Food Recognition & Leftover Estimation

Onur Alp Guvercin 20722549

Suleyman Erim 2072058

Stefano Deriu 2078244

Abstract

This paper presents a computer vision system for object detection, segmentation, and evaluation. The system starts by reading input files, specifically a folder containing the trays to be analyzed. The code automatically scans the folder, reads the images, mask images, and bounding box files, providing feedback on the code's functionality. Various detection techniques such as template matching, template matching with scale and rotation invariance, SIFT feature-based matching and segmentation techniques such as Thresholding, Region Growing, Watershed, and Grabcut were explored, with Grabcut being selected as the most suitable algorithm for accurate segmentation. We only provide template matching and Grabcut algorithms with the code. The Grabcut algorithm effectively differentiated between foreground and background objects, even in the presence of color similarities. The system's segmentation performance was evaluated using ground truth annotations, aligning the segmentation approach with the reference annotations and fine-tuning the algorithm for optimal alignment. The evaluation included metrics such as mean average precision (mAP) for detection, mean intersection over union (mIoU) for segmentation, and food leftover estimation. The results demonstrated the effectiveness of the traditional computer vision techniques in detecting, segmenting, and evaluating objects.

1 Introduction

The main code starts with simple reading of input files. Our choice has been to give as input only path of the folder containing the dataset provided by the drive folder and the tray we want to analyze; an example of command line could be:

CV_SUMMER.exe C:\Users\student\Desktop\CV_SUMMER tray1

The code automatically scans the folder, which must be constructed in a similar way to the one provided by the drive's one.

It automatically reads all the images in the tray, all the mask images, and has access to the bounding_box files. Every information is printed on the screen to give us feedback on functionality of the code.

Three folders must be added in order to make it run:

1) first_course_tempaltes
2) second_course_templates
3) side_dish_templates

which we provide with the code. These template files are utilized for object detection template matching algorithm.

2 Food Annotation

To start, we created template images by extracting specific object parts from the original images. These templates were generated for each ID, with the number of templates depending on the object's frequency in the images. Our aim was to use a template matching algorithm for effective object detection. For this purpose, we found that a simple template matching algorithm worked well for before images.

Template matching involves sliding a template over the entire image to find the best match patch and localize the object. To handle changes in illumination, we utilized the zero-mean normalized cross correlation (ZNCC) similarity metric. However, we encountered difficulties in using template matching for object detection when the plates had varying shapes, particularly for difficulty levels 1, 2, and 3. To address this, we experimented with incorporating rotation (0, 30, 60, 90, 120, 180) and scale (0.4, 0.6, 0.8, 1.0) invariance into the template matching process. As the difficulty increased, there were more scale and rotation changes to consider. Additionally, with higher difficulty, there was less leftover food in the plates, and the changing shape of the remaining food made object detection more challenging.

Even though there are slight improvements, the expectations did not meet the results.

Tray 1, before image and difficulty 1 localization

Then, we explored SIFT features and descriptors-based matching. SIFT offers advantages such as scale and rotation invariance, as well as robustness to occlusions which might be particularly useful for our case. We hypothesized that there would be similarities between partially covered food and food that has been eaten, making SIFT effective in such scenarios. However, we encountered challenges in finding sufficient descriptors for matching the images with the templates. As a result, we decided to increase the threshold for Lowe's ratio test (KNN-based matching). Despite these efforts, the expectations did not align with the results, leading us to quit the use of SIFT-based matching. Instead, we opted to rely on Naïve template matching and matching with rotation-scale invariance.

To further enhance the results, one approach would be to include more templates to increase the similarity of objects in the images. However, we chose not to pursue this direction to maintain the generalization capability of our system. Another alternative would involve employing deep learning models with one-shot learning by fine-tuning the model using our templates. However, since our templates were specifically designed for testing purposes and we lacked additional data, we decided to stick with the computer vision approach.

