****

**감정분석을 통한 메뉴 추천 서비스**

**12기 A반 1조**

**노선호 강민지 김한솔 성유기 양수환 한길영**

**목차**

1. **프로젝트 개요**
   1. 프로젝트 소개
   2. 프로젝트 주제 선정 배경
   3. 기존 서비스 및 관련 논문
2. **본론**
   1. 전체 구조도
   2. 구현 과정
      1. 실시간 감정분석 후 음식추천
      2. 음식 랭킹 순위
      3. 현 위치 기반 음식점 리스트 제공
      4. 앱 개발
      5. 사용 프로그램
3. **결론**
   1. 기대효과
   2. 한계점 및 발전 방향
4. **조원 소개**
5. **참고 문헌**

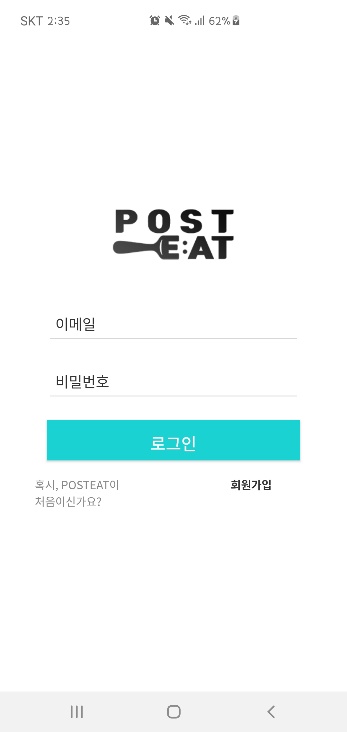
# **프로젝트 개요**

## **프로젝트 소개**

본 프로젝트는 현대인이 일상 속에서 가장 많이 그리고 자주 하는 고민 중 하나는 "오늘 뭐 먹지?"라고 해도 과언이 아니다. 그러나 개인이 음식에 대한 선호가 분명하더라도 더 다양한 음식, 입맛에 맞는 새로운 음식을 먹고 싶기 때문에 하나의 메뉴를 정하기까지 시간뿐 아니라 작거나 큰 감정소비까지 같이 이루어진다. 우리는 빠르고 간단하게 식사 메뉴를 정하고 이 메뉴에 대해 대부분의 선호를 이끌고자 하였다.

‘POSTEAT’은 이처럼 메뉴 결정에 고민이 많은 현대인을 위한 얼굴 감정 분석 기반 음식 추천 서비스이다. 감정분석을 통해 사용자의 감정을 분석하고 선호도를 리뷰수를 통해 랭킹을 정하여 사용자 맞춤 메뉴를 제공한다. 또한 현재 위치를 기반으로 음식점을 추천한다.

이처럼 ‘POSTEAT’은 음식을 고르는 시간과 스트레스를 제거하고 온전히 식사에 집중할 수 있는 것을 목표로 하고 있으며, 더 나아가 음식을 고르는 시간이 즐거운 시간이 될 수 있는 서비스를 제공한다.

[그림1] [그림2]

## **프로젝트 주제 선정 배경 및 필요성**

2019년 12월 이후, COVID-19라는 범유행 전염병으로 ‘비대면’이 일상화되면서 집에서 안전하게 먹을 수 있는 배달음식 수요가 급증하고 있다. 배달음식 수요의 증가에 따라 현대인들의 메뉴 선정에 어려움을 겪고 있다. 메뉴 선정에 겪는 문제점은 다음과 같다.

1. 메뉴를 고르는데 많은 시간을 소비함
2. 음식 메뉴를 고르는데 어려움
3. 음식을 고르는 행위가 주는 스트레스

상기한 유형의 문제점은 많은 사람들이 겪고 있는 문제이며, 이는 메뉴를 선정하는 사람의 시간 낭비, 메뉴 선정 시 스트레스를 초래할 수 있다. 이러한 문제점은 다음과 같이 정의된 기능 구현으로 해결할 수 있을 것이라 판단한다.

[메뉴를 고르는데 소비하는 시간]

서울대학교병원 가정의학과 연구팀의 국민건강영양조사를 바탕으로 19세이상 남녀 4,624명을 대상으로 하루 식사 횟수를 조사하였는데, 2회 이하 9%(429명), 3회 40%(1830명), 4회 35%(1636명), 6회 이상(16%)이라고 한다. 우리가 평균적으로 일주일에 20번 이상의 식사를 하는데‘무엇을 먹을지’고민 빈도는 생각보다 많다.

[음식 메뉴 선택에 있어서 어려움]

성인 남녀 3689명을 대상으로 ‘평소 결정장애를 절감하는 때는 언제’인지 설문을 하였을 때, 55.7%가 음식 메뉴를 선택할 때라고 답하였다.

[음식을 고르는 행위가 주는 스트레스]

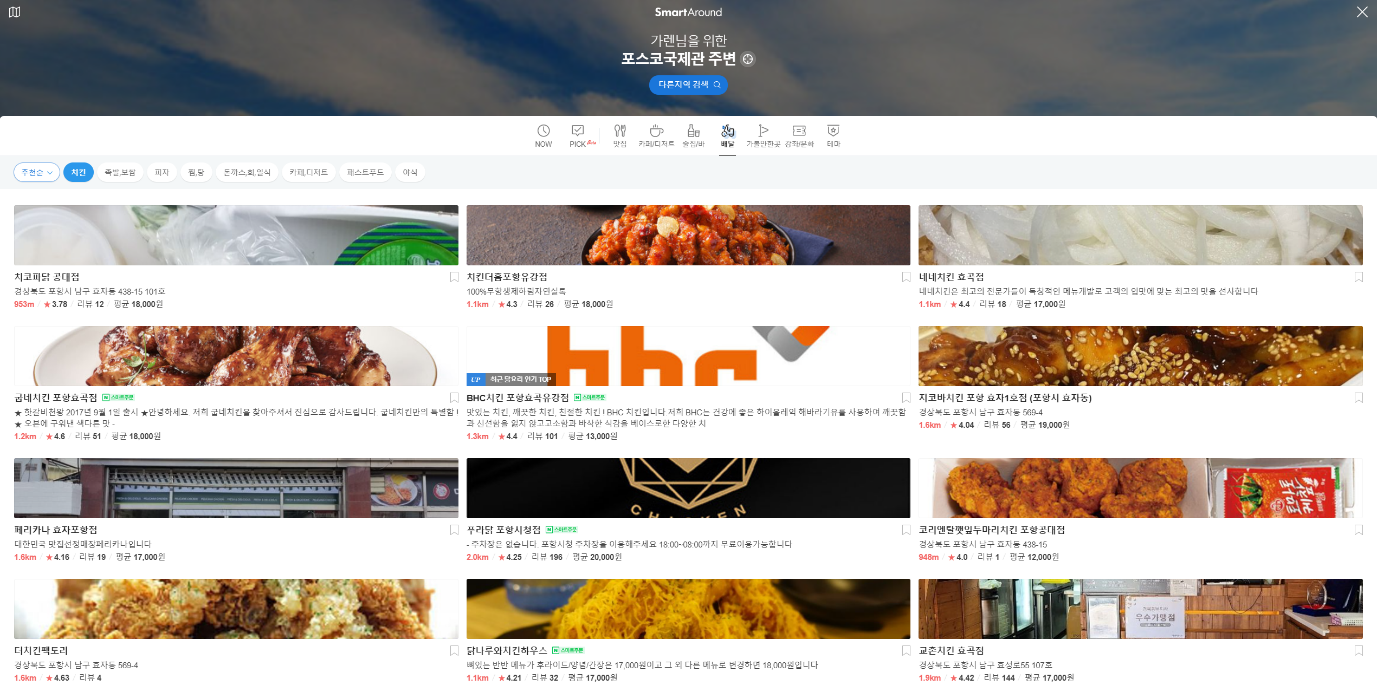
작은 문제에도 결정의 어려움을 느끼는 이유는 선택지가 너무 많기 때문이라고 한다. 너무 많은 선택지가 주어질 경우, 오히려 결정을 내리기가 힘들어지게 만들고, 적은 선택지를 가졌을 때보다 더 좋지 않은 선택을 하거나 심지어 선택 자체를 포기하는 현상이 발생한다. 사람들에게 선택권을 많이 준다는 것은 선택 과정까지 과도한 정보 수집과 다른 것을 포기한다는 스트레스를 유발할 수도 있기에, 최종선택 후에도 높은 만족도를 주지 못하게 된다.

* 기능 1. 감정분석 후 음식추천 : CNN VGG -13 알고리즘을 통하여 사진 속 얼굴 인식 및 감정분석을 하여 CBF 추천 알고리즘을 통하여 해당하는 감정에 맞는 음식 추천한다.
* 기능 2. 음식랭킹순위 지정 : 앞에서 추천된 음식들을 포스트잇 앱 리뷰수를 통하여 음식 랭킹 1~5위까지 순위를 나타낸다.
* 기능 3. 음식 선정시 현 위치 기반의 음식점 리스트 젝공: 앞선 랭킹순위를 통해 나타난 음식들 중 사용자가 음식을 선정하면 카카오지도 API를 이용하여 사용자 현 위치 기반의 음식점 리스트를 제공한다.

상기한 기능은 CNN 기반 감정인식 기술과 CBF 추천 알고리즘을 통해 성공적으로 구현되었으며, 해당 내용은 2장에 상세히 기술되어 있다.

## **기존 서비스 및 관련 연구**

ㅇ 기존 서비스

네이버 스마트어라운드(Smart Around) : AI로 업그레이드된 네이버 지도에서 사용자를 파악하고 위치, 시간대, 성별, 연령대 등 실제 구매전환율이 높은 사용자에게 업체를 노출하여 오프라인 상점과 고객을 연결해 주는 역할을 한다. 그리고 사람에 맞춤 검색 결과로 더욱 정확한 플레이스 정보를 사용자에게 제공하고 업체에게는 구매전환율이 높은 고객에게만 집중적인 홍보를 할 수 있다.

