Food recognition and leftover estimation

Using Computer vision and image processing techniques

MohammadAmin HajiBagher Tehran

Navid Pourhadi Hasanabad

Alessandro Bernava

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DEGLI STUDI DI PADOVA

Computer Vision (2023)

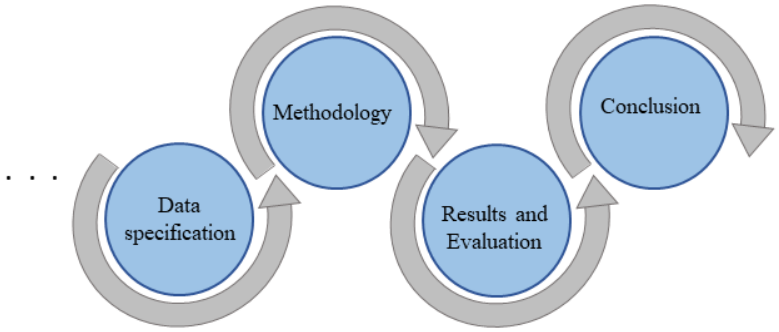
I Introduction

The goal of this project is to develop a computer vision system that can effectively address the issue of food waste in the university canteens. The system aims to tackle the negative impacts caused by food wastage. To achieve this, the system will focus on monitoring consumers' habits and encouraging a culture of respect for food.

The proposed system will employ image recognition and analysis techniques to scan a canteen consumer's food tray at the end of a meal. It will work by comparing two images: one taken before the meal begins and one captured at the end of the meal. By analyzing these images, the system will be able to estimate the amount of leftover food for each type of item present on the tray.

There are different techniques which could be used in order to localize, recognize, and segment the objects in the images. CNNs are the powerful tools to be used for all of the tasks mentioned, but the problem for utilizing these approaches is the **huge amount of dataset** it needs. Due to the lack of dataset, **image processing techniques** are helpful to perform these tasks in this project.

The rest of the report is organized as depicted in Fig. 1. Section II discusses the specification of the trays in the dataset. The proposed system is described in Sec. III. The details of the results in each step of implementation are provided in Sec. IV. performance evaluation results appear in Sec. V followed by concluding remarks in Sec. VI.