3 Food Segmentation

In the segmentation phase of our system, we conducted an analysis of various techniques such as Thresholding, Region Growing, Watershed, and Grabcut. However, we observed that simple Thresholding and Region Growing methods did not yield desirable outcomes in accurately segmenting the images. While implementing the Watershed algorithm, we encountered a specific challenge related to marker detection. Due to color similarities among different regions within the images, automatically obtaining reliable markers became a complex task.

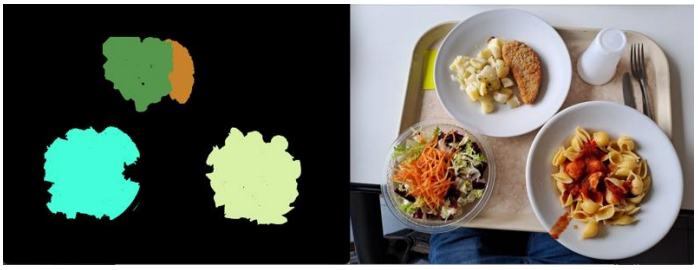
Markers play a vital role in the Watershed algorithm as they help identify and separate distinct regions or objects based on intensity or gradient information. However, the presence of color similarities hindered the automated

detection of suitable markers, subsequently impacting the effectiveness of the Watershed algorithm in correctly segmenting the images. This limitation necessitated additional considerations, such as manual intervention or exploring alternative algorithms that could better handle color similarity challenges in marker detection.

Therefore, we changed our method again and implemented only Grabcut algorithm that works for objects in bounding boxes which is suitable for our task.

In the Grabcut Algorithm, we made use of the GC_BGD (obvious background), GC_PR_BGD (possible background), GC_FGD (obvious foreground/object), and GC_PR_FGD (possible foreground) options to effectively differentiate between foreground and background objects, even in cases where color similarities posed a challenge. By sequentially assigning these options, we were able to identify and emphasize the foreground objects more prominently within the image. This approach allowed us to leverage the inherent color similarities to our advantage and achieve better segmentation results.

To evaluate and optimize our system, we primarily utilized ground truth annotations. We aligned our segmentation approach with these reference annotations, with the goal of achieving optimal alignment with the bounding boxes generated by our system. Our objective was to fine-tune the segmentation algorithm specifically for the provided ground truth annotations, in order to enhance performance and ensure better alignment between the generated segmentation results and the identified bounding boxes. Presented below are some of the results obtained from this process.



Tray 2, food_image, with ground truth annotations

Tray 2, leftover2, with truth annotations

4 Evaluation

The evaluation part of our system, as required by the project, was divided into three parts: localization, segmentation and food leftover estimation.

The evaluation tests were validated, firstly, on the 'before images', as our system uses the images of complete dishes as template dataset, we therefore expected to obtain very good results at least in the "before images" of each tray.

However, due to this we mainly encountered underfitting problems on the localization of the images resulting in the generation of many false negatives (unrecognized trays) and a few false positives (invented trays) in the “leftover” images.

For the evaluation of the metrics, a system for detecting IDs within the files related to bounding boxes has been implemented. In short, all predicted locations are compared with the corresponding .txt file of the reference image. If the ID is present in the file, the cv::Rect corresponding to that ID is extracted and saved in a specific vector for the comparison of each detected prediction. If the

ID is not present in the text file, then we have a false positive, which will be compared with an equal cv::Rect of (0,0,0,0), allowing us to immediately identify the corresponding case for both mAP and mIoU.

4.1 Localization Evaluation

Concerning the localization assessment, we had to implement the mAP with an IoU threshold of 0.5. For the calculation of average Precision, prior assumptions were made due to both the complexity of certain calculation methods and our specific case.

Since we refer to trays in a school canteen, we expect to always find a first course, a second course and side dishes, as is also demonstrated by the pictures (although such a small number of trays is not very valid, we rely on empirical evidence). From this it follows that the calculation of average precision using PASCAL VOC 11 Point Interpolation Method where precision values are recorded across 11 equally spaced recall values may be inefficient with our type of task.