[그림3] 네이버 스마트어라운드

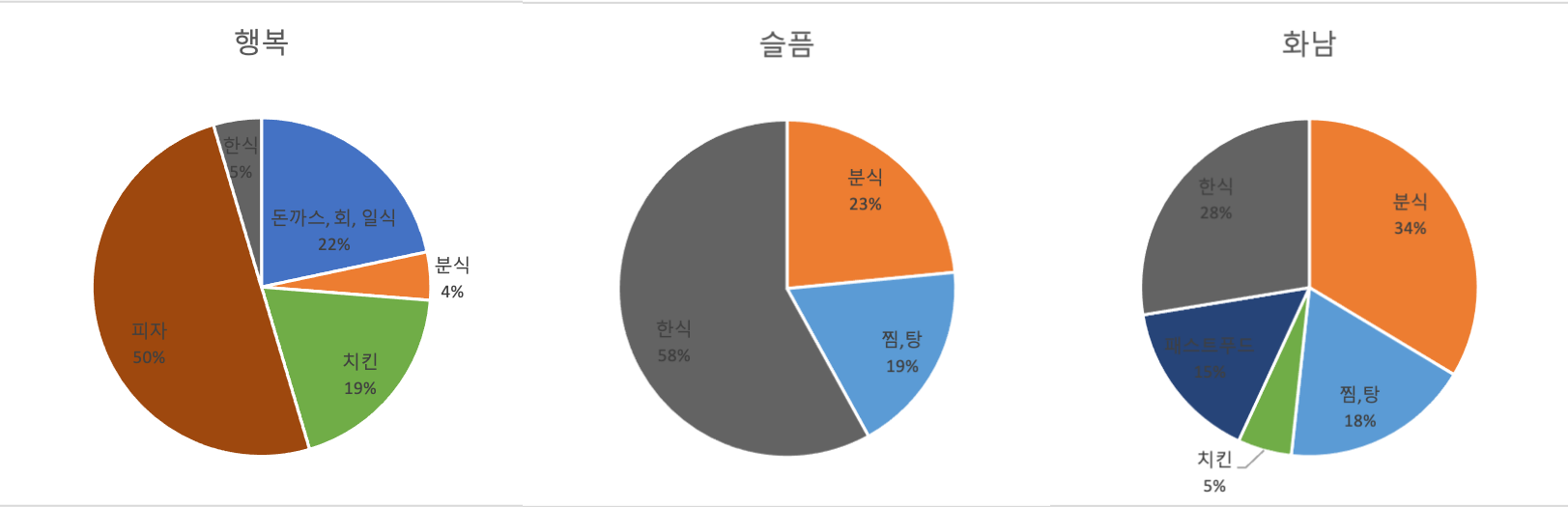
(출저 : <https://s.search.naver.com/p/around/search.naver>)

카카오맵 맛집 (카카오)은 모임, 데이트 등 목적 별 장소를 추천한다. 그리고 T맵 미식로드 (SK 텔레콤)는 1,800만 명이 이용하는 T맵을 기반으로 지역, 성별, 나이 등 개인화에 맞는 음식점을 추천한다. 그 외 망고플레이트(여기어때), 식신 등 주요 맛집 추천 서비스를 살펴보았을 때 사람의 감정을 고려하는 서비스는 존재하지 않았다.

### 관련 논문

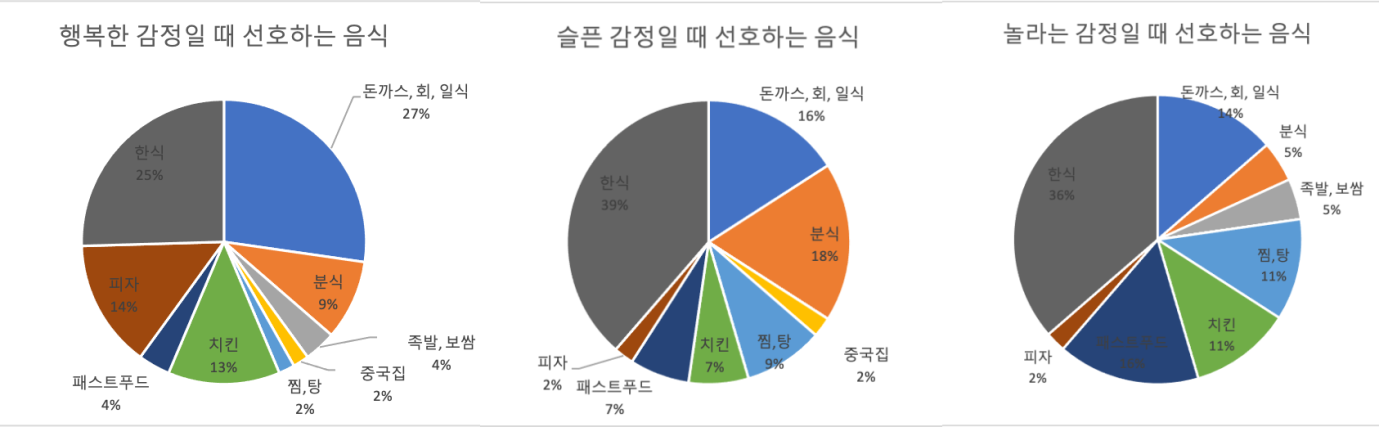
ㅇ 감정에 따른 음식 선호도

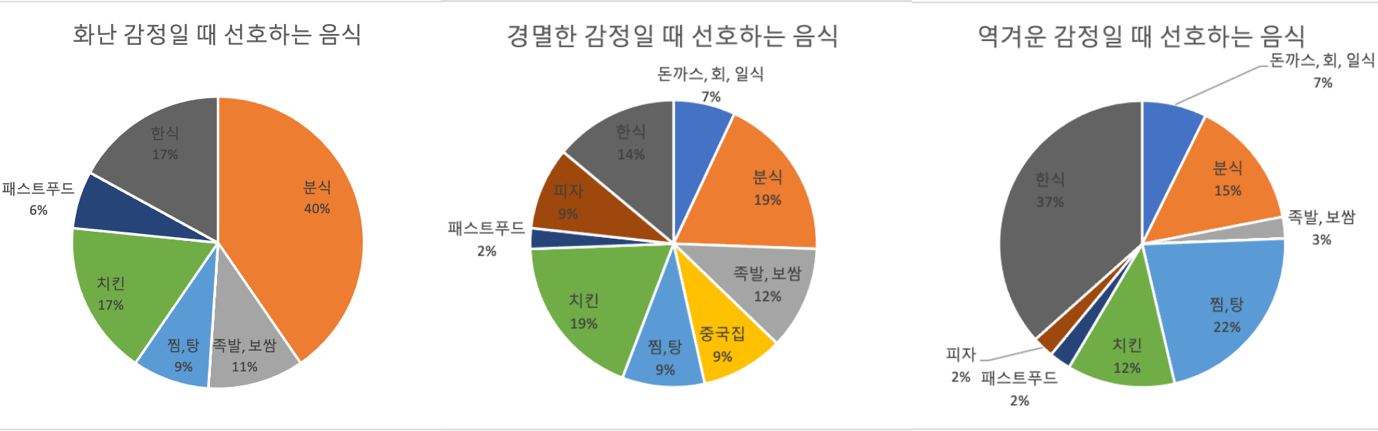
기쁨, 슬픔, 즐거움, 분노, 편안함 및 우울의 감정에 따라 음식 선호에 차이가 있는 것으로 나타났다. 기쁨에서는 피자와 스파게티, 아이스크림, 케이크를 선호하였으며, 슬픔에서는 술, 초콜릿, 음료, 찌개 및 백반을 선호하였다. 즐거움에서는 피자와 스파게티, 아이스크림, 간식, 분식을 선호하는 것으로 나타나 기쁨의 감정에서 나타난 선호와는 차이가 있었다. 분노에서는 술, 매운 음식, 분식, 찜 및 볶음류 순으로 다른 감정보다 매운 음식의 선호가 높은 것으로 나타났다. 편안함에서는 음료와 찌개 및 백반, 아이스크림, 간식의 선호가 높은 것으로 나타난 반면, 우울에서는 초콜릿, 음료, 술과 아이스크림을 선호하는 것으로 나타났다. 모든 감정에서 음식의 맛과 향이 가장 중요한 선호속성으로 나타났으며, 맛과 향, 외관의 선호속성은 부정적 감정에 비해 긍정적 감정인 기쁨, 즐거움과 편안함에서 높았다. 질감의 속성은 다른 감정에 비해 즐거움, 편안함, 우울, 분노의 감정에서 높았으며, 음식의 온도속성은 긍정적인 감정보다 부정적인 감정에서 더 영향을 미치는 것으로 나타났다. 음식에 대한 과거의 경험이나 이미지는 기쁨, 즐거움, 편안함과 우울에서 높았다. 레스토랑 선택에서는 기쁨의 감정에 따라 패밀리 레스토랑, 이탈리안 레스토랑, 뷔페, 호텔 레스토랑을 선택하는 것으로 나타났다. 슬픔의 감정에서는 주점, 카페, 한식 레스토랑 순으로 나타났으며 다른 감정에 비해 가고 싶지 않다고 답한 응답자가 가장 많았다. 즐거움에서는 패밀리 레스토랑, 이탈리안 레스토랑 순으로 기쁨의 감정과 유사하였으나 카페, 분식점, 주점의 선택이 더 높았다. 분노에서는 주점, 분식점, 카페, 구이 전문점 순으로 나타났으며 매운 음식 전문점의 빈도가 다른 감정보다 높았다. 편안함에서는 카페, 한식 레스토랑, 패밀리 레스토랑의 순으로 나타났으며 우울에서는 카페, 주점, 구이 전문점 순으로 나타났다.

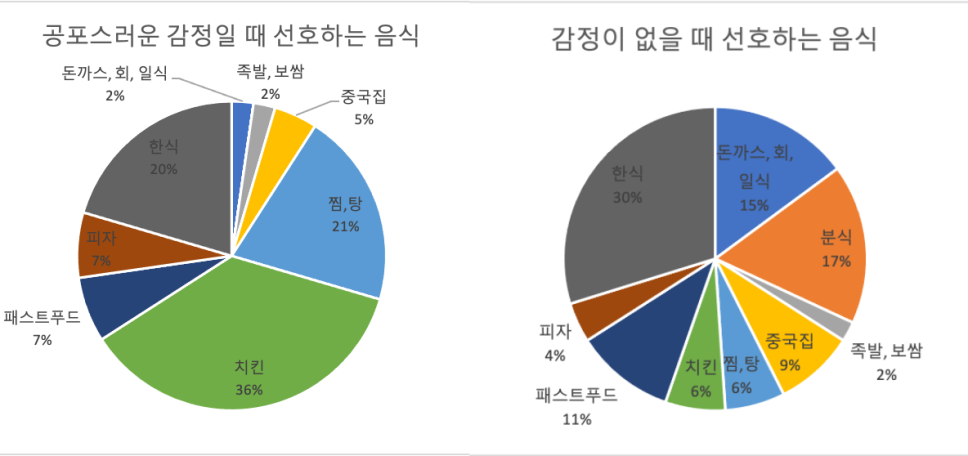


[그림4] 감정에 따른 음식 선호도 논문 결과

본 프로젝트에서는 8개의 감정과 9개의 카테고리의 연관성을 맵핑하고자 한다. 2030연령대 60명을 대상으로 감정에 따른 음식 선호도 설문조사를 진행한 결과는 다음과 같다.







[그림5] 감정에 따른 음식 선호도 설문조사 결과

논문에 나타난 행복, 슬픔, 화남에 대한 감정에 따른 음식 선호도가 설문조사에서도 같은 결과를 보이고 있다.

# **본론**

## **전체 구조도**

****

[그림6] POSTEAT 프로세스

## **구현 과정**

### **실시간 감정 분석 후 음식 추천**

#### 기술 소개

CNN VGG -13 알고리즘을 통하여 사진 속 얼굴 인식 및 **이미지의 감정이 분류가 되면 각 감정에 선호하는 음식을 추천해 주는 알고리즘을 구현하였다.**

[CNN VGG – 13]

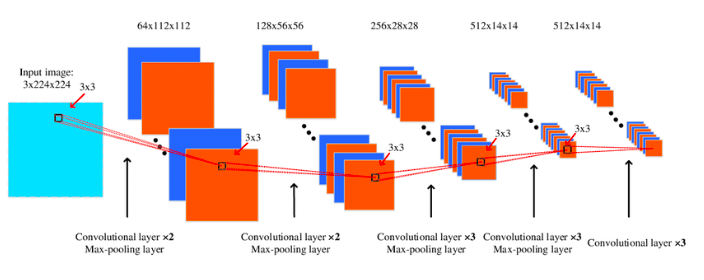
인공 신경망(Artificial Neural Network, ANN)은 인간의 뇌가 감각기관을 통해 입력 받은 정보를 토대로 정보를 인식하는 과정을 모사한 계산 모델이다. ANN은 분류 및 회귀 문제에 적합한 특성을 가지고 있다. 최근 ANN을 깊은 구조로 쌓은 심층 신경망(Deep Neural Network, DNN)이 영상인식, 음성인식, 자연어처리와 같은 다양한 분야에서 뛰어난 성능을 보이고 있다. 이중 컨볼루션 신경망(Convolutional Neural Network, CNN)은 격자 모양의 위상을 가지는 데이터를 처리하기 위한 특수한 종류의 ANN이다. 초기의 CNN은 손글씨 인식을 위해 제안되었으나, 현재는 다양한 분야의 영상 인식에 널리 사용되고 있다 .

