음식 추천 프로그램 만들기

Team : 푸드알

1. 목표

재료가 한정된 상황에서 가지고 있는 재료들을 이미지 인식을 통하여 객체인식을 하고 객체인식을 하면서 나온 결과 값을 가지고 레시피나 요리를 추천하는 프로그램을 만드는 것이다.

(1) 재료들을 모아 데이터 셋을 만들고 요리에 필요한 재료의 데이터 셋을 학습시켜서 카메라 인식을 통한 결과값에 대응하도록 만들어서 프로그램을 완성할 것이다.

(2) 카메라인식을 통하여 이미지 객체인식을 하여 코드로 도출한다. ex) ‘11010과 같은 숫자형’, ‘계란, 김치’와 같은 문자형 결과값을 나오게 할 것이다.

(3) 카메라 인식을 통해 만들어진 결과값을 가지고 train 데이터 셋과 연결시켜서 음식 추천 시스템을 만들어 낼 것이다.

1. 프로젝트의 중요성 및 창의성

이미 비슷한 기술들이 구글링 해보면 많다. 다양한 예시 중 외국음식을 중심으로한 프로젝트가 많았다. 우리가 참고한 코드는 여러 나라 음식을 어떠한 국가의 음식인지를 학습시키는 코드였다. 우리는 이 코드를 이용하여 한국에서 먹는 음식을 위주로 다뤄서 보다 우리에게 맞는 학습모델을 만들 수 있을 것 같다.

직접 데이터셋을 만들어보고 학습 시킨 뒤 트레인 했던 코드와 객체인식 코드를 합쳐서 코드 분석에 대한 이해도를 높이고 이전보다 실력을 향상시킬 것이다.

1. 관련 기술

<https://github.com/madhavambati/Cuisine-classification-using-Ingredients>

레시피 재료를 사용하여 Sci-kit Learn을 사용하여 python에서 요리를 분류합니다.

+레시피 재료를 test.json 파일로 각 국가에 해당하는 재료들을 train.json파일로 만들어 둔뒤에 test.json에 있는 재료들이 train.json에 해당하는 것이 많을 경우 각 국가로 매칭되어서 어떤 국가의 음식인지 알 수 있다.

<https://pysource.com/2019/06/27/yolo-object-detection-using-opencv-with-python/>

yolo를 이용하여 객체 인식을 하여 인식한 객체에 대한 값을 받을 수 있게끔 하여서 그 값과 위 데이터 셋을 이용하여 레시피 추천 시스템을 만들 수 있도록 할 것이다.

1. 기존에 쓰이던 기술.

* CNN 모델을 이용하여 식재료 모델링을 하였고 Flask, Android, Retrofit2, PyTorch를 이용하여 식재료를 촬영하면 재료를 인식하여 레시피를 추천해주는 앱을 만듬.

<https://github.com/yjun1806/find_receipe>

* KONLPY와 MECAB를 이용하여 자연어 처리 작업을 하였고, Python의 keras를 하여 레시피 추천 챗봇을 만듬.

<https://github.com/jinhalim/recipe_bot>

1. 프로젝트 기술

Test.josn 파일 형식

[

{

"id": "Kimchi fried rice",

"ingredients": [

"Kimchi",

"green onion",

"cooking oil",

"dark soy sauce",

"sugar",

"rice",

"seaweed falkes",

"sesame salt"

]

},

{

"id": "Tuna fried rice",

"ingredients": [

"tuna",

"egg",

"rice",

"green onion",

"cooking oil",

"soy sauce",

"oyster sauce",

"pepper"

]

},

{

"id": "Egg fried rice",

"ingredients": [

"rice",

"green onion",

"egg",

"cooking oil",

"dark soy sauce",

"seasoned salt"

]

},

{

"id": "Bacon Cream Pasta",

"ingredients": [

"bacon",

"onion",

"mushroom",

"flour",

"milk",

"butter",

"olive oil",

"parmesan cheese",

"salt",

"pasta noodles"

]

}

]