The traditional approach to computing average precision (AP) in object detection involves creating a precision-recall curve and then integrating the area under the curve to obtain the average precision. This curve provides insights into the trade-off between precision and recall for different classification thresholds.

While the precision-recall curve provides a more comprehensive evaluation of the detection algorithm's performance, calculating it can be complex and computationally intensive. Therefore, in ours, a simpler approximation is used, which involves calculating the average precision as the arithmetic mean of precision values at fixed recall values. This approximation simplifies the process; however, it may not capture the full performance characteristics of the algorithm as effectively as the complete precision-recall curve.

So, we simply calculate the precision value for each class detected considering a threshold of IoU = 0.5, and then compute the mean Average Precision (mAP).

The IoU for localization calculates the intersection and union of bounding boxes through simple operations between cv::Rect, which were also seen in a similar manner during the labs.

An example of output for the before image of the tray1 is this:

```
-------------- Evaluation mAP --------------
Precision for ID 1: 1
Precision for ID 6: 1
Precision for ID 10: 1
Precision for ID 13: 1

 - mean Average Precision (Before): 1
-------------------------------------------
```

4.2 Segmentation Evaluation

Segmentation evaluation, on the other hand, required applying an mIoU calculated on the predicted segmentation masks against the masks extracted from the images in the masks folder (with a process similar to extracting bounding boxes).

The process required us to modify our code to create not only the segmentation of our predicted localizations, but also the segmented image containing the masks in the same manner.

Our segmentation function has also been modified to the image containing the overall segmentations, it saves one image per performed segmentation in an image vector.

The resulting vector contains images with the ith segmented dish in the foreground in white (255), and the background in black (0).

All this so that in the main function we could have two vectors ready for comparison using mIoU function, both ordered so that the ith elements coincide with each other.

For the implementation of the mIoU, we want to implement two specific functions. One for the calculation of the IoU between real mask and prediction and the other for the actual calculation of the mIoU.

The implementation of the IoU with respect to the function for localization was necessarily different since in this case we are dealing with masks whose variable type is cv::Mat whereas in the case of localizations it was of type cv::Rect, obviously.

The functions implemented for intersection and union were bitwise_and and bitwise_or respectively.

The information printed on the screen, as with the mAP, refers to both individual elements and the desired final value.

Here is an example relating to the tray1 of the before image:

```
--------------- Evaluation mIoU ---------------
Intersection over Union for ID 1: 0.965608
Intersection over Union for ID 6: 0.869031
Intersection over Union for ID 10: 0.746998
Intersection over Union for ID 13: 0.85167

 - mean Intersection over Union (Before): 0.858
-----------------------------------------------
```

5 Food Leftover Estimation

In the food leftover estimation phase, our goal is to detect the quantity of leftover food using our algorithm and compare it to the ground truth annotations. By analyzing segmented food regions and assessing the variance between our estimated quantity and the ground truth, we aim to improve the accuracy of our estimation approach.

In our pursuit of accurately estimating the quantity of leftover food, we iteratively refined our code by leveraging the information from ground truth bounding boxes. Initially, we attempted to calculate the number of non-black pixels within each bounding box, treating each object and its corresponding annotation separately. However, we encountered challenges when comparing our results with the ground truth masks provided for evaluation. The issue arose due to the presence of overlapping dishes (e.g., meat overlapping with potatoes) in the images, which resulted in an overestimation of the pixel count.

Considering the challenges posed by overlapping dishes, we made the decision to adjust our approach. Our revised method involves performing a full image segmentation to capture all the dishes simultaneously, similar to the ground truth masks. We then record the colors corresponding to each segmented dish and calculate the pixel counts based on these colors. This new strategy enables us to accurately calculate the pixels, accounting for the overlapping nature of the dishes (such as foods on the same plate). By adopting this approach, we aim to improve the accuracy of our food leftover estimation algorithm.