CNN 모델은 컨볼루션 레이어와 풀링 레이어로 이루어져 있으며, 이를 통해 데이터로부터 공간적인 특징을 추출할 수 있다. 모든 입력 뉴런들이 모든 출력 뉴런들과 연결되어 있는 ANN 모델과 달리, CNN 모델의 컨볼루션 레이어에서 출력 뉴련은 인접한 입력 뉴런하고만 연결되어 있다. 이러한 특징을 희소 연결이라 하며, 이를 통해 CNN은 입력 데이터의 공간적 특징을 유지하고, 비교적 적은 개수의 매개변수를 필요로 하게 된다. 기본적인 CNN 모델에서 컨볼루션 레이어 이후에는 비선형 활성화 함수가 적용되며 풀링 레이어가 뒤따른다. 풀링 레이어는 레이어의 총 출력을 요약하는 기능을 하며 다운샘플링으로 볼 수 있다. 이러한 풀링 레이어 또한 CNN 모델에 노이즈 및 변화에 강인한 특성을 부여하게 된다. 이러한 특성으로 인해 CNN 모델은 이미지 데이터로부터 유의미한 특징을 추출하는 능력을 가진다. 본 논문에서는 이러한 특성을 활용하여 표정기반 감정 상태 추론을 위해 CNN 모델을 사용하여 입력 영상으로부터 특징을 추출하였다.

CNN은 특징 추출에 뛰어나기 때문에 이미지 분류에서 높은 성능을 보이지만 분류하고자 하는 해당 물체의 위치를 검출하는 능력은 없다. 이는 CNN 모델에서 분류기 역할을 하는 최종 완전 연결 (fullyconnected) 레이어 때문이다. 앞서 살펴본 것과 같이 CNN 모델은 다수의 컨볼루션 레이어와 풀링 레이어를 통해 입력 데이터로부터 특징 텐서를 출력한다. 이렇게 추출된 특징 텐서는 3D 형태를 가지며 공간적인 특성을 유지하고 있는 상태이지만, 최종 완전 연결 레이어의 입력으로 들어가는 과정에서 2D 형태의 벡터로 변형된다. 이 과정을 통해 특징 텐서는 공간적 정보를 잃게 된다.

최근 제안된 weakly-supervised object localization 방식은 컨볼루션레이어에서 출력한 특징 텐서를 활용하여 물체의 위치를 추정한다. 분류 과정에서 입력 데이터의 어느 영역이 주로 활성화되었는지를 기준으로 영상 내에 해당 물체가 위치함을 추정할 수 있을 것이라는 가정이다. 이때 컨볼루션 레이어에서 출력된 특징 텐서는 여러 개의 채널을 가진 특징 지도로 생각할 수 있다. 따라서 주어진 특징 지도를 이용하면 CNN의 최종 완전 연결 레이어에서 입력 영상의 어떠한 영역이 주로 영향을 끼쳤는지 해석이 가능하다. 본 논문에서는 특징 지도를 이용하여 영상 내 물체 위치 추정 방식을 응용하여 입력 영상 내 얼굴 영역 검출을 수행하였다.

CNN 초기 모델로 16개의 레이어를 가지고 있다. 2014년 당시에는 엄청나게 Deep한 모델로 평가되었다. 모델은 Conv, Pooling, FC layer만으로 이뤄져 있기 때문에 직접 코딩하기 쉬운 장점이 있다. 단점은 파라미터 많기 때문에 학습이 매우 느리다는 점이다. 또한, 파라미터로 인하여 용량이 매우 크다는 단점도 있다. 이 부분은 다음 모델부터 개선되었다.



출처 : https://www.researchgate.net/figure/The-architecture-of-VGG-Base-It-contains-13-convolutional-layers-and-4-max-pooling\_fig4\_325062382

[그림7] CNN VGG-13

[Google vision API]

얼굴 이미지 감지, 이미지 크기 조정을 위해 Google vision API을 사용했다.



[그림8] Google Cloud Vision API

#### 구현 코드

**face\_crop0-checkpoint.py**

크롤링해서 얻은 연예인 사진에서 얼굴부분을 detecting, 자르기, 저장 등의 함수를 정의한다.

|  |
| --- |
| from googleapiclient import discovery  from oauth2client.client import GoogleCredentials  import sys  import io  import base64  from PIL import Image  from PIL import ImageDraw  from genericpath import isfile  import os  from oauth2client.service\_account import ServiceAccountCredentials  import tensorflow as tf  import matplotlib.pyplot as plt  from keras.models import load\_model    gpus = tf.config.experimental.list\_physical\_devices('GPU')  if gpus:  try:  # Currently, memory growth needs to be the same across GPUs  for gpu in gpus:  tf.config.experimental.set\_memory\_growth(gpu, True)  logical\_gpus = tf.config.experimental.list\_logical\_devices('GPU')  print(len(gpus), "Physical GPUs,", len(logical\_gpus), "Logical GPUs")  except RuntimeError as e:  # Memory growth must be set before GPUs have been initialized  print(e)  import keras  import cv2  import numpy as np  import os  import math  NUM\_THREADS = 10  MAX\_RESULTS = 1  IMAGE\_SIZE = 96,96    class FaceDetector():  def \_\_init\_\_(self):  # initialize library  #credentials = GoogleCredentials.get\_application\_default()  scopes = ['https://www.googleapis.com/auth/cloud-platform']  credentials = ServiceAccountCredentials.from\_json\_keyfile\_name(  '../sunho-299311-f8d55ae6b161.json', scopes=scopes)  self.service = discovery.build('vision', 'v1', credentials=credentials)  #print ("Getting vision API client : %s" ,self.service)    #def extract\_face(selfself,image\_file,output\_file):    def detect\_face(self,image\_file):  try:  with io.open(image\_file,'rb') as fd:  image = fd.read()  batch\_request = [{  'image':{  'content':base64.b64encode(image).decode('utf-8')  },  'features':[{  'type':'FACE\_DETECTION',  'maxResults':MAX\_RESULTS,  }]  }]  fd.close()    request = self.service.images().annotate(body={  'requests':batch\_request, })  response = request.execute()  if 'faceAnnotations' not in response['responses'][0]:  print('[Error] %s: Cannot find face ' % image\_file)  return None    face = response['responses'][0]['faceAnnotations']  box = face[0]['fdBoundingPoly']['vertices']  left = box[0]['x']  top = box[1]['y']    right = box[2]['x']  bottom = box[2]['y']    rect = [left,top,right,bottom]    print("[Info] %s: Find face from in position %s" % (image\_file,rect))  return rect  except Exception as e:  print('[Error] %s: cannot process file : %s' %(image\_file,str(e)) )    def rect\_face(self,image\_file,rect,outputfile):  try:  fd = io.open(image\_file,'rb')  image = Image.open(fd)  draw = ImageDraw.Draw(image)  draw.rectangle(rect,fill=None,outline="green")  image.save(outputfile)  fd.close()  print('[Info] %s: Mark face with Rect %s and write it to file : %s' %(image\_file,rect,outputfile) )  except Exception as e:  print('[Error] %s: Rect image writing error : %s' %(image\_file,str(e)) )    def crop\_face(self,image\_file,rect,outputfile):  try:  fd = io.open(image\_file,'rb')  image = Image.open(fd)  crop = image.crop(rect)  im = crop.resize(IMAGE\_SIZE,Image.ANTIALIAS)  im.save(outputfile,"JPEG")  fd.close()  print('[Info] %s: Crop face %s and write it to file : %s' %(image\_file,rect,outputfile) )  except Exception as e:  print('[Error] %s: Crop image writing error : %s' %(image\_file,str(e)) )    def getfiles(self,src\_dir):  files = []  for f in os.listdir(src\_dir):  if isfile(os.path.join(src\_dir,f)):  if not f.startswith('.'):  files.append(os.path.join(src\_dir,f))    return files    def rect\_faces\_dir(self,src\_dir,des\_dir):  if not os.path.exists(des\_dir):  os.makedirs(des\_dir)    files = self.getfiles(src\_dir)  for f in files:  des\_file = os.path.join(des\_dir,os.path.basename(f))  rect = self.detect\_face(f)  if rect != None:  self.rect\_face(f, rect, des\_file)    def crop\_faces\_dir(self,src\_dir,des\_dir):    # training data will be written in $des\_dir/training  # validation data will be written in $des\_dir/validate    des\_dir\_training = os.path.join(des\_dir,'training')  des\_dir\_validate = os.path.join(des\_dir,'validate')    if not os.path.exists(des\_dir):  os.makedirs(des\_dir)  if not os.path.exists(des\_dir\_training):  os.makedirs(des\_dir\_training)  if not os.path.exists(des\_dir\_validate):  os.makedirs(des\_dir\_validate)    path,folder\_name = os.path.split(src\_dir)  label = folder\_name    # create label file. it will contains file location  # and label for each file  training\_file = open('training\_file.txt','a')  validate\_file = open('validate\_file.txt','a')    files = self.getfiles(src\_dir)  cnt = 0  for f in files:  rect = self.detect\_face(f)    # replace ',' in file name to '.'  # because ',' is used for deliminator of image file name and its label  des\_file\_name = os.path.basename(f)  des\_file\_name = des\_file\_name.replace(',','\_')    if rect != None:  # 70% of file will be stored in training data directory  if(cnt < 8):  des\_file = os.path.join(des\_dir\_training,des\_file\_name)  self.crop\_face(f, rect, des\_file )  training\_file.write("%s,%s\n"%(des\_file,label) )  # 30% of files will be stored in validation data directory  else: # for validation data  des\_file = os.path.join(des\_dir\_validate,des\_file\_name)  self.crop\_face(f, rect, des\_file)  validate\_file.write("%s,%s\n"%(des\_file,label) )    if(cnt>9):  cnt = 0  cnt = cnt + 1    training\_file.close()  validate\_file.close()    def getdirs(self,dir):  dirs = []  for f in os.listdir(dir):  f=os.path.join(dir,f)  if os.path.isdir(f):  if not f.startswith('.'):  dirs.append(f)    return dirs    def crop\_faces\_rootdir(self,src\_dir,des\_dir):  # crop file from sub-directoris in src\_dir  dirs = self.getdirs(src\_dir)    #list sub directory  for d in dirs:  print('[INFO] : ### Starting cropping in directory %s ###'%d)  self.crop\_faces\_dir(d, des\_dir)  #loop and run face crop      def main():  srcdir= '../img'  desdir = '../img'  detector = FaceDetector()    detector.crop\_faces\_rootdir(srcdir, desdir)    model = load\_model('fer2013plus\_model.h5')  gray = cv2.imread('../img',cv2.IMREAD\_GRAYSCALE)  img = cv2.resize(gray, (64,64))  img = img.reshape((1,64,64,1))  pred\_y = model.predict(img)  if pred\_y[0][0]==max(pred\_y[0]):  print("{'emotion':'중립'}")  elif pred\_y[0][1]==max(pred\_y[0]):  print("{'emotion':'행복'}")  elif pred\_y[0][2]==max(pred\_y[0]):  print("{'emotion':'놀라움'}")  elif pred\_y[0][3]==max(pred\_y[0]):  print("{'emotion':'슬픔'}")  elif pred\_y[0][4]==max(pred\_y[0]):  print("{'emotion':'분노'}")  elif pred\_y[0][5]==max(pred\_y[0]):  print("{'emotion':'혐오'}")  elif pred\_y[0][6]==max(pred\_y[0]):  print("{'emotion':'두려움'}")  elif pred\_y[0][7]==max(pred\_y[0]):  print("{'emotion':'경멸'}")  #detector.crop\_faces\_dir(inputfile,outputfile)  #rect = detector.detect\_face(inputfile)  #detector.rect\_image(inputfile, rect, outputfile)  #detector.crop\_face(inputfile, rect, outputfile)    if \_\_name\_\_ == "\_\_main\_\_":  main() |

