CS 725: Foundations of Machine Learning
Calorie Estimation of Food item from Images

CV Mavericks*
23m2154, 23m2156, 23m2157, 23m2158, 23m2162

November 28, 2023

Abstract

Our project addresses the limitations of current food image datasets in accurately estimating calories. We introduce a food image dataset that includes both food annotations and detailed volume and mass records for each item, along with a calibration reference. Our dataset comprises 2978 images, each accompanied by annotations and essential food metrics. To estimate the calorie content of food in this dataset, we employ a deep learning approach utilizing YOLO for food detection and a GrabCut algorithm to outline the contours of each food item. With this information, we calculate the volume and subsequently, the calorie content of each food.

1 Problem Statement

Estimating the calorie content from food images requires the food image datasets which faces a critical challenge in accurately estimating calorie content due to the choice of object detection algorithm and volume estimation method. This project aims to fill this gap by using a detailed food image dataset provided by ECUST[2]. It includes precise food annotations, along with volume, mass records, and a reliable calibration reference (one Yuan coin) for each item. This dataset is a collection of 2978 images, each meticulously annotated and supplemented with necessary food metrics i.e. volume, density, calorie content etc. To achieve accurate calorie estimation, a deep learning methodology is employed. This involves the adaptation of You Only Look Once (YOLO) for precise food detection, followed by the implementation of the GrabCut algorithm to precisely outline the contours of each food item. The segmented images would be utilized to calculate the volume and, subsequently, the calorie content of each food item.

The project aims to address two key challenges in dietary assessment from food images:

Detection and Classification of Food Items: Accurate identification and categorization of food items from images represent the initial hurdle. Existing methodologies often fall short in providing precise detection and classification. This project seeks to leverage advanced techniques, particularly YOLO, to enhance the capability to not only detect but also classify food items within images. By employing YOLO, the system aims to achieve an improvement in the accuracy and efficiency of food item recognition.

*The project has undergone revisions based on the feedback provided on 18/10/2023. Initially, the project focused on *Image-Based Disease Diagnosis using Deep Learning*. The new project concept has received approval from Krishnakant Bhatt (21q050016@iitb.ac.in).

Estimation of Food Volume for Calorie Content Determination: The second significant challenge lies in estimating the volume of identified food items, a crucial step in determining their calorie content. Traditional approaches struggle to provide the necessary precision for this task. To overcome this limitation, the project proposes the integration of GrabCut, an advanced segmentation algorithm. By applying GrabCut, the system aims to accurately delineate the contours of each food item, enabling precise volume estimation. This, in turn, facilitates a more accurate and reliable assessment of calorie content.

By combining YOLO for food item detection and classification with GrabCut for precise volume estimation, this project seeks to accurately estimate the calorie content from food images. The successful execution of this approach promises to benefit individuals seeking improved dietary choices, weight management, fitness goals, and specific dietary requirements.

2 Proposed Solution Approach

The solution methodology for estimating the calorie content from food images can be summarized as follows:

Object Detection: Our aim is to help people who want to keep track of the calories they consume. We would create a system that works with a smartphone. Before eating, users need to take pictures of their food from the top and side views. They should include a One Yuan coin in each picture. For the top view, we use YOLO to recognize the types of food and draw an outline or bounding box around them in the photos. We do the same for the side view. We reframe object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities. Using our system, you only look once (YOLO) at an image to predict what objects are present and where they are. YOLO is refreshingly simple. A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance. This unified model has several benefits over traditional methods of object detection.

Image Segmentation: Before figuring out the volume, we first break down each box into its separate parts. We would use something called GrabCut algorithm. It's a way to process images that helps us focus on specific areas. Normally, users would have to highlight areas themselves, but we've made it so the system can do this on its own. After this step, we get a precise outline or contour of each food item. Then, we can estimate the volume and calorie content. This process uses an image processing approach to segment each bounding box. As mentioned above, the bounding boxes around the object that GrabCut needs can be provided by YOLO. After segmentation, we can get a series of food images stored in matrix, but with the the values of the background pixels being replaced by zeros. This will leave only the foreground pixels.

