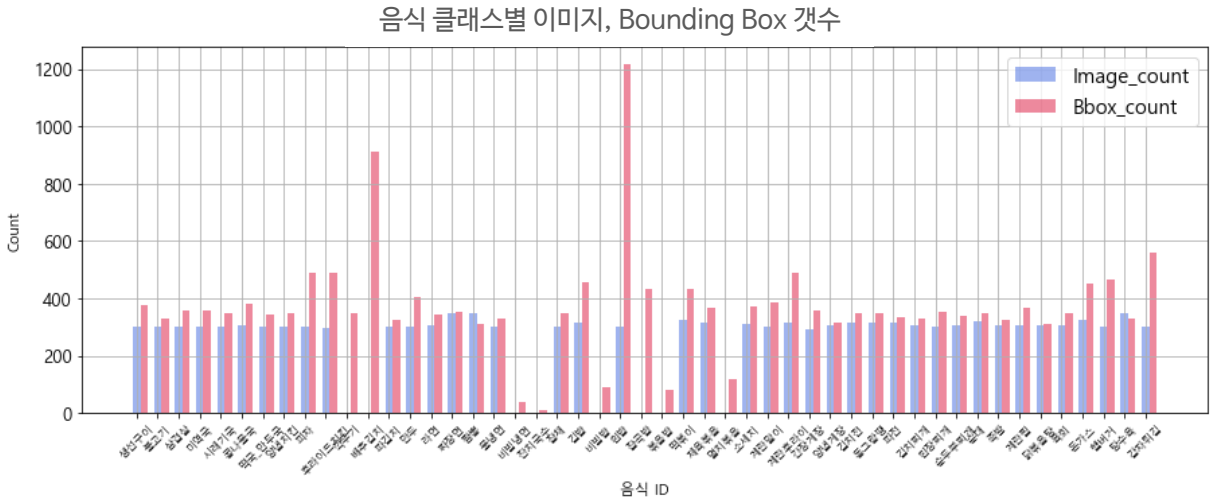
국민영양통계 다빈도 음식 데이터를 토대로 50가지 선정

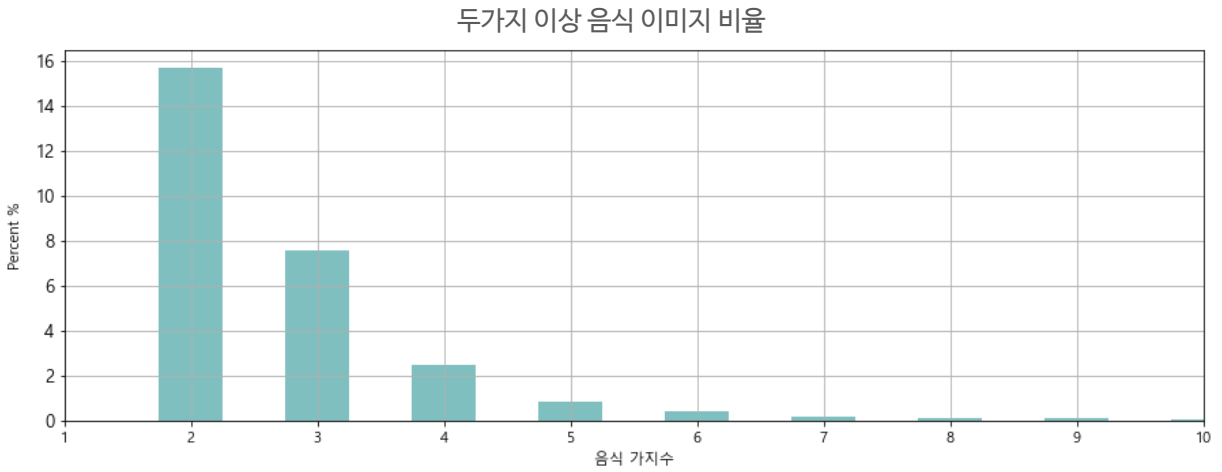
A-4. 데이터셋 재수집 및 학습(2/3)

Custom Data 50

출처	구글 크롤링 & bounding box 직접 제작
음식 종류	50가지
각 클래스당 이미지 장수	300장
특징	한 장의 사진에 여러가지 음식 존재