**fer2013plus\_model.ipynb**

CNN VGG - 13 알고리즘을 통하여 사진 속 얼굴의 감정을 분석하는 모델을 생성하여 fer2013plus\_model.h5 로 저장하였다.

|  |
| --- |
| import tensorflow as tf  gpus = tf.config.experimental.list\_physical\_devices('GPU')  if gpus:  try:  # Currently, memory growth needs to be the same across GPUs  for gpu in gpus:  tf.config.experimental.set\_memory\_growth(gpu, True)  logical\_gpus = tf.config.experimental.list\_logical\_devices('GPU')  print(len(gpus), "Physical GPUs,", len(logical\_gpus), "Logical GPUs")  except RuntimeError as e:  # Memory growth must be set before GPUs have been initialized  print(e)  import keras  import cv2  import numpy as np  import os  import math  # read all dataset for training  traindir = '/home/piai/Desktop/ferplus/FER2013Train/'  d\_ = {}  maxnum = 0  for i in range(8):  fname = '/home/piai/Desktop/ferplus/ferplus\_keras\_labe/train-ori-c%d.txt' % (i)  f = open(fname)  alllines = f.readlines()  f.close()  if maxnum < len(alllines):  maxnum = len(alllines)  d\_[i] = alllines  x\_train = np.zeros((maxnum\*8, 64, 64, 1), dtype='f')  x\_p = np.zeros((maxnum\*8, 64, 64, 1), dtype='f')  y\_p = np.zeros((maxnum\*8), dtype=np.uint8)  idxout = 0  for i in range(8):  data = d\_[i]  num = 0  idx = 0  while num < maxnum:  sp = (data[idx]).split(" ")  y\_p[idxout ] = int(sp[1])  img = cv2.imread(os.path.join(traindir, sp[0]), cv2.IMREAD\_GRAYSCALE)  img = cv2.resize(img, (64,64))  x\_p[idxout, :, :, 0] = img    idxout = idxout + 1  idx = idx + 1  if idx >= len(data):  idx = 0  num = num + 1  randomid = np.random.permutation(x\_p.shape[0])  y\_p = keras.utils.to\_categorical(y\_p, num\_classes=8)  y\_train = np.zeros(y\_p.shape, dtype=np.uint8)  for i in range(x\_p.shape[0]):  x\_train[i, :, :, :] = x\_p[randomid[i], :, :, :].copy()  y\_train[i] = y\_p[randomid[i]]  x\_p = 0  #Validation data를 불러온다.  valdir = '/home/piai/Desktop/ferplus/FER2013Valid/'  f = open('/home/piai/Desktop/ferplus/ferplus\_keras\_labe/validation.txt')  alllines = f.readlines()  f.close()  x\_val = np.zeros((len(alllines), 64, 64, 1), dtype='f')  y\_p = np.zeros((len(alllines)), dtype=np.uint8)  idx = 0  for line in alllines:  sp = line.split(" ")  y\_p[idx] = int(sp[1])  img = cv2.imread(os.path.join(valdir, sp[0]), cv2.IMREAD\_GRAYSCALE)  img = cv2.resize(img, (64,64))  x\_val[idx, :, :, 0] = img  idx = idx + 1  y\_val = keras.utils.to\_categorical(y\_p, num\_classes=8)  # Model defenition  from keras.preprocessing.image import ImageDataGenerator  ### data augmentation  datagen = ImageDataGenerator(featurewise\_center=True,  featurewise\_std\_normalization=True, \  width\_shift\_range=0.08, \  height\_shift\_range=0.08, \  zoom\_range=0.05, \  rotation\_range=20, \  shear\_range=0.05, \  horizontal\_flip=True)  ### model description  vgg13 = keras.models.Sequential()  vgg13.add(keras.layers.Conv2D(64, (3, 3), activation='relu',  input\_shape=(64,64,1), padding='same'))  vgg13.add(keras.layers.Conv2D(64, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.MaxPooling2D(pool\_size=(2, 2)))  vgg13.add(keras.layers.Dropout(0.25))  vgg13.add(keras.layers.Conv2D(128, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.Conv2D(128, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.MaxPooling2D(pool\_size=(2, 2)))  vgg13.add(keras.layers.Dropout(0.25))  vgg13.add(keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.MaxPooling2D(pool\_size=(2, 2)))  vgg13.add(keras.layers.Dropout(0.25))  vgg13.add(keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.Conv2D(256, (3, 3), activation='relu', padding='same'))  vgg13.add(keras.layers.MaxPooling2D(pool\_size=(2, 2)))  vgg13.add(keras.layers.Dropout(0.25))  vgg13.add(keras.layers.Flatten())  vgg13.add(keras.layers.Dense(1024, activation='relu'))  vgg13.add(keras.layers.Dropout(0.5))  vgg13.add(keras.layers.Dense(1024, activation='relu'))  vgg13.add(keras.layers.Dropout(0.5))  vgg13.add(keras.layers.Dense(8, activation='softmax'))  x\_train.shape  # Training Process  def steplr(epoch):  lr = 0.025  max\_epochs=100.0  lr = lr \* (1.0 - epoch/max\_epochs)  return lr  sgd = keras.optimizers.SGD(lr=0.025, decay=0.0005, momentum=0.9, nesterov=True)  vgg13.compile(loss='categorical\_crossentropy', optimizer=sgd,  metrics=['accuracy'])  datagen.fit(x\_train)  vgg13.fit\_generator(datagen.flow(x\_train, y\_train, batch\_size=128),  steps\_per\_epoch=x\_train.shape[0]/128,  epochs=100,  validation\_data=datagen.flow(x\_val, y\_val, batch\_size=128),  validation\_steps=x\_val.shape[0]/128,  callbacks=[  keras.callbacks.LearningRateScheduler(steplr, verbose=1),  keras.callbacks.ModelCheckpoint('vgg13-baseline.h5',  monitor='val\_acc',  verbose=1,  save\_best\_only=True)  ]  )  from keras.models import load\_model  vgg13.save('fer2013plus\_model.h5') |

o FER2013데이터에 라벨링을 한 FERPLUS데이터를 사용한다.

Raw Data - FER2013 and Label FERPLUS

48 X 48 gray scale images

28,709 training samples

3,589 validation data

3,589 test data

ㅇ 데이터셋

* Pubfig : 200명의 58,797개의 DATA SET이 존재한다.
* VGGFace : 10,000개 이상 데이터 확보 시 모델 생성가능하기에 사용하지 못하였다.
* VGGFace2 : 페이지가 소실되어 사용하지 못하였다.
* 공공 데이터셋 : 한국인 안면 이미지 AI에 무표정 찡그림, 행복 3가지 감정 이미지만 존재하여 사용하지 않았다.
* FER+ : ai 모델구현을 위해 FER2013데이터에서 감정의 라벨링을 수정한 FER+ 데이터를 사용하였다. fer2013과 fer+데이터와의 차이는 이미지 크기가 48\*48에서 64\*64로 이미지 크기조정과 이미지를 뒤집기 정도이며, 이만구천여개의 트레이닝셋과 약 4천개의 validation data와 test data로 나누었다.

o FER2013 원본데이터 : 48 X 48 그레이 스케일 이미지(48개), 교육 샘플 (28,709개), 검증 데이터(3,589개), 테스트 데이터(3,589개), 감정 레이블(0 = 화난, 1 = 혐오, 2 = 두려움, 3 = 행복, 4 = 슬픔, 5 = 놀람, 6 = 중립, 7개) , 라벨링의 인간 정확도 (60~65%)

o CLOVA Face Recognition (9가지 감정 라벨링)  
 화남(Anger), 혐오(Disgust), 공포(Fear), 웃음(Laugh), 무표정(Neutral), 슬픔(Sadness), 놀람

ㅇ 데이터 전처리

* FER +에는 8 가지 감정의 새로운 수정 된 레이블이 있다.  
  ( 0 : '중립', 1 : '행복', 2 : '놀람', 3 : '슬픔', 4 : '분노', 5 : '혐오', 6 : '두려움', 7 : '경멸')
* FER + 데이터 세트의 수정된 사항으로는 이미지 확대 부분(이미지의 자르기면 상자 48 X 48-> 64 X 64에서 이미지 크기 조정)과 경계 이미지 이동 이미지 뒤집기 이미지 회전 (각도) 이미지의 픽셀 정규화 등을 수행하여 fer2013 데이터와 ferPlus 데이터는 실제 데이터는 같지만 감정의 라벨링이 다르다.

텍스트이(가) 표시된 사진

자동 생성된 설명

[그림9] FER Plus 데이터 셋

VGGFACE 다운 및 구글 크롤링하여 다운 받고 Google Cloud Platform Vision API를 사용하여 얼굴 Crop을 진행하였다. 차후 데이터 라벨링을 하기 위하여 MicroSoft Azure API로 8가지 감정 라벨링을 시도했으나, 이미 얼굴부분만 잘린 데이터로는 감정분석이 불가능했다. 이는 전체적인 사진에서 Face Detection이 불가능한 것 같다고 예측된다. 이에 따라 방법을 모색 중에 FER2013 데이터를 찾았다. 기존 전기수에서도 이 데이터를 사용했으나, 차별화 한 부분은 다르게 적용한 부분은 사진데이터는 같으나 라벨링이 더욱 정확하게 업그레이드되었다.

데이터 전처리 과정을 위해 8개의 감정을 딕셔너리 형태로 생성하며, train, validation, test 데이터의 라벨과 이미지를 불러온다.