Train 파일 형식

[

{

    "id": 6703,

    "cuisine": "spanish",

    "ingredients": [

      "worcestershire sauce",

      "chili sauce",

      "olive oil",

      "chopped celery",

      "red bell pepper",

      "oysters",

      "paprika",

      "carrots",

      "ground red pepper",

      "all-purpose flour",

      "evaporated skim milk"

    ]

  },

  {

    "id": 12312,

    "cuisine": "mexican",

    "ingredients": [

      "lime juice",

      "purple onion",

      "ground cumin",

      "lettuce",

      "boneless skinless chicken breasts",

      "chopped cilantro fresh",

      "knorr garlic minicub",

      "hellmann' or best food real mayonnais",

      "avocado",

      "whole wheat tortillas",

      "mango"

    ]

  },

  {

    "id": 15129,

    "cuisine": "cajun\_creole",

    "ingredients": [

      "top sirloin steak",

      "corn starch",

      "garlic powder",

      "green pepper",

      "dried thyme",

      "salt",

      "canola oil",

      "pepper",

      "stewed tomatoes",

      "onions"

    ]

  },

]

학습 코드

import pandas as pd                   #for data handling (espically with data\_frames)

from sklearn import preprocessing     #for labeling cuisines (apply numerical labels or ids on cuisines )

from sklearn.feature\_extraction.text import TfidfVectorizer #to convert text data to numericals

                                                             #without losing any property or parameter

from sklearn.model\_selection import train\_test\_split  #for spilitting data set into test data set and train data set

from sklearn.metrics import accuracy\_score, confusion\_matrix  #for confusion matrix and accuracy score

from sklearn.svm import SVC

from sklearn import svm #our training algorithm

from sklearn.model\_selection import GridSearchCV # optional(for hyper parameter tuning)

import os

print(os.listdir("폴더 경로"))

print("done")

#retrieving data from .json files

df\_train = pd.read\_json("train.json 경로")

df\_test = pd.read\_json("test.json 경로")

df\_train

df\_train['ingredients'] = df\_train['ingredients'].apply(','.join) # this converts ingredients columns into arrays

df\_test['ingredients'] = df\_test['ingredients'].apply(','.join)   # for test data, above one is for train data

X\_train = df\_train['ingredients']  # assigning a new variable for ingredients array(for train data)

X\_test = df\_test['ingredients']    # assigning a new variable for ingredients array(for test data)

print("done")

print(X\_train)

#all our parameters(i.e cuisine,ingredients) are in text format

#most of the machine learning algorithms cannot handle text data forms

#so we have to convert the text data into some numerical data without losing its quality

encoder = preprocessing.LabelEncoder() #encoder for our y-values i.e cuisine

y\_train\_transformed = encoder.fit\_transform(df\_train['cuisine'])#this labels each y\_value(which is in text) to a number

#as our X-values i.e ingredients are grouped as array for each cuisine

#we need to vectorize or simply assign a set of numerical value to each element in array without losing its quality.

#one of the effective tool for this is TfidVectorizer

vec = TfidfVectorizer(binary = True).fit(X\_train.values) #assign our vectoriser

X\_train\_transformed = vec.transform(X\_train.values) #applying vectorizer for x-values of train set

X\_test\_transformed = vec.transform(X\_test.values)   #applying vectorizer for x-values of test set

print("done")

#spiliting test-data set into further test-set and train-set

#In my view This spiliting process is for calculating accuracy\_score

X\_for\_train, X\_for\_test, y\_for\_train, y\_for\_test = train\_test\_split(X\_train\_transformed, y\_train\_transformed ,test\_size= 0.25, random\_state = 0)

#best set algorithm for current problem

clf = svm.LinearSVC(C=0.5, max\_iter=100, random\_state=20, tol=0.5) #these are hyper parameters best suitable for this problem

                                                                #I will post the code at the end for hyper parameter tuning

clf.fit(X\_for\_train, y\_for\_train) #training our train-set

y\_pred = clf.predict(X\_for\_test)  #predicting our test-set

print("done")

#accuracy calculation

accuracy = accuracy\_score(y\_for\_test, y\_pred)

print('accuracy\_score = ', accuracy)