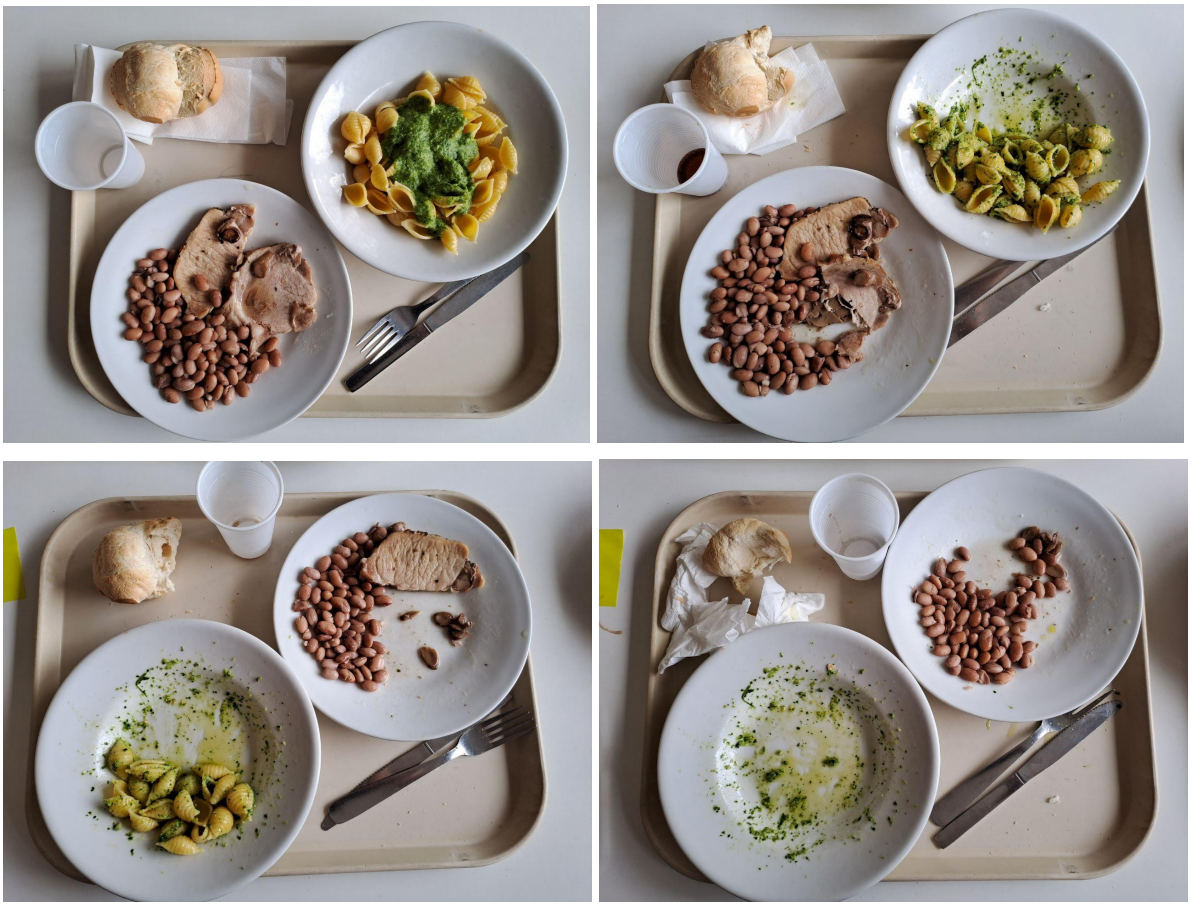
II Dataset specification

The provided dataset for this project consists of 8 series of the images of food in trays “before” starting to be eaten, and the same tray “after” the meal is eaten in 3 steps. The targeted objects in the trays consist of :

1) “Piatto primo” which contains risotto or a type of pasta,
2) “Piatto secondo” which contains a combination of two types of food,
3) It may contain bread,
4) It may contain a bowl of salad.

Also, there might be some objects inside the tray which are not intended in the system and have to be rejected as non-targeted objects (e.g. spoon, fork, dessert ticket, …).

In the images of the same tray, the organization of plates and objects might be changed or not. Furthermore in some cases the distance of the camera to the tray or the viewing angle are different, so the image has a wider region of background.



III Methodology

Proposed System Architecture

The architecture of our proposed system comprises 4 modules.

1) Pre-processing: applying laplacian filter and extracting some regions as the possible objects in the input image based on intensity slicing of RGB channels,
2) Food Recognition: using 3 Bag of Words algorithm with a combination of SIFT and DAISY features to recognize the foods based on a proposed procedure.
3) Food segmentation: using intensity slicing method similar to pre-processing module based on the differences of general color specification of the foods and the surrounding regions (plates).
4) Comparison: applying the previous modules on “before” and “after” eating images of trays and making the comparison based on the amount of pixels of each category.

We discuss each step of our implementation in detail separately as follows:

Pre-Processing module

In this module, we tried to have an approximate estimation of the possible object coordinates and reject the regions of the image which are estimated as background (non-objects). The pre-processing module consists 3 steps as follows:

1) We apply laplacian filter in order to sharpen the image which can help for better detection of the objects using lines and edges. As the images are colored and have 3 channels of RGB, we first split the channels and apply laplacian filter on each channel separately. then , we merge the acquired result of the laplacian filter on each channel, and subtract it from the input image to obtain the **SharpenImage.**
2) As the shape of the plates in the trays are mostly circular, we implemented a function named **"extract_plates()"** and used cv::HoughCircle() function in it to detect circles with the radius in the range of [150, 800].
    - The evaluation of this function is fairly good to detect the regions of plates and bowl of salad, except in the cases where the camera has a viewing angle which makes the plates do not look like circles, or some parts of the plates are not captured in the image. In these cases, we implement another function to extract objects based on the color filtering in step three.
3) In the third step of the pre-processing, we implemented a function named **"extract_objects()"** to filter out some regions which are considered as background. In this function, we used the remaining parts of the image (after using extract_plate() function) and applied a color filtering. Based on the analysis on color characteristics of the trays and the objects inside it, we comprehend that the pixels of the region that may contain an object has at least one channel (R, G, B) with a very different value in comparison to the other channels. So, we create a mask based on the distance of pixel channels. For each pixel we compute the distance between the values of three channels. Then, if the difference between these three distances were higher than 35, we consider this pixel as a pixel of the possible object. Otherwise, we label it as background.