**pred\_emotion.ipynb**

생성한 모델(fer2013plus\_model.h5)에 감정 이미지를 불러와 예측해본다.

|  |
| --- |
| import tensorflow as tf  import matplotlib.pyplot as plt    gpus = tf.config.experimental.list\_physical\_devices('GPU')  if gpus:  try:  # Currently, memory growth needs to be the same across GPUs  for gpu in gpus:  tf.config.experimental.set\_memory\_growth(gpu, True)  logical\_gpus = tf.config.experimental.list\_logical\_devices('GPU')  print(len(gpus), "Physical GPUs,", len(logical\_gpus), "Logical GPUs")  except RuntimeError as e:  # Memory growth must be set before GPUs have been initialized  print(e)  import keras  import cv2  import numpy as np  import os  import math  from keras.models import load\_model  model = load\_model('fer2013plus\_model.h5')import cv2  original = cv2.imread('/home/piai/Desktop/asia\_crop\_data/validate/남자 배우 얼굴169.jpg', cv2.IMREAD\_COLOR)  gray=cv2.imread('/home/piai/Desktop/server\_crop\_picture/training/행복띠-checkpoint.jpg',cv2.IMREAD\_GRAYSCALE)  img = cv2.resize(gray, (64,64))  print(img.shape)  plt.imshow(img ,cmap='gray')  img = img.reshape((1,64,64,1))  img.shape  pred\_y = model.predict(img)  pred\_y  if pred\_y[0][0]==max(pred\_y[0]):  print("{'emotion':'중립'}")  elif pred\_y[0][1]==max(pred\_y[0]):  print("{'emotion':'행복'}")  elif pred\_y[0][2]==max(pred\_y[0]):  print("{'emotion':'놀라움'}")  elif pred\_y[0][3]==max(pred\_y[0]):  print("{'emotion':'슬픔'}")  elif pred\_y[0][4]==max(pred\_y[0]):  print("{'emotion':'분노'}")  elif pred\_y[0][5]==max(pred\_y[0]):  print("{'emotion':'혐오'}")  elif pred\_y[0][6]==max(pred\_y[0]):  print("{'emotion':'두려움'}")  elif pred\_y[0][7]==max(pred\_y[0]):  print("{'emotion':'경멸'}")  if pred\_y[0][0]==max(pred\_y[0]):  print('8')  elif pred\_y[0][1]==max(pred\_y[0]):  print('7')  elif pred\_y[0][2]==max(pred\_y[0]):  print('6')  elif pred\_y[0][3]==max(pred\_y[0]):  print('5')  elif pred\_y[0][4]==max(pred\_y[0]):  print('4')  elif pred\_y[0][5]==max(pred\_y[0]):  print('3')  elif pred\_y[0][6]==max(pred\_y[0]):  print('2')  elif pred\_y[0][7]==max(pred\_y[0]):  print('1')  .  max(pred\_y[0])  model.summary()  import pandas as pd  df = pd.read\_csv("/home/piai/Desktop/ferplus/fer2013new.csv")  df.head() |

**EmotionResultActivity.java**

**이미지의 감정이 분류가 되면 각 감정에 선호하는 음식을 추천해 주는 알고리즘을 구현하였다.**

|  |
| --- |
| package com.example.beamin.activity;  import android.content.Intent;  import android.support.v7.app.AppCompatActivity;  import android.os.Bundle;  import android.util.Log;  import android.view.View;  import android.view.Window;  import android.widget.ImageView;  import android.widget.LinearLayout;  import android.widget.TextView;  import android.widget.Toast;  import com.example.beamin.HttpConnection;  import com.example.beamin.R;  import org.json.JSONArray;  import org.json.JSONException;  import java.io.IOException;  import java.lang.reflect.Field;  import okhttp3.Call;  import okhttp3.Callback;  import okhttp3.Response;  public class EmotionResultActivity extends AppCompatActivity implements View.OnClickListener{  int key;  Class<R.drawable> drawable = R.drawable.class;  ImageView emotion\_result\_back\_view, emotion\_result\_rating\_aceept, emotion\_result\_image;  String category1, category2;  TextView rest\_name\_text;  LinearLayout resultlayout;  int restaurantNumber;  private HttpConnection httpConn = HttpConnection.getInstance();  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_emotion\_result);  Intent intent = getIntent();  key = intent.getExtras().getInt("key");  Log.d("@@@@@@@@@@", String.valueOf(key));  switch (key){  case 0:  category1 = "chiness";  category2 = "fastfood";  break;  case 1:  category1 = "japanese";  category2 = "pizza";  break;  case 2:  category1 = "fastfood";  category2 = "korean";  break;  case 3:  category1 = "snack";  category2 = "korean";  break;  case 4:  category1 = "snack";  category2 = "chicken";  break;  case 5:  category1 = "soup";  category2 = "korean";  break;  case 6:  category1 = "soup";  category2 = "chicken";  break;  case 7:  category1 = "jokbo";  category2 = "japanese";  break;  }  resultlayout = findViewById(R.id.emotion\_layout);  resultlayout.setOnClickListener(this);  emotion\_result\_image = findViewById(R.id.emotion\_result\_image);  rest\_name\_text = findViewById(R.id.emotion\_result\_restaurant\_name);  emotion\_result\_back\_view = findViewById(R.id.emotion\_result\_back\_tv);  emotion\_result\_back\_view.setOnClickListener(this);  emotion\_result\_rating\_aceept = findViewById(R.id.emotion\_result\_accept\_rating);  emotion\_result\_rating\_aceept.setOnClickListener(this);  try {  Log.d("쿼리문을 보자", category1 + category2);  httpConn.emotion\_result(category1, category2, callback);  } catch (JSONException e) {  e.printStackTrace();  }  }  @Override  public void onClick(View v) {  switch (v.getId()){  case R.id.emotion\_result\_back\_tv:  finish();  break;  case R.id.emotion\_result\_accept\_rating:  Toast.makeText(getApplicationContext(), "별점 저장 완료!", Toast.LENGTH\_LONG).show();  break;  case R.id.emotion\_layout:  Intent intent = new Intent(getApplicationContext(), MenuDetailActivity.class);  intent.putExtra("key",restaurantNumber);  startActivity(intent);  break;  }  }  private final Callback callback = new Callback() {  @Override  public void onFailure(Call call, IOException e) {  Log.d("Error", "콜백오류:"+e.getMessage());  }  @Override  public void onResponse(Call call, Response response) throws IOException {  try {  JSONArray arr = new JSONArray(response.body().string());  final String rest\_name = arr.getJSONObject(0).getString("restaurantName");  restaurantNumber = arr.getJSONObject(0).getInt("restaurantNumber");  Field field;  field = drawable.getField("a" + restaurantNumber);  final int imgurl = field.getInt(null);  runOnUiThread(new Runnable() {  @Override  public void run() {  rest\_name\_text.setText(rest\_name);  emotion\_result\_image.setImageResource(imgurl);  }  });  } catch (JSONException | NoSuchFieldException | IllegalAccessException e) {  e.printStackTrace();  }  }  };  } |

### **음식 랭킹 순위**

#### 구현 코드

**RankingActivity**

음식점 순위를 나타내는 주요 코드로 1위부터 5위까지의 음식점을 나타낸다.

|  |
| --- |
| 버튼을 누를 때 이벤트를 처리하기 위한 코드로 각 순위일 때, 실행할 문장을 나타낸다. 새로운 Intent를 만들고, RankingData 파일의 getRestnumber()를 통해 해당 순위에 해당하는 음식점 번호를 intent.putExtra()로 값을 전달해준다. 전달받은 음식점 번호를 key값으로 하여 getRestname()을 통해 각 순위에 해당하는 음식점을 나타낸다. |
| @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_ranking);  httpConn.ranking(callback);  ranking\_back\_button = findViewById(R.id.ranking\_back\_tv);  ranking\_back\_button.setOnClickListener(this);  ranking\_first = findViewById(R.id.ranking\_text\_first);  ranking\_first.setOnClickListener(this);  }  @Override  public void onClick(View v) {  switch (v.getId()){  case R.id.ranking\_back\_tv:  finish();  break;  case R.id.ranking\_text\_first:  Intent intent = new Intent(getApplicationContext(), MenuDetailActivity.class);  intent.putExtra("key",rankingData\_arr.get(0).getRestnumber());  startActivity(intent);  break;  }  }  @Override  public void onResponse(Call call, Response response) throws IOException {  try {  JSONArray arr = new JSONArray(response.body().string());  for(int i = 0; i < arr.length(); i++){  String name = arr.getJSONObject(i).getString("restaurantName");  key = arr.getJSONObject(i).getInt("restaurantNumber");  RankingData rankingData = new RankingData(name, key);  Log.d("@@@@@@@@@@@@", rankingData.getRestname());  Log.d("@@@@@@@@@@@@", String.valueOf(rankingData.getRestnumber()));  Log.d("@@@@@@@@@@@@", String.valueOf(i));  rankingData\_arr.add(rankingData);  }  } catch (JSONException e) {  e.printStackTrace();  }  runOnUiThread(new Runnable() {  @Override  public void run() {  ranking\_first.setText(rankingData\_arr.get(0).getRestname());  ranking\_second.setText(rankingData\_arr.get(1).getRestname());  }  });  } |

### **현 위치 기반 음식점 리스트 제공**

#### 기술 소개

카카오 지도 API를 이용하여, 사용자의 현재 위치를 지도에 나타낸다. 사용자의 현재 위치를 기반으로 음식점 리스트를 제공한다. 카카오 지도 API는 웹사이트와 모바일 애플리케이션에서 지도를 이용한 서비스를 제작할 수 있도록 다양한 기능을 제공하고 있다. 지도 API에는 네이버 지도 API, 카카오 지도 API, Google 지도 API 등이 있다.

[그림10] 카카오 지도

#### 구현 코드

**MapSearchActivity**

사용자의 현재 위치 정보를 검색한 주소 항목 정보를 MapResultActivity Class로 전달한다.

|  |
| --- |
| 이름과 주소, 경도, 위도의 주소 항목 정보를 MapResultActivity Class에 전달한다. |
| @Override  public void onItemClick(AdapterView<?> adapterView, View view, int position, long id) {  MapSearchData item = (MapSearchData) adapterView.getItemAtPosition(position);  String name = adapter.itemList.get(position).getName();  String address = adapter.itemList.get(position).getAddress();  double x = adapter.itemList.get(position).getX();  double y = adapter.itemList.get(position).getY();  Intent intent = new Intent(context, MapResultActivity.class);  intent.putExtra("name", name);  intent.putExtra("address", address);  intent.putExtra("x", x);  intent.putExtra("y", y);  startActivity(intent);  finish();  }  }); |
| 사용자 현재 위치의 주소, 위도, 경도를 전달한다. |
| @Override  public void onClick(View v) {  switch (v.getId()) {  case R.id.map\_search\_back\_tv:  finish();  break;  case R.id.map\_search\_finish\_tv:  adapter.itemList.clear();  httpConn.mapSearchList(searchEt.getText().toString(), callback);  adapter.notifyDataSetChanged();  listView.setAdapter(adapter);  break;  case R.id.map\_search\_present\_tv:  locationManager = (LocationManager) getSystemService(Context.LOCATION\_SERVICE);  Location userLocation = getMyLocation();  double latitude=0;  double longitude=0;  if(userLocation != null){  latitude = userLocation.getLatitude();  longitude = userLocation.getLongitude();  }  Geocoder geo = new Geocoder(this, Locale.getDefault());  List<Address> names = null;  try {  names = geo.getFromLocation(latitude,longitude,1);  } catch (IOException e) {  e.printStackTrace();  }  if(names.size()!=0) {  Intent intent = new Intent(context, MapResultActivity.class);  intent.putExtra("name", names.get(0).getAddressLine(0));  intent.putExtra("address", names.get(0).getAddressLine(0));  intent.putExtra("x", latitude);  intent.putExtra("y", longitude);  startActivity(intent);  finish();  }  break;  }  }  private Location getMyLocation(){  Location currentLocation = null;  if(ActivityCompat.checkSelfPermission(this,Manifest.permission.ACCESS\_FINE\_LOCATION) !=  PackageManager.PERMISSION\_GRANTED &&  ActivityCompat.checkSelfPermission(this,Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {ActivityCompat.requestPermissions(this,  new String[]{Manifest.permission.ACCESS\_FINE\_LOCATION},  this.REQUEST\_CODE\_LOCATION);  getMyLocation();  }  else{  String locationProvider = LocationManager.GPS\_PROVIDER;  currentLocation = locationManager.getLastKnownLocation(locationProvider);  if(currentLocation != null){  double lng = currentLocation.getLongitude();  double lat = currentLocation.getLatitude();  }  }  return currentLocation;  } |
| 카카오 rest API를 이용하여 사용자 위치 검색 목록을 가져온다. |
| @Override  public void onResponse(Call call, Response response) throws IOException {  try {  JSONObject jsonObject = new JSONObject(response.body().string());  Log.d("ggggggggggggggggggg", jsonObject.toString());  JSONArray arr = jsonObject.getJSONArray("documents");  for(int i=0;i<arr.length();i++){  String name = arr.getJSONObject(i).getString("place\_name");  double x = arr.getJSONObject(i).getDouble("y");  double y = arr.getJSONObject(i).getDouble("x");  String addr = arr.getJSONObject(i).getString("address\_name");  adapter.addList(name,addr,x,y);  activity.runOnUiThread(new Runnable() {  public void run() {  adapter.notifyDataSetChanged();  listView.setAdapter(adapter);  }  });  }  } catch (JSONException e) {  e.printStackTrace();  } |

