THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

Final Project Report

Chirag Lakhanpal

The George Washington University

DATS 6303: Deep Learning

Professor AmirJafari

Table of Contents

Introduction

The Food Detection Benchmark 2022 dataset is a multi-faceted challenge aiming at improving and assessing computer vision systems for food detection tasks. This report details my contributions to a project that makes use of this dataset. My research centered on improving the accuracy and efficiency of object identification and segmentation models, namely through data preparation, augmentation approaches, and hyperparameter tuning. Finally presenting everything through a user-friendly interface via a Streamlit app. The parts that follow will go into the approaches I used, the outcomes I obtained, and the lessons I learned from this project.

Dataset

Link - https://www.aicrowd.com/challenges/food-recognition-benchmark-2022

Individual Work (My contribution and Work Description)

In my individual work on the project, I began by performing exploratory data analysis (EDA), preparing the dataset, pre-processing the data, and testing with various models to improve performance. These contributions are summarized in the following paragraphs:

Explanatory Data Analysis

1. Category Distribution

Findings: The findings from the data visualization indicate that the dataset contains various food categories with differing image counts. The category 'Water' has the highest count, and the category "Veggie Burger" has the lowest, suggesting a potential class imbalance.

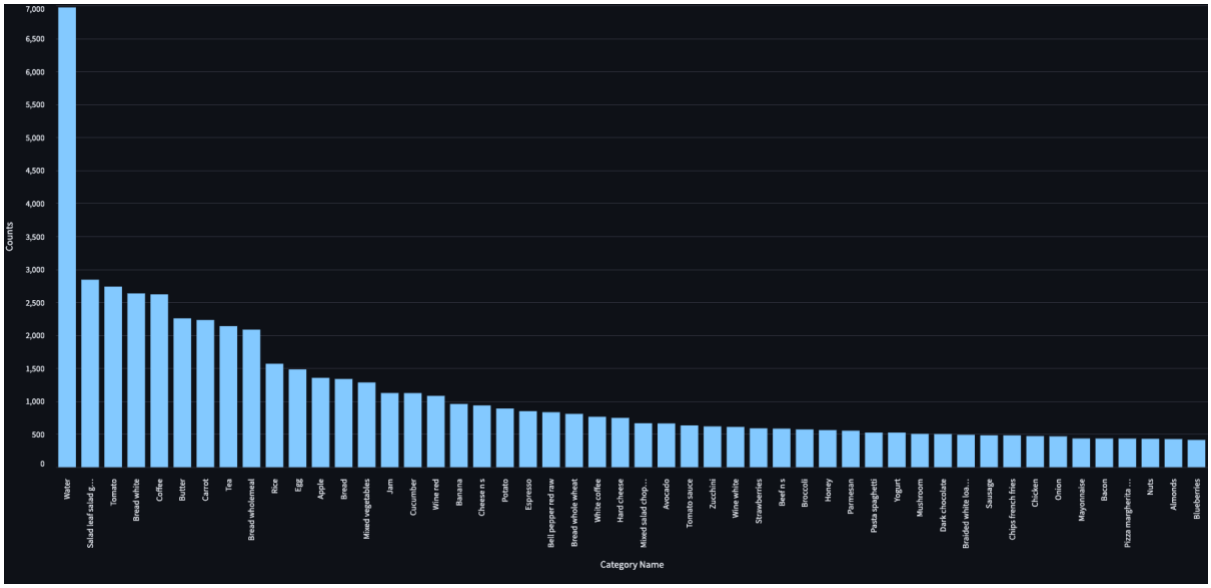


Figure 1: Frequency of Food Categories. The chart shows the number of images for each food category in the dataset, with 'Water' being most common.

2. Distribution of heights and widths of images

Findings: The scatter plot reveals a positive correlation between image width and height, indicating that as images get wider, they also tend to get taller. There are few potential outliers. This is for setting up image sizes in as a parameter to the model.