    Subsequently, we apply some dilation and erosion to remove small distinct detected regions from the mask and expand the connected regions.

    Finally, by finding contours using cv::findcontour() function, we extract connected regions in the mask and add them to the possible object image.

The results of this part is shown as below (example: tray1/food_image.jpg):

Segmentation module

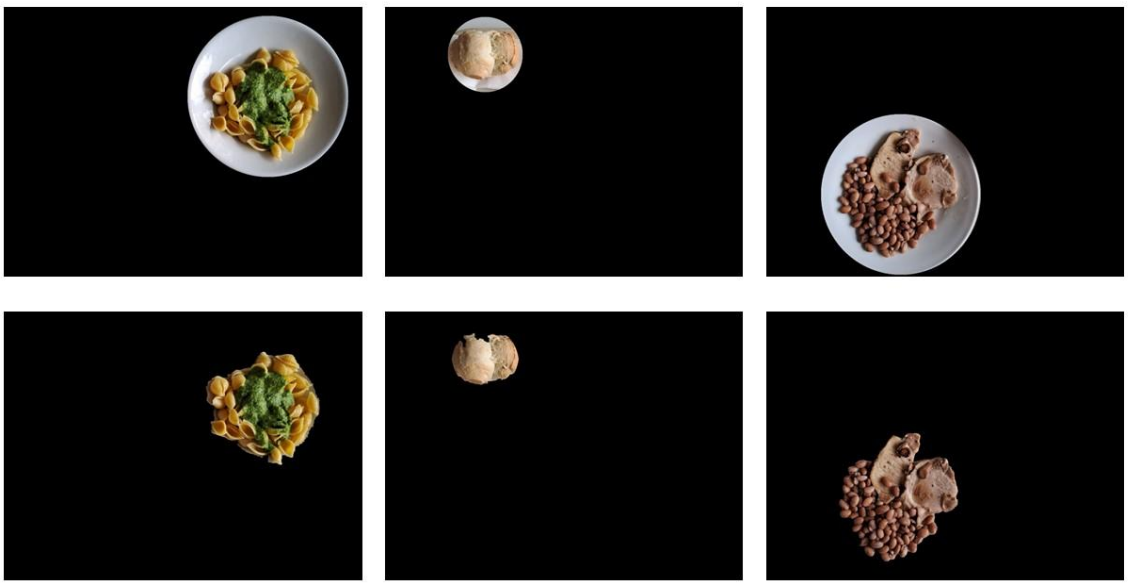
In this module, our main focus is on the Regions of Interest (ROIs) obtained in the previous module. Specifically, for segmenting foods inside the plates, we isolate each plate, which is represented as a circle. To achieve this, we apply a color-based segmentation technique, using a color slicing method similar to the preprocessing module. However, this time, the approach is tailored to the characteristics of the objects.

We set a threshold of 25 for the color difference in RGB values for each pixel. Next, we find and fill the contours of the resulting segmentation, creating a binary mask that represents the food regions.

We then employ a circular mask with the same dimensions as the original circle. In this way artifacts like the border of the plate can be removed from the mask.