[그림13,14] Custom Data 50 Bounding Box 정보를 포함한 예시 이미지



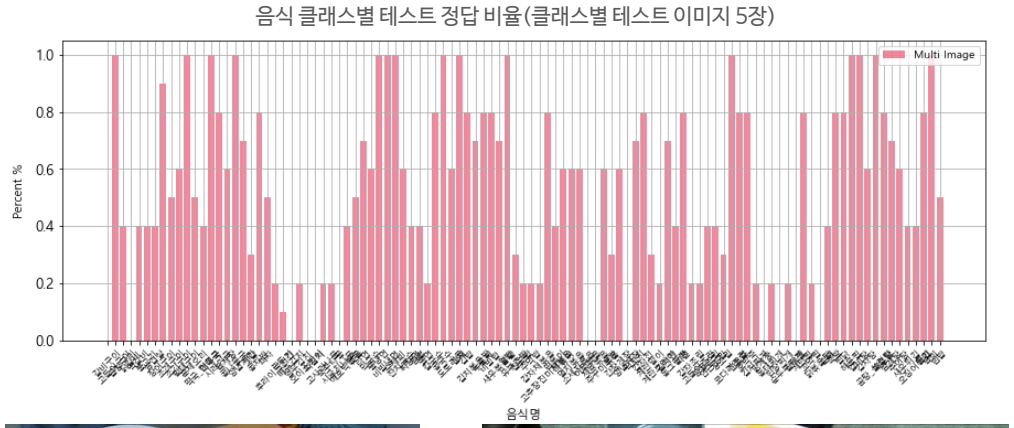


A-4. 데이터셋 재수집 및 학습(3/3)

재수집한 데이터(Custom Data 50)와 기존 데이터(AI HUB 한식 데이터)에 대한 테스트 결과는 아래와 같다.

AI HUB 한식 데이터

YOLOv5l에 학습시켰을 때 여러가지 음식을 동시에 찾아내는데 실패했다. (정답 비율 **51.3%**)



[그림15,16] AI HUB 한식 데이터 테스트 결과 이미지

Custom Data 50

Better

YOLOv5l에 학습시켰을 때 여러가지 음식을 동시에 잘 찾아내면서 높은 정확도를 보여주었다. (정답 비율 **89.8%**)