**MapResultActivity**

MapResultActivity Class에 전달 받은 주소 항목 정보를 지도에 나타낸다.

|  |
| --- |
| MapSearchActivity Class에 전달 받은 이름과 주소, 경도, 위도의 주소 항목 정보를 생성한다. 주소 항목 정보를 나타낼 mapview를 설정하고, 경도와 위도를 기본값으로 초기화 한다. |
| protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_map\_result);  Intent intent = getIntent();  name = intent.getExtras().getString("name");  address = intent.getExtras().getString("address");  x = intent.getExtras().getDouble("x");  y = intent.getExtras().getDouble("y");  mapViewContainer = (ViewGroup) findViewById(R.id.map\_show);  detailEt = findViewById(R.id.map\_result\_detail\_addr\_et);  finishBt = findViewById(R.id.map\_result\_finish\_tv);  addrTv = findViewById(R.id.map\_result\_default\_addr\_tv);  backTv = findViewById(R.id.map\_result\_back\_tv);  marker = findViewById(R.id.marker\_result);  mapView = new MapView(this);  mapViewContainer.addView(mapView);  mapView.setMapCenterPointAndZoomLevel(MapPoint.mapPointWithGeoCoord(x, y), 2, true);  SharedPreferences pref = getSharedPreferences("pref", MODE\_PRIVATE);  SharedPreferences.Editor editor = pref.edit();  editor.putString("longitude", String.valueOf(y));  editor.putString("latitude", String.valueOf(x));  editor.commit();  mapView.setMapViewEventListener(this);  backTv.setOnClickListener(this);  finishBt.setOnClickListener(this);  marker.bringToFront();  } |
| 새로운 주소 형식을 생성하고, mapview를 이동할 때 주소를 설정한다. |
| @Override  public void onMapViewMoveFinished(MapView mapView, MapPoint mapPoint) {  Geocoder geo = new Geocoder(this, Locale.getDefault());  List<Address> names = null;  try {  names = geo.getFromLocation( mapView.getMapCenterPoint().getMapPointGeoCoord().latitude, mapView.getMapCenterPoint().getMapPointGeoCoord().longitude, 10);  } catch (IOException e) {  e.printStackTrace();  }  if (names.size() > 0) {  String tmp = names.get(0).getAddressLine(0);  if(tmp.contains("대한민국 ")){  tmp = tmp.replace("대한민국 ", "");  }  if(tmp.contains("서울특별시")){  tmp = tmp.replace("서울특별시 ", "");  }  SharedPreferences pref = getSharedPreferences("map", MODE\_PRIVATE);  SharedPreferences.Editor editor = pref.edit();  editor.putString("addressMain", tmp);  editor.commit();  addrTv.setText(tmp);  }  } |
| 전달 받은 주소 항목 정보를 이용해 음식점을 지도에 표시한다. |
| @Override  public void onClick(View v) {  if (v.getId() == R.id.map\_result\_back\_tv) {  finish();  } else if (v.getId() == R.id.map\_result\_finish\_tv) {  SharedPreferences pref = getSharedPreferences("map", MODE\_PRIVATE);  SharedPreferences.Editor editor = pref.edit();  editor.putString("addressMain", addrTv.getText().toString());  editor.putString("addressDetail", detailEt.getText().toString());  editor.putString("x", String.valueOf(x));  editor.putString("y", String.valueOf(y));  editor.commit();  finish();  }  } |

### **앱개발**

**SignupActivity**

|  |
| --- |
| #회원가입 창 구현  @Override  protected void onCreate(Bundle savedInstanceState) {  activity = this;  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_signup);  final String[] age\_group = {"-연령대 선택-", "10대", "20대", "30대", "40대", "50대"};  final String[] gender = {"-성별 선택-", "남성", "여성"};  backView = findViewById(R.id.signup\_back\_tv);  nickName = findViewById(R.id.signup\_nickname\_et);  id = findViewById(R.id.signup\_id\_et);  pw = findViewById(R.id.signup\_pw\_et);  signupBt = findViewById(R.id.signup\_finish\_bt);  spinner\_age\_group = (Spinner)findViewById(R.id.signup\_age\_group);  spinner\_gender = (Spinner)findViewById(R.id.signup\_gender);  ArrayAdapter adapter\_age\_group = new ArrayAdapter(this, android.R.layout.simple\_spinner\_item, age\_group);  spinner\_age\_group.setAdapter(adapter\_age\_group);  ArrayAdapter adapter\_gender = new ArrayAdapter(this, android.R.layout.simple\_spinner\_item, gender);  spinner\_gender.setAdapter(adapter\_gender);  backView.setOnClickListener(this);  signupBt.setOnClickListener(this);  }  @Override  public void onClick(View v) {  if(v.getId() == R.id.signup\_back\_tv){  finish();  }  else if(v.getId() == R.id.signup\_finish\_bt){  sendData();  //Synchronous action  }  }  private void sendData() {  new Thread() {  public void run() {  try {  httpConn.signUp(id.getText().toString(),pw.getText().toString(),nickName.getText().toString(), spinner\_age\_group.getSelectedItem().toString(), spinner\_gender.getSelectedItem().toString(), callback);  } catch (JSONException e) {  e.printStackTrace();  }  }  }.start();  //use thread for Asynchronous action  }  private final Callback callback = new Callback() {  @Override  public void onFailure(Call call, IOException e) {  Log.d("Error", "콜백오류:"+e.getMessage());  }  @Override  public void onResponse(Call call, Response response) throws IOException {  try {  final JSONObject jsonObject = new JSONObject(response.body().string());  if (jsonObject.getInt("code") == 200) {  SignupPhoneActivity.signupPhoneActivity.finish();  finish();  }  else{  new Handler(Looper.getMainLooper()).post(new Runnable() {  @Override  public void run() {  try {  tmp = jsonObject.getString("message");  } catch (JSONException e) {  e.printStackTrace();  }  Toast.makeText(getApplicationContext(), tmp, Toast.LENGTH\_SHORT).show();  }  });  //Use handler because callback function is Asynchronous function.  }  } catch (JSONException e) {  e.printStackTrace();  }  //JSON Passing  }  }; |

**LoginActivity**

|  |
| --- |
| # 고객 로그인 진행시 저장된 회원정보를 바탕으로 아이디, 패스워드 확인  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_login);  httpConn = HttpConnection.getInstance();  loginActivity = this;  id = findViewById(R.id.login\_id\_et);  pw = findViewById(R.id.login\_pw\_et);  signUpBt = findViewById(R.id.login\_signup\_tv);  loginBt = findViewById(R.id.login\_login\_bt);  loginBt.setOnClickListener(this);  signUpBt.setOnClickListener(this);  }  # 회원정보 일치시 로그인 진행  try {  JSONObject jsonObject = new JSONObject(response.body().string());  if (jsonObject.getInt("code") == 200) {  MainActivity.userNickname = jsonObject.getJSONObject("data").getString("userEmailId");  MainActivity.userID = jsonObject.getJSONObject("data").getString("userNickname");  MainActivity.loginSuccess = true;  finish();  }else{  Toast.makeText(getApplicationContext(), "회원정보가 다릅니다.", Toast.LENGTH\_LONG).show();  } |

**EmotionActivity / EmotionResultActivity**

|  |
| --- |
| # 감정분석을 받고싶은 사진을 갤러리에서 호출  private void getGallery() {  // File System.  final Intent galleryIntent = new Intent();  galleryIntent.setType("image/\*");  galleryIntent.setAction(Intent.ACTION\_PICK);  // Chooser of file system options.  final Intent chooserIntent = Intent.createChooser(galleryIntent, "사진을 선택하세요.");  startActivityForResult(chooserIntent, 1010);  }  # URI 정보를 이용한 사진 정보 호출  super.onActivityResult(requestCode, resultCode, data);  if (resultCode == RESULT\_OK && requestCode == 1010) {  if (data == null) {  Snackbar.make(findViewById(R.id.emotionView), "Unable to Pickup Image", Snackbar.LENGTH\_INDEFINITE).show();  return;  }  Uri selectedImageUri = data.getData();  imageView.setImageURI(selectedImageUri);  String[] filePathColumn = {MediaStore.Images.Media.DATA};  Cursor cursor = getContentResolver().query(selectedImageUri, filePathColumn, null, null, null); |
| # 감정 분석 결과에 따라 논문과 설문조사 근거를 바탕으로 한 음식점 추천 진행  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_emotion\_result);  Intent intent = getIntent();  key = intent.getExtras().getInt("key");  Log.d("@@@@@@@@@@", String.valueOf(key));  switch (key){ 8개의 감정과 카테고리 맵핑 }  resultlayout = findViewById(R.id.emotion\_layout);  resultlayout.setOnClickListener(this);  emotion\_result\_image = findViewById(R.id.emotion\_result\_image);  rest\_name\_text = findViewById(R.id.emotion\_result\_restaurant\_name);  emotion\_result\_back\_view = findViewById(R.id.emotion\_result\_back\_tv);  emotion\_result\_back\_view.setOnClickListener(this);  emotion\_result\_rating\_aceept = findViewById(R.id.emotion\_result\_accept\_rating);  emotion\_result\_rating\_aceept.setOnClickListener(this);  try {  Log.d("쿼리문을 보자", category1 + category2);  httpConn.emotion\_result(category1, category2, callback);  } catch (JSONException e) {  e.printStackTrace();  }  } |
| # 감정 분석 결과에 대한 사용자의 만족도를 확인하기 위한 소비자 만족도 수집  public void onClick(View v) {  switch (v.getId()){  case R.id.emotion\_result\_back\_tv:  finish();  break;  case R.id.emotion\_result\_accept\_rating:  Toast.makeText(getApplicationContext(), "별점 저장 완료!", Toast.LENGTH\_LONG).show();  break;  case R.id.emotion\_layout:  Intent intent = new Intent(getApplicationContext(), MenuDetailActivity.class);  intent.putExtra("key",restaurantNumber);  startActivity(intent);  break; |