In our food leftover estimation process, we utilized the following formula for calculating R_i. We provided the results in two different ways. The first option involved considering the entire segmented image and calculating R_i based on that. The second option involved calculating R_i for each object separately and providing a tuple of R_i values for each dish.

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

However, due to the varying order of dishes and the potential disappearance of some dishes in the second and third difficulty images, we addressed this issue by adopting a sequential approach (using category_id). We treated the dishes in a sequential manner and padded the non-existing dishes with a count of 0 pixels, based on their presence in the previous image. This sequential padding ensured that we maintained consistency in our calculations, even when certain dishes were

missing. Below are some of the results we obtained using this sequential padding technique:

```
Ri score for predicted image: 0.687166
Ri score for mask image: 0.57351
************************************************************
Ri scores for before after image for each food sorted by id:
0.572604
0.672176
0.78956
0.74066
************************************************************
number of pixels in before image or each food sorted by id:
91049
15731
54405
95261
************************************************************
number of pixels in after image for each food sorted by id:
52135
10574
42956
70556
```

6 Conclusion

In conclusion, our system for object detection, segmentation, and evaluation has demonstrated relatively good results for segmentation but not for localization. We first focused on localization, accurately detecting and identifying objects within the images. Leveraging the localization results, we employed the Grabcut algorithm for segmentation, effectively separating foreground objects from the background, even in challenging scenarios with color similarities. The segmentation process significantly improved the accuracy of object boundaries and further refined our results. Finally, we conducted a comprehensive evaluation, assessing metrics such as average precision (mAP) for localization, mean intersection over union (mIoU) for segmentation, and estimating food leftovers. The evaluation results provided valuable insights into the system's performance and highlighted areas for optimization. Overall, our approach showcases the effectiveness but also downsides of traditional techniques like combining localization, segmentation, and evaluation to achieve accurate object analysis and holds potential for further advancements in various applications.

```
---------- Before Image ----------

---------- Evaluation mAP ----------
Precision for ID 1: 1
Precision for ID 6: 1
Precision for ID 10: 1
Precision for ID 13: 1

 - mean Average Precision (Before): 1
-----------------------------------------------

---------- Evaluation mIoU ----------
Intersection over Union for ID 1: 0.965608
Intersection over Union for ID 6: 0.869031
Intersection over Union for ID 10: 0.746998
Intersection over Union for ID 13: 0.85167

 - mean Intersection over Union (Before): 0.858327
-----------------------------------------------
```

Before image evaluation results- Tray1

```
---------- Leftover 1  ----------

---------- Evaluation mAP ----------
Precision for ID 1: 1
Precision for ID 8: 0

 - mean Average Precision (Before): 0.5
-----------------------------------------------

---------- Evaluation mIoU ----------
Intersection over Union for ID 1: 0.54305
Intersection over Union for ID 8: 0

 - mean Intersection over Union (Before): 0.271525
-----------------------------------------------
```

Difficulty 1 image evaluation results- Tray1

```
---Leftover with difficulty 2---

---------- Evaluation mAP ----------
Precision for ID 1: 1
Precision for ID 8: 0

 - mean Average Precision (Before): 0.5
-----------------------------------------------

---------- Evaluation mIoU ----------
Intersection over Union for ID 1: 0.468049
Intersection over Union for ID 8: 0

 - mean Intersection over Union (Before): 0.234025
-----------------------------------------------
```

Difficulty 2 image evaluation results- Tray1

7 Contributions

Onur Alp (55 hours): Segmentation, Ri calculation, reporting

Suleyman (55 hours): Localization, code integration, evaluation, reporting

Stefano (55 hours): Evaluation, code integration, localization, reporting

- All the output (localization results, segmentation results, evaluation) can be found in the results folder.