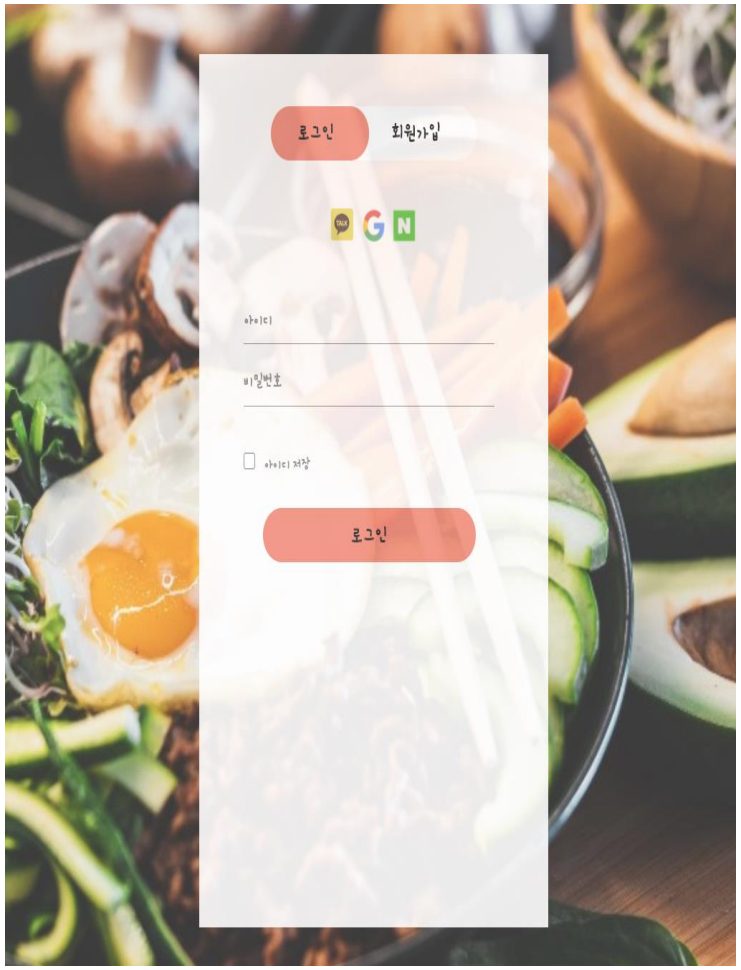
[그림17,18] Custom Data 50 테스트 결과 이미지

5. 프로젝트 결과

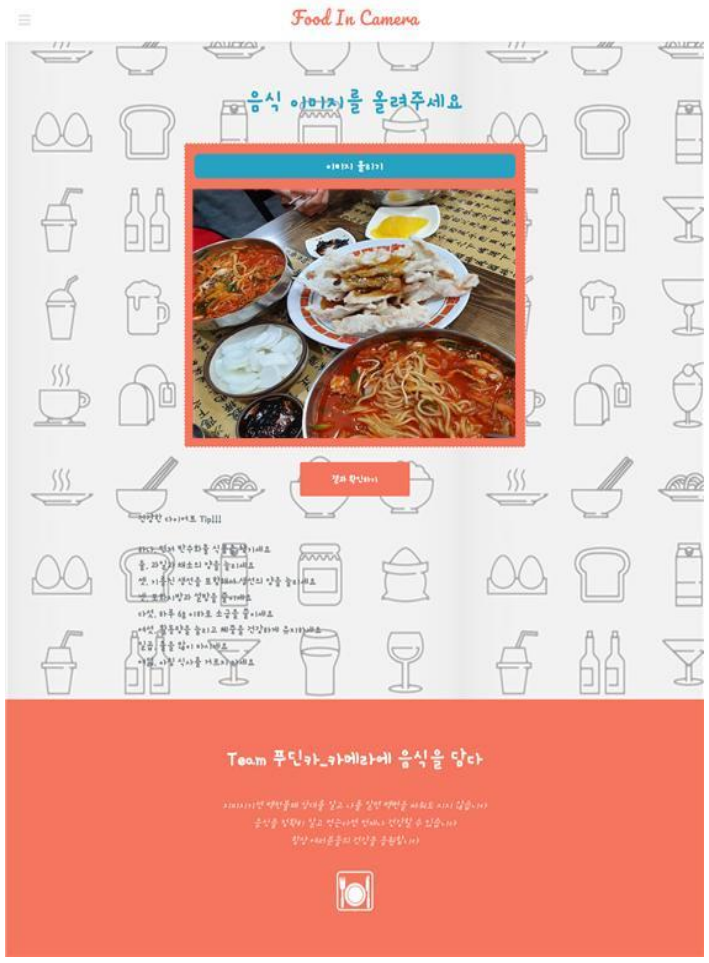
1. 웹 서비스 이용 예시(1/3) *https://www.foodinca.tk*

1) 웹

① 푸딘카에 접속합니다.



② 로그인을 합니다.

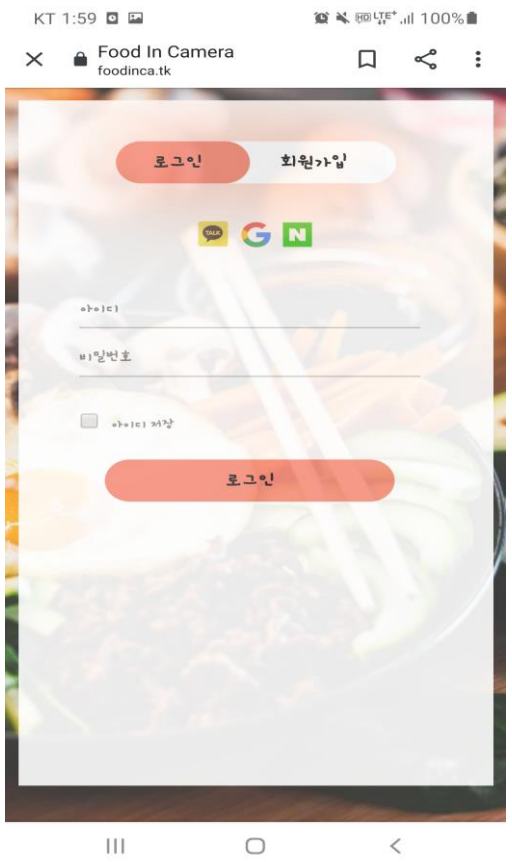


③ 오늘 먹은 음식 사진을 업로드합니다.