Calorie Estimation: The user needs to provide the image clicked from both side and top view along with the One Yuan coin. According to the One Yuan coin detected in the top view, the true size of a pixel is known. Similarly, we know actual size of of a pixel in the side view. Then we use different formulas to estimate volume of each food. After getting volume, food's calorie is obtained by searching related calorie tables.

3 Datasets

We utilize the ECUST dataset, which encompasses 19 diverse food types including apple, banana, bread, bun, doughnut, egg, fired doughnut, grape, lemon, litchi, mango, mooncake, orange, peach, pear, plum, kiwi, sachima, and tomato as shown in Figure 1. The model is trained on a total of 2978 images from this dataset. Each food item is captured in multiple sets of photos, showcasing both its top and side views (see Figure 2). Each image is calibrated using a one Yuan coin as shown in Figure 3, and no more than two food items are present in a single picture. Additionally, all images in the dataset are smaller than 1000 by 1000 pixels. The dataset provides density and energy content information for all 19 food items.

To ensure accurate measurement of food volume and mass, foods that are sufficiently large, stable, and less likely to deform. are selected. Foods with small volumes, like peanuts, are challenging to measure accurately and can lead to significant errors when compared to their actual volume. In ECUST dataset, every food item is presented in its entirety. In the dataset, the whole foods, such as entire apples rather than sliced ones were prioritized to facilitate volume and weight measurements. Photos were taken in various mobile devices, under lighting conditions, including low light settings. Different shooting angles were used for top and side views. The position of the food in images was flexible, as long as it was fully captured. A One Yuan coin was chosen as the calibration object due to its common availability, with a diameter of 25.0mm.