#predicting values for test set

y\_pred = clf.predict(X\_test\_transformed)

y\_pred\_transformed = encoder.inverse\_transform(y\_pred) #result will be encoded by label encoder, so it should be

                                                    #decoded to view or pass into dataframe

predictions = pd.DataFrame({'id': df\_test['id'], 'cuisine': y\_pred\_transformed}) #constructing a data frame with ids

                                                                                   #and predictions as columns

predictions.to\_csv('submit.csv', index = False)

print('done')

#for hyper parameter tuning

#using grid search

random\_state = []

for i in range(1, 110, 10):

    random\_state.append(i)

param\_grid = {'max\_iter': [10, 100, 1000],

               'random\_state': random\_state, 'tol':[0.01, 0.1, 0.5 ], 'C':[0.5, 1, 1.5]}

# all the numbers in param\_grid are for an optimal parameter tuning

# tested with many possibilities and provided the best amongest them

#note: these params vary from problem to problem. These are some of best suitable for this problem

optimal\_clf = svm.LinearSVC()

param = optimal\_clf.get\_params().keys()

grid\_search = GridSearchCV(optimal\_clf, param\_grid)

grid\_search.fit(X\_for\_train, y\_for\_train)

print(grid\_search.best\_params\_)

better\_model = grid\_search.best\_estimator\_

better\_pred = better\_model.predict(X\_for\_test)

better\_accuracy = accuracy\_score(y\_for\_test, better\_pred)

print(better\_accuracy)

나오는 결과 파일 형식

submit.csv 파일

Yolo 예시 코드 (이용할 방법 찾는 중)

import cv2

import numpy as np

# 웹캠 신호 받기

VideoSignal = cv2.VideoCapture(0)

# YOLO 가중치 파일과 CFG 파일 로드

YOLO\_net = cv2.dnn.readNet("yolov2-tiny.weights","yolov2-tiny.cfg")

# YOLO NETWORK 재구성

classes = []

with open("yolo.names", "r") as f:

    classes = [line.strip() for line in f.readlines()]

layer\_names = YOLO\_net.getLayerNames()

output\_layers = [layer\_names[i[0] - 1] for i in YOLO\_net.getUnconnectedOutLayers()]

while True:

    # 웹캠 프레임

    ret, frame = VideoSignal.read()

    h, w, c = frame.shape

    # YOLO 입력

    blob = cv2.dnn.blobFromImage(frame, 0.00392, (416, 416), (0, 0, 0),

    True, crop=False)

    YOLO\_net.setInput(blob)

    outs = YOLO\_net.forward(output\_layers)

    class\_ids = []

    confidences = []

    boxes = []

    for out in outs:

        for detection in out:

            scores = detection[5:]

            class\_id = np.argmax(scores)

            confidence = scores[class\_id]

            if confidence > 0.5:

                # Object detected

                center\_x = int(detection[0] \* w)

                center\_y = int(detection[1] \* h)

                dw = int(detection[2] \* w)

                dh = int(detection[3] \* h)

                # Rectangle coordinate

                x = int(center\_x - dw / 2)

                y = int(center\_y - dh / 2)

                boxes.append([x, y, dw, dh])

                confidences.append(float(confidence))

                class\_ids.append(class\_id)

    indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.45, 0.4)

    for i in range(len(boxes)):

        if i in indexes:

            x, y, w, h = boxes[i]

            label = str(classes[class\_ids[i]])

            score = confidences[i]

            # 경계상자와 클래스 정보 이미지에 입력

            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 5)

            cv2.putText(frame, label, (x, y - 20), cv2.FONT\_ITALIC, 0.5,

            (255, 255, 255), 1)

    cv2.imshow("YOLOv3", frame)

    if cv2.waitKey(100) > 0:

        break

참고 :

학습

<https://github.com/madhavambati/Cuisine-classification-using-Ingredients>

yolo

<https://deep-eye.tistory.com/6>

<https://pysource.com/2019/06/27/yolo-object-detection-using-opencv-with-python/>

스위프트 영상인식

<https://developer.apple.com/documentation/vision/recognizing_objects_in_live_capture?language=objc>

레시피 인식 앱

<https://github.com/yjun1806/find_receipe>