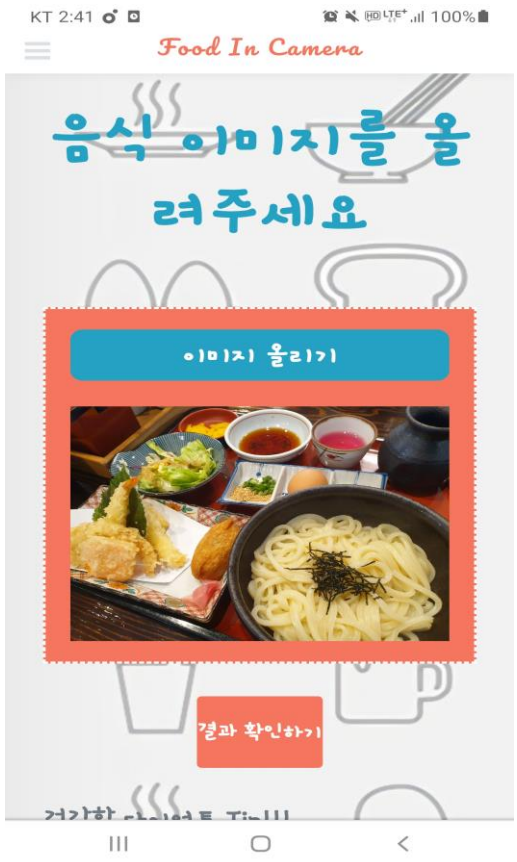
1. 웹 서비스 이용 예시(3/3) https://www.foodinca.tk

2) 모바일

① 푸딘카에 접속합니다.



② 로그인을 합니다.



③ 핸드폰으로 맛있는
앞의 음식을 찍습니다.

④ 맛있는 결과를 보고
음식을 맛있게 먹습니다.