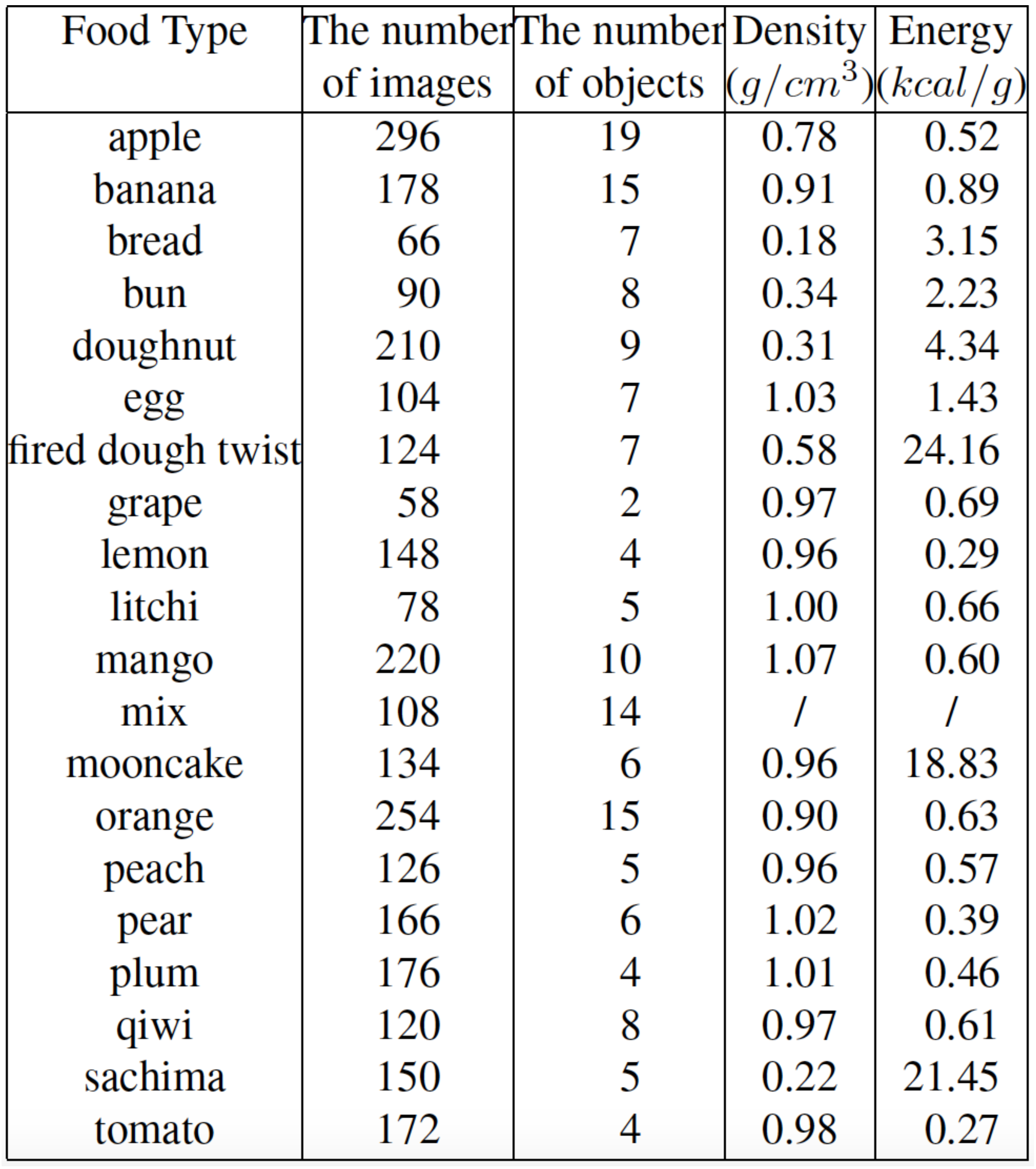
4 Implementation Details

4.1 YOLO

This algorithm[3] takes an image as input and produces a convolution feature map. We unify the separate components of object detection into a single neural network. Our network uses features from the entire image to predict each bounding box. It also predicts all bounding boxes across all classes for an image simultaneously. This means our network reasons globally about the full image and all the objects in the image. The YOLO design enables end-to-end training and realtime speeds while maintaining high average precision. Our system divides the input image into an S × S grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts. Formally we define confidence as $Pr(Object) \times IOU_{pred}^{truth}$. If no object exists in that cell, the confidence scores should be zero. Otherwise we want the confidence score to equal the intersection over union (IOU) between the predicted box and the ground truth. Each bounding box consists of 5 predictions: x, y, w, h, and confidence. The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell. The width and height are predicted relative to the whole image. Finally the confidence prediction represents the IOU between the predicted box and any ground truth box. Each grid cell also predicts C conditional class probabilities, $Pr(Class_i|Object)$. These probabilities are conditioned on the grid cell containing an object. We only predict one set of class probabilities per grid cell, regardless of the number of boxes B. At test time we multiply the conditional class probabilities and the individual box confidence predictions,

$$Pr(Class_i|Object) \times Pr(Object) \times IOU_{pred}^{truth} = Pr(Class_i) \times IOU_{pred}^{truth} \tag{1}$$

which gives us class-specific confidence scores for each box. These scores encode both the



Food Type	The number of images	The number of objects	Density (g/cm^3)	Energy $(kcal/g)$
apple	296	19	0.78	0.52
banana	178	15	0.91	0.89
bread	66	7	0.18	3.15
bun	90	8	0.34	2.23
doughnut	210	9	0.31	4.34
egg	104	7	1.03	1.43
fired dough twist	124	7	0.58	24.16
grape	58	2	0.97	0.69
lemon	148	4	0.96	0.29
litchi	78	5	1.00	0.66
mango	220	10	1.07	0.60
mix	108	14	/	/
mooncake	134	6	0.96	18.83
orange	254	15	0.90	0.63
peach	126	5	0.96	0.57
pear	166	6	1.02	0.39
plum	176	4	1.01	0.46
qiwi	120	8	0.97	0.61
sachima	150	5	0.22	21.45
tomato	172	4	0.98	0.27