**MenuListActivity / MenuDetailActivity**

|  |
| --- |
| #음식 카테고리별 음심적 불러오기  # 한식, 분식, 돈까스-회-일식, 치킨, 피자, 중국집, 패스트푸드, 족발-보쌈, 찜-탕 9개 카테고리 생성  @Override  public void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_menu\_list);  activity = this;  backView = findViewById(R.id.menu\_list\_back\_tv);  viewPager = (ViewPager) findViewById(R.id.menu\_list\_pager);  mTabLayout = (TabLayout) findViewById(R.id.menu\_list\_tab);  menuView = findViewById(R.id.menu\_list\_name\_tv);  mTabLayout.addTab(mTabLayout.newTab().setText("한식"));  mTabLayout.addTab(mTabLayout.newTab().setText("분식"));  mTabLayout.addTab(mTabLayout.newTab().setText("돈까스-회-일식"));  mTabLayout.addTab(mTabLayout.newTab().setText("치킨"));  mTabLayout.addTab(mTabLayout.newTab().setText("피자"));  mTabLayout.addTab(mTabLayout.newTab().setText("중국집"));  mTabLayout.addTab(mTabLayout.newTab().setText("패스트푸드"));  mTabLayout.addTab(mTabLayout.newTab().setText("족발-보쌈"));  mTabLayout.addTab(mTabLayout.newTab().setText("찜-탕"));  # 카테고리 클릭시 알맞은 음식점 제공  @Override  public void onPageSelected(int position) {  switch (position){  case 0:  menuView.setText("한식");  break;  case 1:  menuView.setText("분식");  break;  case 2:  menuView.setText("돈까스-회-일식");  break;  case 3:  menuView.setText("치킨");  break;  case 4:  menuView.setText("피자");  break;  case 5:  menuView.setText("중국집");  break;  case 6:  menuView.setText("패스트푸드");  break;  case 7:  menuView.setText("족발-보쌈");  break;  case 8:  menuView.setText("찜-탕");  break;  default:  break;  }}  @Override  public void onPageScrollStateChanged(int i) {  }  });  //Set viewpager change listener  Intent intent = getIntent();  tab = intent.getIntExtra("tab",0);  //read tab from mainactivity grid item to click  backView.setOnClickListener(this);  viewPager.setAdapter(menuAdapter);  new Handler().postDelayed(  new Runnable(){  @Override  public void run() {  mTabLayout.getTabAt(tab).select();  }  },0); |
| # 가게 클릭시 가게 메뉴 정보 얻기  @Override  public void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_menu\_detail);  detailActivity = this;  activity = this;  nameTv = findViewById(R.id.menu\_detail\_store\_tv);  minTv = findViewById(R.id.menu\_detail\_min\_price\_tv);  payHowTv = findViewById(R.id.menu\_detail\_pay\_tv);  backView = findViewById(R.id.menu\_detail\_back\_tv);  phoneBt = findViewById(R.id.activity\_menu\_detail\_phone\_bt);  mTabLayout = (TabLayout) findViewById(R.id.menu\_detail\_tab);  viewPager = (ViewPager) findViewById(R.id.menu\_detail\_pager);  mTabLayout.addTab(mTabLayout.newTab().setText("메뉴"));  mTabLayout.addTab(mTabLayout.newTab().setText("정보"));  //Add tablayout information  menuAdapter = new MenuDetailAdapter(getSupportFragmentManager(), mTabLayout.getTabCount());  viewPager.setAdapter(menuAdapter);  viewPager.addOnPageChangeListener(new TabLayout.TabLayoutOnPageChangeListener(mTabLayout));  //set view pager  Intent intent = getIntent();  key = intent.getExtras().getInt("key");  Log.d("@@@@@@@@@@@@@@@@@@", String.valueOf(key));  backView.setOnClickListener(this);  phoneBt.setOnClickListener(this);  } |

### **사용 프로그램**

#### Node.js

Node.js는 확장성 있는 네트워크 애플리케이션(특히 서버 사이드) 개발에 사용되는 소프트웨어 플랫폼이다. 작성 언어로 자바스크립트를 활용하며 Non-blocking I/O와 단일 스레드 이벤트 루프를 통한 높은 처리 성능을 가지고 있다.

내장 HTTP 서버 라이브러리를 포함하고 있어 웹 서버에서 아파치 등의 별도의 소프트웨어 없이 동작하는 것이 가능하며 이를 통해 웹 서버의 동작에 있어 더 많은 통제를 가능케 한다.

#### 안드로이드 스튜디오

안드로이드 스튜디오(영어: Android Studio)는 안드로이드 및 안드로이드 전용 어플(앱)제작을 위한 공식 통합 개발 환경 (IDE)이다.

#### SQL

관계형 데이터베이스 관리 시스템의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어이다. 관계형 데이터베이스 관리 시스템에서 자료의 검색과 관리, 데이터베이스 스키마 생성과 수정, 데이터베이스 객체 접근 조정 관리를 위해 고안되었다. 많은 수의 데이터베이스 관련 프로그램들이 SQL을 표준으로 채택하고 있다.

#### 플라스크(Flask)

파이썬으로 작성된 마이크로 웹 프레임워크의 하나로, Werkzeug 툴킷과 Jinja2 템플릿 엔진에 기반을 둔다. BSD 라이선스이다. 플라스크 프레임워크를 사용하는 애플리케이션에는 핀터레스트, 링크드인, 플라스크 자체를 위한 공동체 웹 페이지를 포함한다.

플라스크는 특별한 도구나 라이브러리가 필요 없기 때문에 마이크로 프레임워크라 부른다. 데이터베이스 추상화 계층, 양식 유효성 확인, 기타 기존의 서드파티 라이브러리가 공통 기능을 제공하는 구성 요소가 없다. 그러나 플라스크는 플라스 자체에서 구현된 것처럼 애플리케이션 기능을 추가할 수 있는 확장 기능을 지원한다. 확장 기능은 객체 관계 매퍼, 양식 유효성 확인, 업로드 관리, 다양한 개방형 인증 기술, 여러 공통 프레임워크 관련 도구들을 위해 존재한다. 확장 기능들은 코어 플라스크 프로그램에 비해 훨씬 더 정기적으로 업데이트된다.

#### Pytorch

PyTorch는 Python을 위한 오픈소스 머신 러닝 라이브러리이다. Torch를 기반으로 하며, 자연어 처리와 같은 애플리케이션을 위해 사용된다. GPU사용이 가능하기 때문에 속도가 상당히 빠르다.

파이토치는 파이썬 기반의 오픈 소스 머신러닝 라이브러리로, 페이스북 인공지능 연구집단에 의해 개발되었다. 간결하고 구현이 빨리되며, 텐서플로우보다 사용자가 익히기 훨씬 쉽다는 특징이 있다. 또한 코드를 직접 다룬다 사람들에게 설명해 주기에도 효과적이다.

텐서플로우와 Pytorch의 가장 큰 차이점은 딥러닝을 구현하는 패러다임이 다르다는 것이다. 텐서플로우는 Define-and-Run 프레임워크인 반면에, Pytorch는 Define-by-Run이다.

Define and Run는코드를 직접 돌리는 환경인 세션을 만들고, placeholder를 선언하고 이것으로 계산 그래프를 만들고(Define), 코드를 실행하는 시점에 데이터를 넣어 실행하는(Run) 방식. 이는 계산 그래프를 명확히 보여주면서 실행시점에 데이터만 바꿔줘도 되는 유연함을 장점으로 갖지만, 그 자체로 비직관적이다. 그래서 딥러닝 프레임워크 중 난이도가 가장 높은 편이다.

Pytorch는 언어 자체에 대한 어려움은 없다. 일반적인 파이썬 코딩과 비슷하기 때문이다. 선언과 동시에 데이터를 집어넣고 세션도 필요없이 돌리면 끝이다. 덕분에 코드가 간결하고 난이도가 낮은 편이다.사용자가 아직 적어 구글링으로 공부하기 힘든 환경에 있지만 이는 시간이 해결해 줄 문제이다.

이 패러다임의 차이로 텐서플로우의 경우 먼저 모델을 만들고 값을 다 따로 넣어주어야 해서 직관적이지 않지만, Pytorch의 경우 간단하고 직관적이다.

Tensorflow와 Pytorch는 둘다 계산 그래프를 정의하고 자동으로 그래디언트를 계산하는 기능이 있다. 하지만 Tensorflow의 계산 그래프는 정적이고 Pytorch는 동적이다. 즉 tensorflow에서는 계산 그래프를 한 번 정의하고 나면 그래프에 들어가는 입력 데이터만 다르게 할 수 있을 뿐 같은 그래프만을 실행할 수 있다. 하지만 PyTorch는 각 순전파마다 새로운 계산 그래프를 정의하여 이용한다.

# **결론**

## **기대효과**

* 기존의 배달 어플과는 다르게 감정을 반영하여 음식 메뉴 선정에 새로운 기준을 부여할 수 있음. 사용자의 현재 감정을 개선하는 음식을 제공 해준다는 점에서 사용자의 흥미를 끌 것으로 예상
* COVID-19라는 범유행 전염병으로 ‘비대면’이 일상화되면서 집에서 안전하게 먹을 수 있는 배달음식 수요 증가에 따라 사용자 유입 예상
* 배달 어플에 다양한 종류의 음식이 있지만 어떤 음식을 먹을지 결정하기 힘들거나, 음식을 먹음으로써 기분이 현재보다 더 나아지기를 원하는 사람들을 위한 시스템
* 실제로 사용자가 먹은 음식이 사용자의 기분에 큰 영향을 미치므로, 추천된 음식을 주문해 먹음으로써 기분이 개선되는 효과를 누릴 것으로 예상
* 감정인식 데이터에 따른 음식 만족도의 데이터 구축으로 인해 사용자의 감정에 따른 메뉴 추천의 정확도를 높여준다. 이것을 통하여 실제 구매전환율이 높은 사용자에게 업체를 노출하여 업체와 고객을 연결해 주는 역할을 할 것으로 예상

## **한계점 및 발전 방향**

#### 한계점

배달의 민족에서 API를 제공하지 않고, 가게별로 가격 및 정보들을 직접 어플 회사에 제공하는 방식을 이용하기 때문에 가게 정보 데이터베이스를 저장하는데 한계가 있었다. 이를 해결하기 위해 우리는 현재 포항공대 주변의 음식점 중 9개의 카테고리로 나누어 310개의 음식 데이터를 수집하였다.

개인 맞춤형 데이터를 제공하는 만큼 시간, 바이오 리듬 등 개인별 환경을 고려할만한 알고리즘을 추가하여 정확도 개선

#### 결제 시스템 관련 한계

배달의 민족 : 한주간 총 판매금액을 배달의 민족 측에서 3.3%를 떼고 난 금액을 다음주 수요일 가게에 입금해주는 시스템을 사용 (3.3%는 카드 결제 수수료보다 조금 더 높은 정도. 외부결제수수료)

요기요 : 1일~15일 보름간 총 판매금액을 요기요가 모았다가 배달중개수수료, 외부결제수수료 등등 약 13%를 떼고 5영업일 이내에 입금해주는 시스템을 사용함.

요기요는 바로결제시 따로 광고비를 받지 않지만, 배달의 민족은 수수료가 적은 대신 바로결제와 상관없이 어플에 등록하기 위한 광고비가 추가로 필요함.

위와 같이 배달어플에서 직접 결제하는 것은 어플회사에서 돈을 받아서 모든가게로 나눠주는 형태로 지금 상황에서 구현 불가능하다고 판단 -> 이에 따라 차선책으로 웹으로 만들지만 배달어플과 유사한 방식으로 추천한 음식들 실제 구매 횟수 및 추천 서비스 구현으로 변경

#### 발전 방향

음식 추천에 적용한 CBF 알고리즘을 이용하여 영화, 음악 등 사용자의 취향이 반영된 시장에 적용 가능할 것으로 예상된다.

Cnn vgg 13알고리즘을 통하여 감성 컴퓨팅분야에서의 사람과 컴퓨터의 상호작용 및 사람과 로봇의 상호작용을 할때에, 인간의 감성을 인식하고 이를 바탕으로 효과적인 상호작용 하는데 활용 할 수있다. 예를 들어 인간의 감정이 기쁨일때에는 기계 또한 기쁨의 말, 어조 등을 사용하며, 우울일때에는 위로, 응원등의 메시지를 담은 말, 어조등을 사용한다.