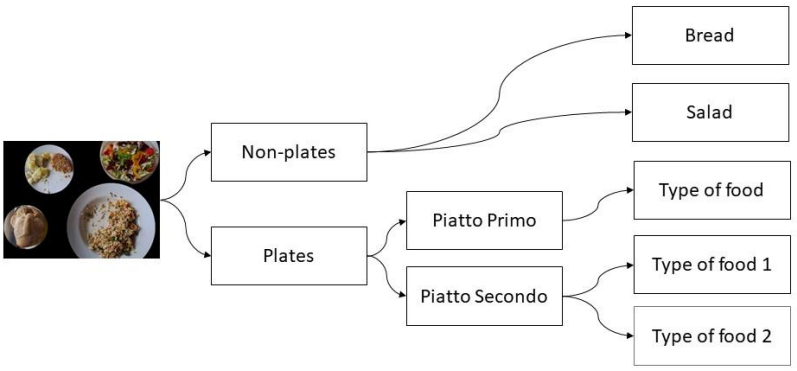
To address the issue of part of plates that overlap with the food contours in the mask, we introduce a color mask. We sample the color of the plate and search for colors in the pixels of the original image that correspond to the food region defined by the mask. If these colors are proportional within a specific threshold, and at the same time the pixel is black in the mask related to the initial segmentation (color based) we remove (make black) the corresponding pixels from the mask.

By following this approach, we can isolate and segment the food regions with sufficient accuracy, with the drawback that some pixels related from the food in the original image are removed from the mask due to the color mask approach, which is in any case necessary to remove part of plates inside foods contour.



Food Recognition module

In this module, we tried to recognize the regions of interest selected in the previous module based on the following procedure:



For this reason, we implemented three Bags of Words algorithms using a combination of SIFT and DAISY features (in which we detect the keyPoints using SIFT, then use DAISY to recognize the descriptor of each keyPoint) of the predefined objects (extracted objects of different categories using bounding boxes presented in the dataset). The reason for using the DAISY feature descriptor is because of its float type that performs better in BoW algorithms which use Hamming distance. Also, for training the BoW algorithm, we used k-means clustering to generate the codebook.

Each BoW component is used for the following targets:

1) BoW1: for recognizing between Bread and Salad,
2) BoW2: for recognizing between categories of food in piatto primo (pastas and risotto),,
3) BoW3: for recognizing between categories of food in piatto secondo.

The output of each BoW component would be a list of distances between the considered segment to each word of that BoW.

Regarding the mentioned procedure for recognition, we implemented a function named **findPlatesSuperTypes()** which perform sequentially for each segment to recognize its final category as below:

1) Detecting plate or non-plate: we apply all of the BoWs on the selected segment. Each BoW returns a list of distances for the corresponding categories. Then, we give an **average** on the **most probable categories (less than a threshold)** for each BoW and compare their results. If the distance for each BoW2 and BoW3 is less than BoW1, then we consider this segment as plate, otherwise we consider it as non-plate.
2) Detecting Piatto Primo or Secondo: if the detected segment was plates, by comparing the average distances of most possible categories in BoW2 and BoW3 we can comprehend which one is Piatto Primo and Piatto secondo.
3) Detecting the final category: based on the SuperTypes recognized in the previous stages, we can now use the best distance of selected BoW for the segment to return the final category of that segment (id of food).

Also, for differentiating between the two types of foods in the Piatto secondo, we split the segment of this plate and make a list of patches, then we utilize the BOW3 on each patch to recognize the two types of foods.

Comparison module

In this module, by applying the previous modules on the “before” and “after” images of the trays, we only count the pixels in each segment and compare them based on their assigned categories. So, for example, if in the food_image.jpg (before meal) we have a bread segment with 1000 pixels and in the leftover1.jpg the bread segment has 500 pixels, based on the following formula, the leftover estimation would be 0.5, which means that half of the bread remains as leftover.

$$R_i = \frac{\#pixels\ for\ food\ i\ in\ the\ "after"\ image}{\#pixels\ for\ food\ i\ in\ the\ "before"\ image}$$

IV Results and Evaluation

In this section of the report, we illustrate the final result of our work and the Evaluation on our proposed method in comparison to the bounding boxes and masks presented in the benchmark dataset.

V Conclusion

Regarding the evaluation and results of our proposed method, it seems that our method works fairly well in the segmentation and recognition of the foods inside the plates, but it needs to be combined with other techniques to perform with high accuracy on the objects outside of the plates.