Figure 1: Details of food item from ECUST dataset

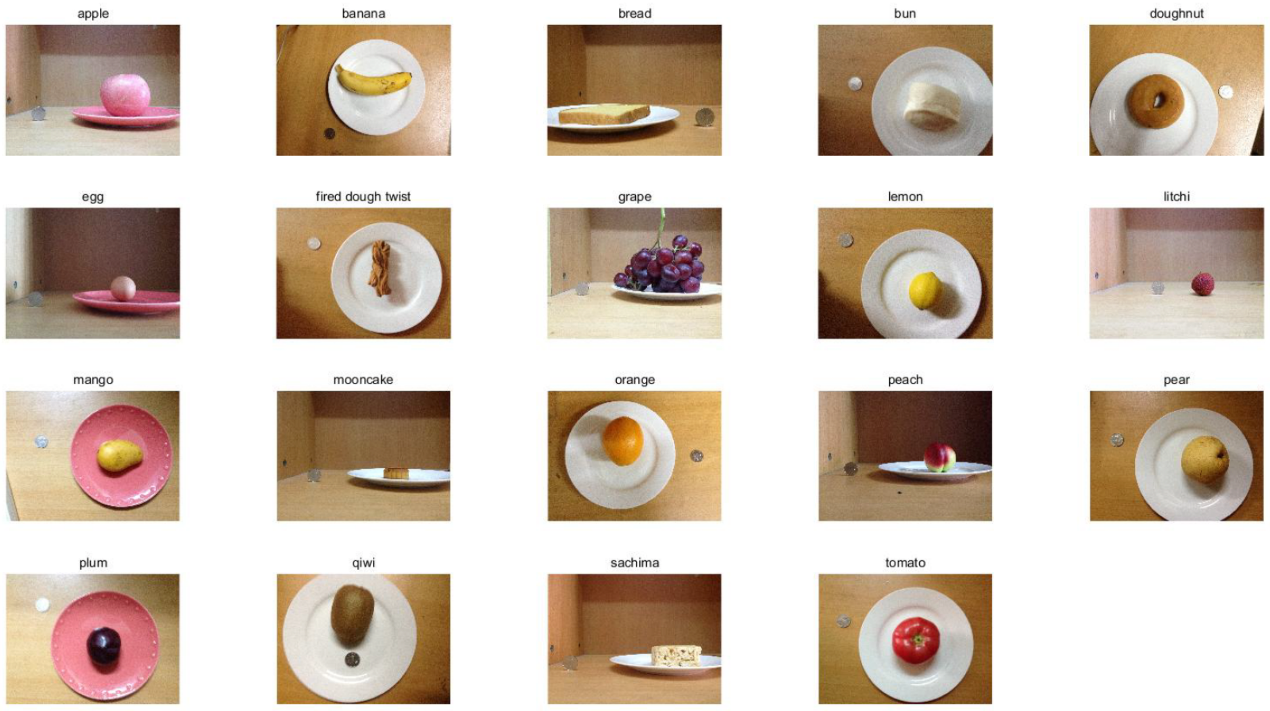


Figure 2: Sample food items from ECUST dataset

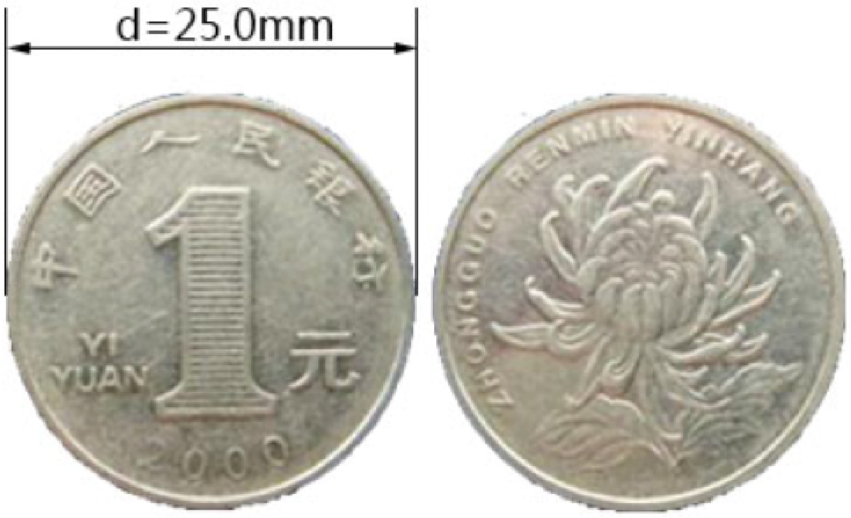


Figure 3: Dimension of One Yuan coin

probability of that class appearing in the box and how well the predicted box fits the object. For evaluating YOLO on PASCAL VOC, we use S = 7, B = 2. PASCAL VOC has 20 labelled classes so C = 20. Our final prediction is a 7 × 7 × 30 tensor

4.2 GrabCut Algorithm

The GrabCut algorithm[1] employs Image Segmentation. In this method, we use a box to define the object, effectively separating it from the background. The outer region of the box represents the background, while the inner portion combines some background and the object. Subsequently, an iterative process designates each pixel as either background or foreground. Initially, the computer makes an educated guess based on user-provided data. Then, we utilize a Gaussian mixture model to characterize both the foreground and background. Next, we label pixels as probable foreground or probable background, designating others as unknown.

In this algorithm, every pixel becomes a node in a graph. Additional nodes like source and sink nodes are introduced. The source node connects to the foreground pixels, while the sink node connects to the background pixels. The likelihood of a pixel being foreground or background influences the edge weights connecting the pixels to these nodes. The similarity between pixels, indicated by the edge information, determines these weights. If there's a substantial color difference between pixels, a low weight is assigned to the edge. Following this, the graph undergoes Segmentation using min-cut. This operation divides the graph into two parts, separating the source and sink nodes, and does so with the least possible cost. The total weights of the cut edges constitute the cost function. Post-cut, all pixels connected to the Source node become foreground, while those linked to the Sink node become background. This process continues until the classification stabilizes.

