Food Nutrition Estimation

Jiaqi Wang

Contents

1. Introductions

The main goal of the project is to estimte the nutrients consumed by the user, using a captured images as input. We use the glasses with the camera to capture the image of the certain food, at the same time, several sensors for recording arm movements while eating can also be helpful for the estimation. The sensors can get the data of the movements and store the data into the database. To estimate the nutrients, the project should contain the function of the food category classification, volume estimation, and some other related functions. The equipment should also be convenient for the user to wear.

2. Composition of the Systems

2.1Sensors Reading

Now we use two sensors to collect data. One is piezo and another one is accel. We use Arduino to collect data and use the Bluetooth to transmit the data to the raspberry pi. The port of the piezo is COM3 and the port of the accel is COM5. If we choose the wrong port number, the program can not run on the laptop. The board is RFduino. We can set the configuration in the Arduino. we can detect two sensors after we open the Bluetooth of the raspberry pi zero. The devices' name are piezo and accel. After we compile and run the ble.py the raspberry can get the data and display them on the monitor.

2.2Google Vision Kit

The Google Vision Kit is the core of the system. And the deployment of the machine learning and the control of the whole system should be utilized on the board. The Google Vision Kit has a tensorflow accelerated raspberry pi zero board. However, due to the fact that the memory of the raspberry pi zero is limited, we can only run the small model such as SueezeNet and MobileNet on the board. According to the tutorials on the official website, the model which can be run on the board with the parameters are as followed:

MODEL TYPE	SUPPORTED CONFIGURATION
MobileNetV1	input size: 160x160, depth multiplier = 0.5 input size: 192x192, depth multiplier = 1.0
MobileNetV1 + SSD	input size: 256x256, depth multiplier = 0.125
SqueezeNet	input size: 160x160, depth multiplier = 0.75

Figure :Allowed Models on the Google Vision Kit

We also have trained several models with a bigger input size than the official recommendation, but they can not be deployed on the Google Vision Kit.

2.3Glasses

The glasses is mainly used to take the picture of the certain food. Because the glasses can work for little time without electricity, we use the portable power to provide energy. We connect the raspberry pi board with the glasses with wireless network. After the board is connected with the glasses, we can get the camera of the glasses on http://192.168.10.1/media/?action=snapshot. The user name is admin and there is no password. We also use the button on the Google Vision Kit to control the glasses. If we push the button, the camera will take the picture and the file name is the time when the picture is taken.

2.4Database

We use the SQLite3 database to store the data collected from the sensors. We create three tables to store the certain data. The tables are as followed:

Table Sensor

Sensor_Id	Name	Comments

Table Events

Id	Time	Value	Sensor_Id	Series_Id

Table Series_Events

Series_Id	StartTime	EndTime

2.5Deployment

The Deployment can be divided into three parts:

a. Get the final result: After we import the model which we use to make food classification, the final result can be gotten after we run the image_classification.py file. The file provides the function to get the food classification result.

b. Import the model: We import the model utilizing the file foodtype.py, and the path of this file is “/opt/aiy/projects-python/src/aiy/vision/models”. The model which we import is a binaryproto file, which can be gotten through the official compiler in the Ubuntu system. The official recommendation version of the Ubuntu is 14.04. More details and the official compiler can be found on the website: https://aiyprojects.withgoogle.com/vision/#makers-guide

c. The label: The class contains the name of all the food in the datasets. Once we get the final result of the food classification, the program can find the exact name of the food type according to the label.

3.Future Work

3.1.The problem is that the ble.py can not always work well. Sometimes after collecting some data from sensor and interrupt the program, we connect make the bluetooth work well next time.

3.2.Until now we still do not make a complete demo to combine the sensors and other parts of the system together. In that case, some data in the database such as Series_Id may not change during different experiment.