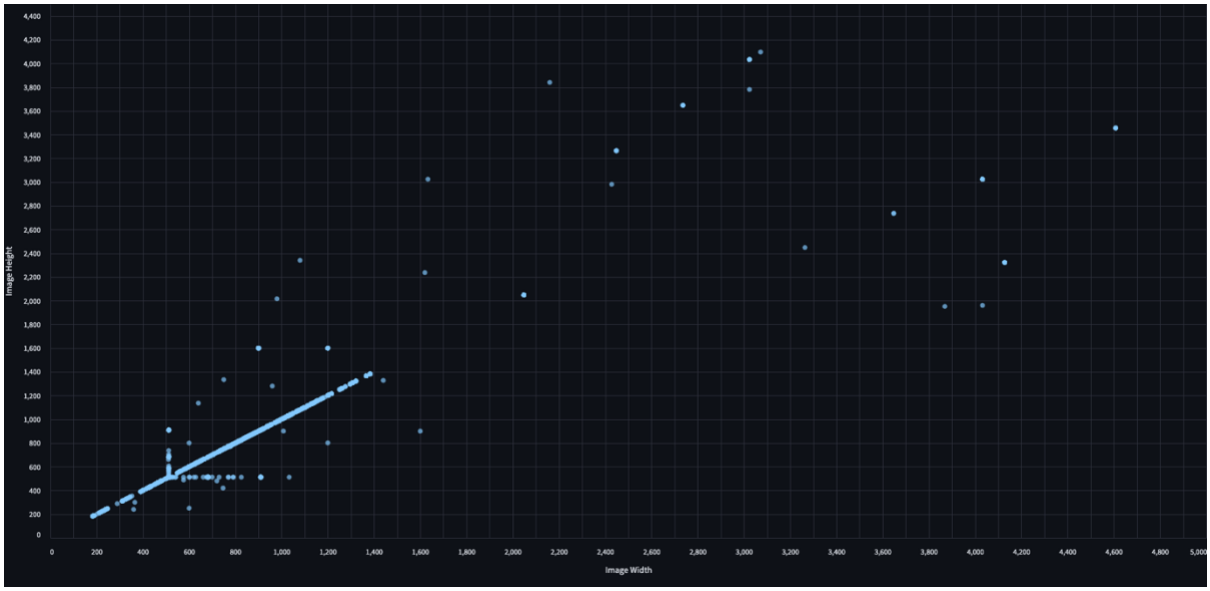


Figure 2: Scatter plot of Image Dimensions. This plot shows the relationship between image width and height in the dataset, with a trend indicating a proportionate increase in height with width.

3. Distribution of heights and widths of images

Findings: The density plot exhibits a sharp peak at an aspect ratio of 1, indicating a large number of square images in the dataset, with fewer rectangular images.

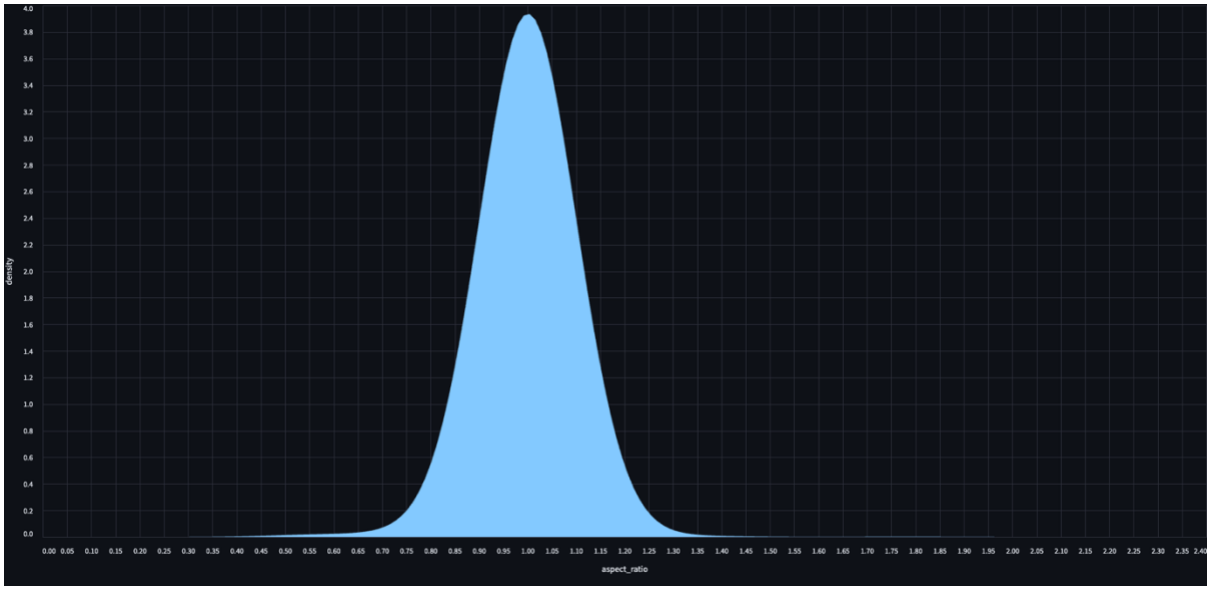


Figure 3: Aspect Ratio Distribution. The density plot showcases the predominance of square images within the dataset.