4.3 Volume Calculation

To estimate the volume[4], we calculate the scale factors based on calibration objects. We use a One Yuan coin to show the specific process of calculating the volume. The diameter of the coin is 2.5 cm, and the side view's and top view's scale factor (α_S and α_T) is calculated with Equation 2 and 3.

$$\alpha_S = \frac{2.5}{(W_S + H_S)/2} \tag{2}$$

$$\alpha_T = \frac{2.5}{(W_T + H_T)/2} \tag{3}$$

Where W_S and H_S are the width and height of bounding box in side view. Moreover, W_T and H_T are the width and height of bounding box in top view.

Furthermore, we divide the foods into three categories based on shape: ellipsoid, column, irregular. Different volume estimation formula will be selected for different types of food, according to Equation 4.

$$v = \begin{cases} \beta \cdot \frac{\pi}{4} \cdot \alpha_S^3 \cdot \sum_{k=1}^{H_S} (L_S^k)^2, & \text{if shape is ellipsoid} \\ \beta \cdot s_T \cdot \alpha_T^2 \cdot H_S \cdot \alpha_S, & \text{if shape is column} \\ \beta \cdot s_T \cdot \alpha_T^2 \cdot \alpha_S \cdot \sum_{k=1}^{H_S} \left(\frac{L_S^k}{L_S^{max}} \right)^2, & \text{if shape is irregular} \end{cases} \tag{4}$$

Where L_S^k is the number of foreground pixels in side view of row k ($k \in 1, 2, 3, \ldots, H_S$). $L_S^{max} = \max_k L^k$, it records the maximum number of foreground pixels in side view. β is a compensation factor (default value = 1.0). s_T is the surface area of the top view.

After estimating the volume, the next step is to estimate each food's mass (m in g). It can be calculated in Equation 5, Where v represents the volume of food in cm^3, and ρ represents its density value in g/cm^3. Then the calorie (C in cal) of the food can be calculated by Equation 6 using the unit energy (c in cal/g).

$$m = \rho \cdot v \tag{5}$$

$$C = c \cdot m \tag{6}$$

The density, unit energy values and shape for each food items can be obtained from ECUST dataset.