2. 보완사항 및 향후계획

맞춤 음식 추천 서비스
히스토리를 탐색하여 사용자에게 부족해 보이는 영양 요소를 파악하여 해당 영양 요소를 보충해줄 수 있는 음식을 추천

04

사용자 편의 기능
사용자가 직접 음식을 추가하거나 음식의 양을 선택할 수 있게 하여 보다 정확한 칼로리와 영양 수치 제공

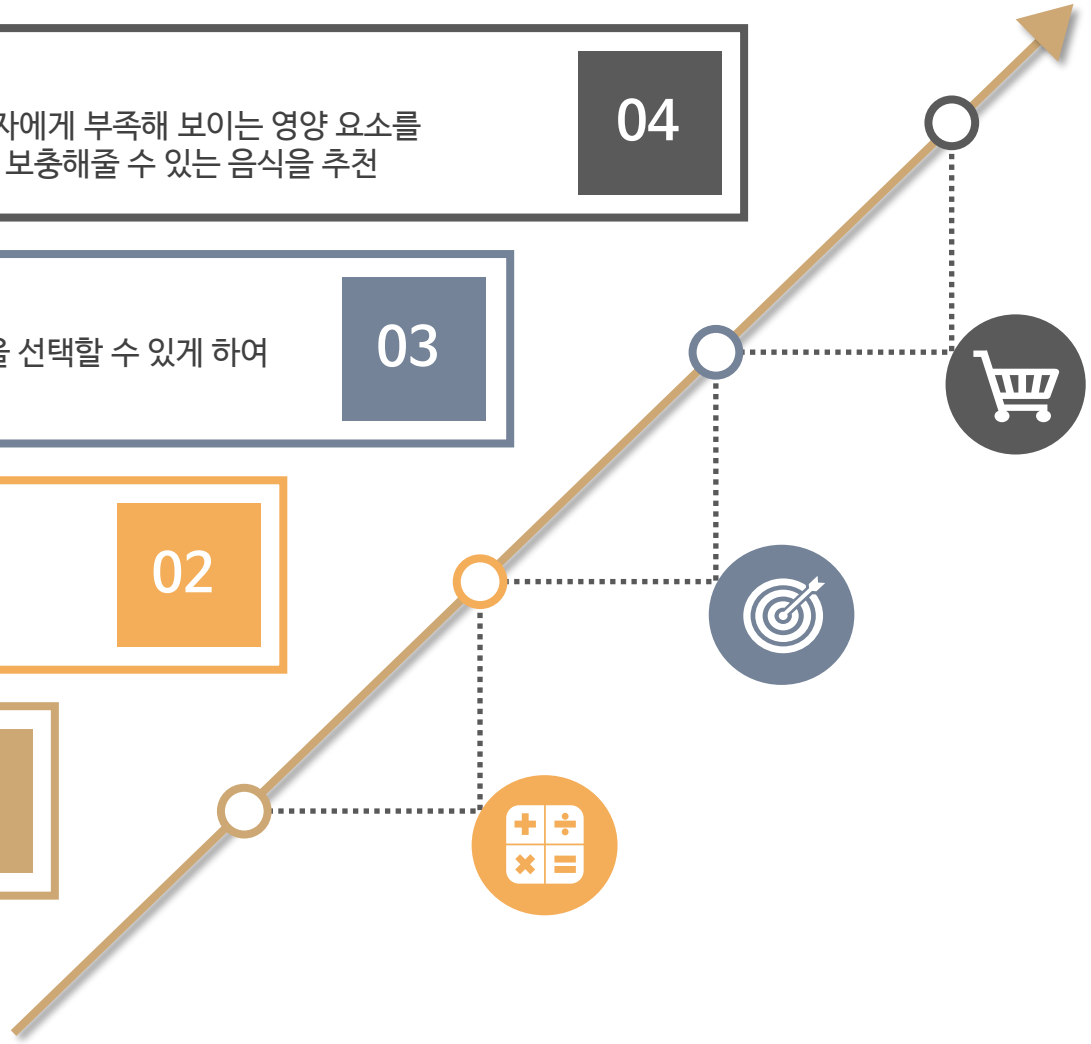
03

분류의 정확도
비슷한 모양과 색을 지닌 음식들의 학습 데이터 수를 증가하여 정확도 개선

02

데이터의 다양성
100가지 이상의 음식 이미지 데이터를 수집하여 인식하는 음식의 종류 증가

01



6. 개발환경

Model	
OS	Windows 10
GPU	GTX 1660Ti
Cuda	v10.1
Cudnn	v7.6.5
Pytorch	V1.7.0(gpu)
Tensorflow	V2.3.0(gpu)
Yolo	Yolov5

WEB	
Cloud	AWS
Ubuntu	18.04
Server	Apache / 2.4.29 (Ubuntu)
PHP	7.4.12
MySQL	5.7.32

7. 관련 논문 및 레퍼런스

dataset

- https://storage.googleapis.com/openimages/web/index.html (Open Images Dataset V6)
  http://kface.kist.re.kr/#/kfoodIntro?bbstypecd=kfoodIntro (AI HUB 한식 이미지 데이터베이스)
- https://github.com/tzutalin/labelImg (labelImg : 라벨링 작업을 위한 프로그램)
- https://www.data.go.kr/data/15050912/fileData.do (공공데이터포털 : 농림수산식품교육문화정보원_칼로리 정보)

Model Train

- https://github.com/fizyr/keras-retinanet (retinanet - github)
- https://github.com/tensorflow/models/tree/master/research/object_detectio (Tensorflow Object Detection API - github)
- https://github.com/ultralytics/yolov5 (yolov5 - github)
- https://inf.run/daGJ (Object Detection 인프런 강의)

Web

- https://github.com/ kamranahmedse/developer-roadmap(Web develop)
- https://pixlr.com/kr/x/ (이미지 편집)
- https:// www.youtube.com/channel/UCax1DP6hqZowNWF2lquKk0w(웹퍼블리싱 기술)

Thank you

https://www.foodinca.tk