감성 컴퓨팅(affective computing) 분야 : 표정인식 및 인간 감성 인식을 바탕으로 한 적용의 대표적인 사례 중에 하나

# **조원 소개**

## **강민지**

**김한솔** : 고민이 생기면 민지에게 먼저 이야기할 수 있을 만큼 포항생활에서 큰 힘이 되었던 민지. 프로젝트를 하면서 배운 점도 많지만, 민지에게 배웠던 점도 정말 많은 것 같다.

**노선호** : 빅데이터 프로젝트에서 한 사람 이상의 몫을 해내며 큰 기여를 해준 민지누나. 발표까지 담당하며 보여준 멋진 모습을 보여주었습니다. 다음번에도 프로젝트를 함께하고 싶을만큼 믿음직스러운 모습을 보여주었습니다.

**성유기** : 민지의 긍정적인 말로 팀원들을 응원해주던 모습이 기억에 남습니다. 프로젝트를 수행할때에도 전체 흐름을 이해하여 팀의 방향성을 제시해주며 책임감 있게 자신의 역할을 다하는 팀원이었습니다. 포항에서 빅데이터 프로젝트를 함께하며 보여준 민지의 열정을 통해 저 또한 많은 것을 배울 수 있었습니다.

**양수환** : 적극적으로 프로젝트에 임하며 프로젝트의 성공에 많은 기여를 하였습니다. 프로젝트가 매끄럽게 진행될 수 있도록 노력하는 모습이 보기 좋았습니다.

**한길영** : 빅데이터 프로젝트부터 AI 프로젝트까지 중요한 역할을 담당하며 책임감 있는 모습을 보여주었습니다. 하나의 과제를 하더라도 꼼꼼하게 이해하고 스스로 해내려고 하는 모습이 너무 멋지다고 생각했습니다.

## **김한솔**

**강민지** : 시간이 요구되고 지루할 수 있는 작업들을 묵묵하게 진행해준 팀원입니다. 조 전체의 분위기를 유하게 해주고 맡은 일을 빠르게 처리해주었습니다.

**노선호** : 비전공자로서 걱정도 많고 따라오기 힘들었겠지만 포기하지 않고 따라와줘서 고마운 팀원입니다. 혼자서의 힘으로 해결할 수 없을 경우엔 물어봐 가면서 이해하려 하는 모습이 멋있었습니다.

**성유기** : 정이 많고 팀을 위해 희생하는 친구입니다. 추운 바깥 날씨에도 불구하고 팀원들을 위해 야식과 커피를 배달해 주는 한솔이의 모습이 기억에 남습니다. 팀의 분위기 메이커로 언제나 팀원들의 사기를 높여주는 친구입니다. 암기력과 업무처리 능력이 뛰어나 팀원들이 생각하지 못했던 내용들까지 상기시켜주어 팀의 프로젝트가 더욱 발전할 수 있었습니다.

**양수환** : 꼼꼼함이 장점인 팀원이었습니다. 파이썬 과정에서 힘들어하는 모습을 보였지만, 두 프로젝트를 진행하면서 가장 성장한 팀원이라고 생각합니다.

**한길영** : 탁월한 업무처리 능력을 소유하고 날카로운 피드백으로 프로젝트의 퀄리티를 향상시키는 데 큰 도움을 준 조원입니다. 프로젝트 결과물을 더 발전시킬 수 있는 방향의 피드백을 제공해 주고 처음 접해보는 내용에 대한 정보를 잘 검색해서 찾아주어 결과물을 한 층 더 업그레이드시켜주었습니다. 그리고 어떠한 역할을 맡더라도 믿을 수 있고 맡은 부분에 대해 책임감이 강해 어떤 과제도 거리낌 없이 맡길 수 있는 팀원입니다.

## **노선호**

**강민지** : 우리 조의 유일한 컴퓨터 공학 전공자 선호. 뛰어난 코딩 실력으로 프로젝트에 많은 기여를 해주었다. 평소에는 장난기도 많고 팀에서 활기찬 분위기를 담당하는 막내였는데 회의 때만큼은 누구보다 진지하고 카리스마 있는 모습을 보여주며 반전 매력을 보여준 우리의 조장!!

**김한솔** : 선호 덕분에 프로젝트를 잘 마무리할 수 있었다고 할 만큼 팀에서 중추적인 역할을 해준 조장입니다. 프로젝트 경험이 부족한 조원들에게 방향성을 제시해주고 함께 지내는 동안 보여준 솔선수범의 모습을 보여주었습니다. 앞으로 본인이 원하는 분야에서 인정받는 전문가가 되길!

**성유기** : 조장을 맡아 부담감이 느껴질 수 있는데도 항상 조원들을 편안하게 해주고, 분위기 메이커의 역할을 해주었습니다. 프로그래밍 실력이 뛰어나며, 프로젝트를 할 때 업무의 전체적인 구조를 파악하는 안목이 있어 앞으로의 미래가 기대되는 친구입니다. 앞으로 목표로 하는 곳에 꼭 취업했으면 좋겠습니다.

**양수환** : 우리 조에서 가장 막내지만 배울 점이 아주 많은 친구. 동생이지만 프로젝트 기간 동안 가장 의지를 많이 할 수 있었습니다. 프로젝트를 진행하면서 분명 힘든 점이 있었을텐데 팀원들에게 티내지 않고 항상 밝은 모습을 보여주었습니다.

**한길영** : 조를 이끌어나가기 위해 수면시간을 줄여가며 보이지 않는 곳에서도 항상 조원들을 생각해주었습니다. 조장으로서 책임감과 부담감을 가지고 있었을 텐데 티 내지 않고 묵묵하게 우리 조를 이끌어줘서 고맙다는 이야기를 전하고 싶습니다.

## **성유기**

**강민지** : 자신이 이해한 것을 누군가에게 설명해주는 것을 좋아하는 조원. 어떠한 프로젝트에서도 항상 끝까지 남아서 열심히 하고, 학구열이 가장 높았던 조원입니다.

**김한솔** : 프로젝트 내내 꾸준하고 우직하게 참여했습니다. 컴퓨터 관련 전공자가 아니지만 매일 따로 남아서 코딩 공부를 하던 유기에게 멋있다는 이야기를 해주고 싶습니다.

**노선호** : 열심히 수업을 듣고 모르는 부분이 나와도 포기하지 않고 끈기있는 모습이 기억에 남는 것 같습니다.

**양수환** : 팀원들의 의견을 항상 진지하게 들어주고 지지해줘서 힘이 되었습니다. 항상 팀에 도움이 될 방법을 생각하고 자신의 역할을 묵묵하게 해내는 모습이 본받을만하다고 생각하였습니다.

**한길영** : 맡은 일에 대해서 최선을 다하는 것은 물론 자신의 업무가 아닌 다른 분야에도 도움을 주려고 노력하였습니다.

## **양수환**

**강민지** : 매일 아침 7:30 누구보다 빠르게 등교하여 자기의 할 일을 하던 조원. 팀 회의 시간에도 팀에게 필요한 것이 무엇인지 생각하며 좋은 아이디어들을 이야기해주고, 피드백도 항상 꼼꼼하게 해주던 조원입니다.

**김한솔** : 팀에서 같은 문과로 의지를 많이 했던 친구. 이 친구 없었으면 프로젝트 기간을 버틸 수 없었을 것 같다고 생각할 정도로 정신적으로 의지를 많이 했다. 모르는 부분이 생기거나 어려움에 직면했을 때 가장 먼저 생각이 나던 친구. 앞으로도 친하게 지내고 싶다!

**노선호** : 우리 조에서 없었으면 큰일 날뻔한 인재! 살면서 본 사람 중 가장 인터넷을 잘 활용하고 필요한 정보를 찾아내는 능력이 탁월합니다. PPT, 영상제작, 발표 못하는 것 없이 팀에 정말 큰 기여를 한 형입니다.

**성유기** : 노는 것을 좋아하는 줄 알았는데 뒤에서 자기 할 일은 미리미리 하는 모습을 보여줬던 반전 매력의 친구입니다. 일 처리가 빠르고 아이디어도 좋아서 앞으로의 모습이 기대됩니다.

**한길영** : 반 사람들과도 잘 지내는 친화력 갑! 놀 때 놀고 공부할 때 공부하는 게 무엇인지 보여준 조원. 재능이 많아서 앞으로 뭘 해도 먹고살겠다는 생각이 들었습니다.

## **한길영**

**강민지** : 항상 조원들을 배려해주며 팀을 위해 궂은일도 마다치 않고 프로젝트에 참여해주었습니다. 피드백을 할 때도 상대방의 기분을 신경 쓰며 말을 하고 있다는 것이 항상 느껴질 정도로 배려심이 깊은 조원이라고 생각합니다.

**김한솔** : 정말 착한 오빠. 모르는 부분을 질문했을 때 자기가 모르는 내용이면 찾아서라도 알려주려고 했던 모습에 감동했던 기억이 난다. 장난치는 것도 다 받아주고 항상 편안하게 대해주셔서 감사합니다~

**노선호** : 조원들의 편의를 위해서 보이지 않는 곳에서 신경 써준 형. 팀 내에서 자신이 할 수 있는 부분을 찾기 위해 끊임없이 노력하고, 맡은 역할을 해내기 위해 끝없이 노력했던 팀원입니다.

**성유기** 묵묵히 본인이 맡은 역할 그 이상을 해내는 팀원입니다. 프로젝트에서도 좋은 결과물을 위해 항상 노력하고 완벽하게 해내려는 모습이 인상 깊었습니다. 프로젝트 기간동안 항상 긍정적인 마인드로 칭찬과 응원의 말을 해주며, 팀원들을 챙겨주는 모습이 참 고마웠습니다.

**양수환** : 동료와의 상호작용에 매우 능하며 주변 사람들을 편안하게 해주는 친구였습니다. 책임감을 가지고 항상 목적한 바를 훌륭히 수행하는 팀원으로 앞으로 좋은 일만 있었으면 좋겠습니다.

# **5.참고 문헌**

[1] 심희린, 심귀보. 『 딥러닝을 이용한 연속 시간 감정 상태 추론 시스템 설계. 한국지능시스템학회 논문지 』. 중앙대학교. 2019.

[2] 이수상. 『 KSLA 주요 발표자료 및 기획기사 - 이용자 추천 서비스 : 추천 알고리즘의 소개 』. 부산대학교. 2018.

[3] 이은영. 『지각된 감정이 음식 선호와 레스토랑 선택에 미치는 영향 』. 이화여자대학교 2008

[4] 윤호민, 최규완. 『사용자 선호기반 개인화 음식메뉴 추천 기법 연구』. 한국호텔외식관광경영학회. 2020.

[5] 조진관. 『빅데이터 환경에서의 음식 추천 서비스를 위한 사용자 텍스트 감성사전에 관한 연구』.한국지식정보기술학회. 2019.

[6] 이강현 외. 『감성분석 기반 음식추천 어플리케이션 설계』.한국정보과학회. 2012.

[7] <https://brunch.co.kr/@businessinsight/64>

[8] <https://noa-xyz.tistory.com/21>

[9] <https://brunch.co.kr/@wikitree/159>

[10] <https://bcho.tistory.com/1166>

[11] Lyman B.1989.A psychology offood ;more than a matteroftaste.aviBook. USA

[12]ChristensenL,BrooksA.2006.ChangingFoodPreferenceasaFunctionofMood.TheJournalofPsychology,140