5 Results

The main results are as follows:

- Volume estimation results are shown in the below figure. For most types of food in our experiment, the estimation volume are closer to reference volume.
- We use mean error as the evaluation metric for the volume estimation of each of the food item class. It is assumed that within a single food class, the estimated volume is either overestimated or underestimated for all the instances.

$$ME = \sum_{k=1}^{N} \frac{v^k_{true} - v^k_{pred}}{v^k_{true}} \times 100\% \tag{7}$$

- The mean error between estimation volume and true volume does not exceed 20% except banana, grape, bread. For some food types such as lemon, orange, tomato, our estimation result is close enough to the true value.

References

[1] Yubing Li et al. "Grab Cut Image Segmentation Based on Image Region". In: (2018), pp. 311–315. DOI: 10.1109/ICIVC.2018.8492818.

[2] Yanchao Liang and Jianhua Li. "Computer vision-based food calorie estimation: dataset, method, and experiment". In: *arXiv preprint arXiv:1705.07632* (2017).

[3] Joseph Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: (2016), pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[4] V.Subapriya. "Prediction of Health Risks Using Semi-Supervised Learning". In: *International Journal of Scientific Research and Engineering Development* (2020).

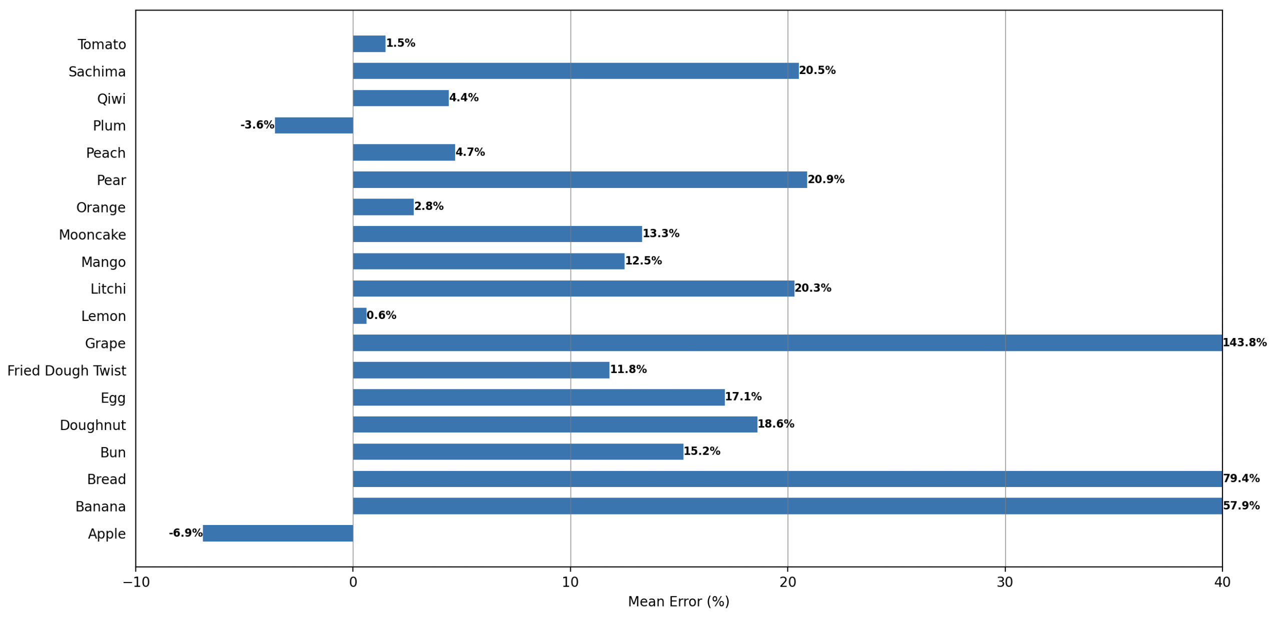


Figure 4: Volume estimation results of food items from images

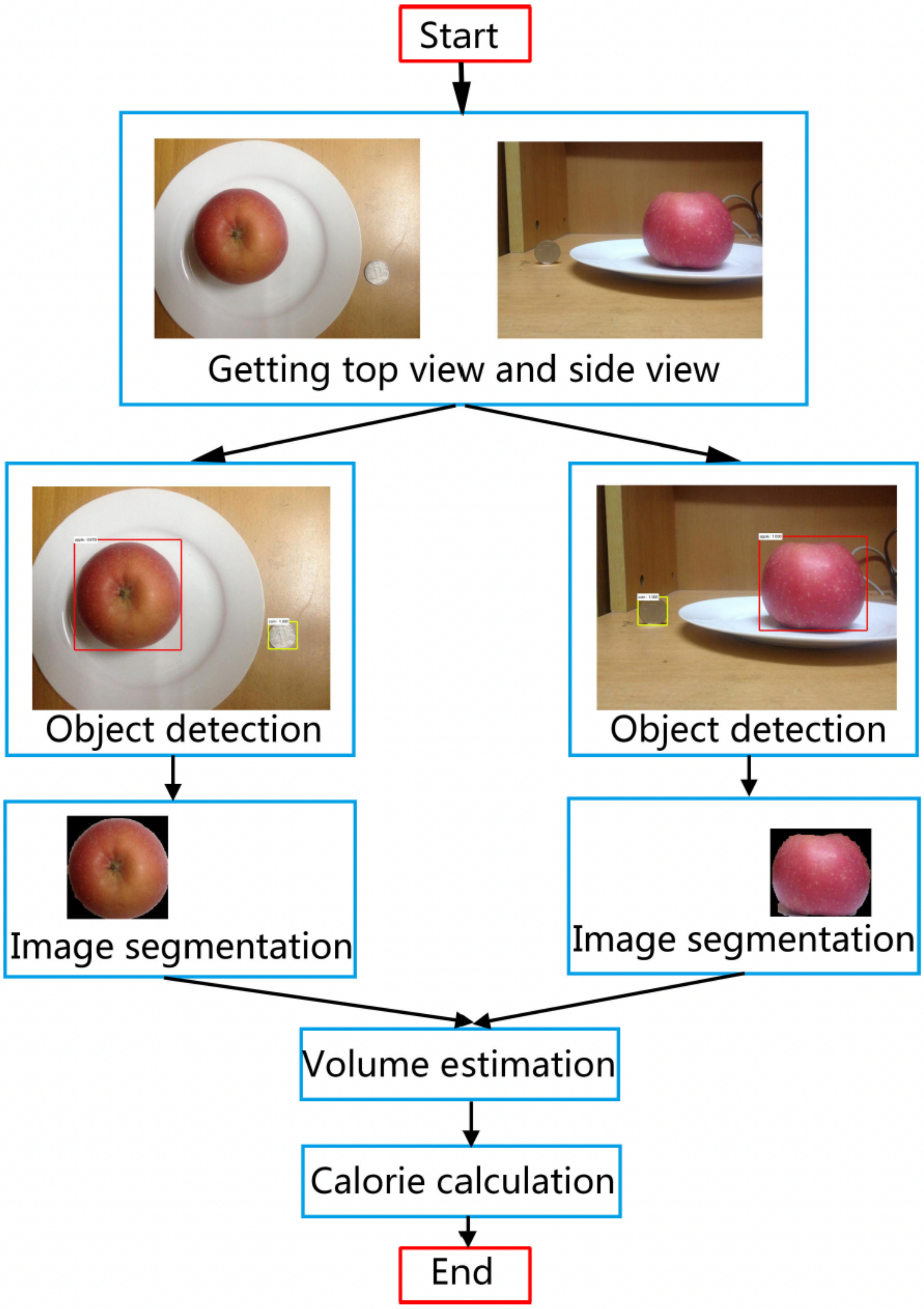


Figure 5: Flowchart to implement the project[2]