4. Distribution of annotation counts per image (log scale)

Findings: The logarithmic distribution graph indicates most images have a low number of annotations, with a steep decline as the annotation count per image increases.

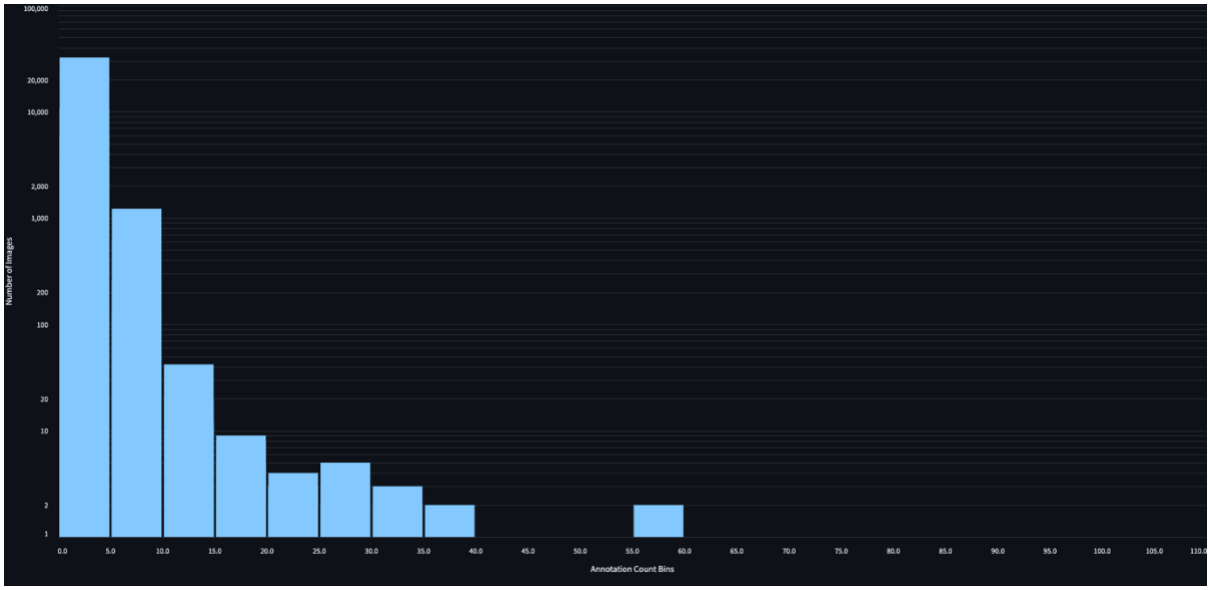


Figure 4: Logarithmic Distribution of Annotations per Image. The graph shows the skewed distribution towards images with fewer annotations.

5. Distribution of area of annotations per category

Findings: The treemap depicts the proportion of annotation regions across several dietary categories, with 'Water' taking up the most space, indicating its widespread presence in the dataset.

Figure 5: Treemap of Annotation Areas by Category. This visualization represents the relative size of annotations for each food category, highlighting the dominance of certain categories like 'Water' and 'Salad Leaf Salad Green'.

Data Preprocessing

Issue Identified	Solution Implemented
Misaligned segmentation and bounding boxes	**Adjusted bounding boxes based on segmentation data**
Incorrect image dimensions	**Updated image dimensions using actual image files**
Rotated images	**Removed rotated annotations from the dataset**
Non-food item annotations	**Excluded annotations not related to food items**
Small bounding boxes	**Small bounding boxes retained to assess model performance (Did not improve the model)**
Rotated annotations	**Removed annotations with a rotated bounding box mode**
Images with aspect ratio > 2	**Choose to remove them**

Data Augmentation

For enhancing the model's ability to generalize from the Food Recognition dataset, I applied a series of data augmentation techniques. The transformations included:

- Rotating images by up to 30 degrees to introduce variability in orientation.
- Zooming in on images by a factor of up to 1.5 to simulate different image scales.
- Modifying image brightness and contrast by up to 40% to simulate varying lighting conditions.
- Warping images slightly with a warp factor of up to 0.4 to introduce geometric distortions.
- Random erasing was also applied to parts of images with a probability of 50% and a maximum of 2 applications, to mimic occlusion and encourage the model to learn from partial views of objects.

Models

I trained two variants of the Mask R-CNN model with different backbones - mask_rcnn_R_50_FPN_3x and mask_rcnn_R_101_FPN_3x and experimented with MMdetection and HTC models. However, due to installation issues, I was unable to make them work, hence, I focused on the Mask R-CNN models, tuning them for the best performance on the given dataset. I have used average precision (AP) as a metric to compare the models.

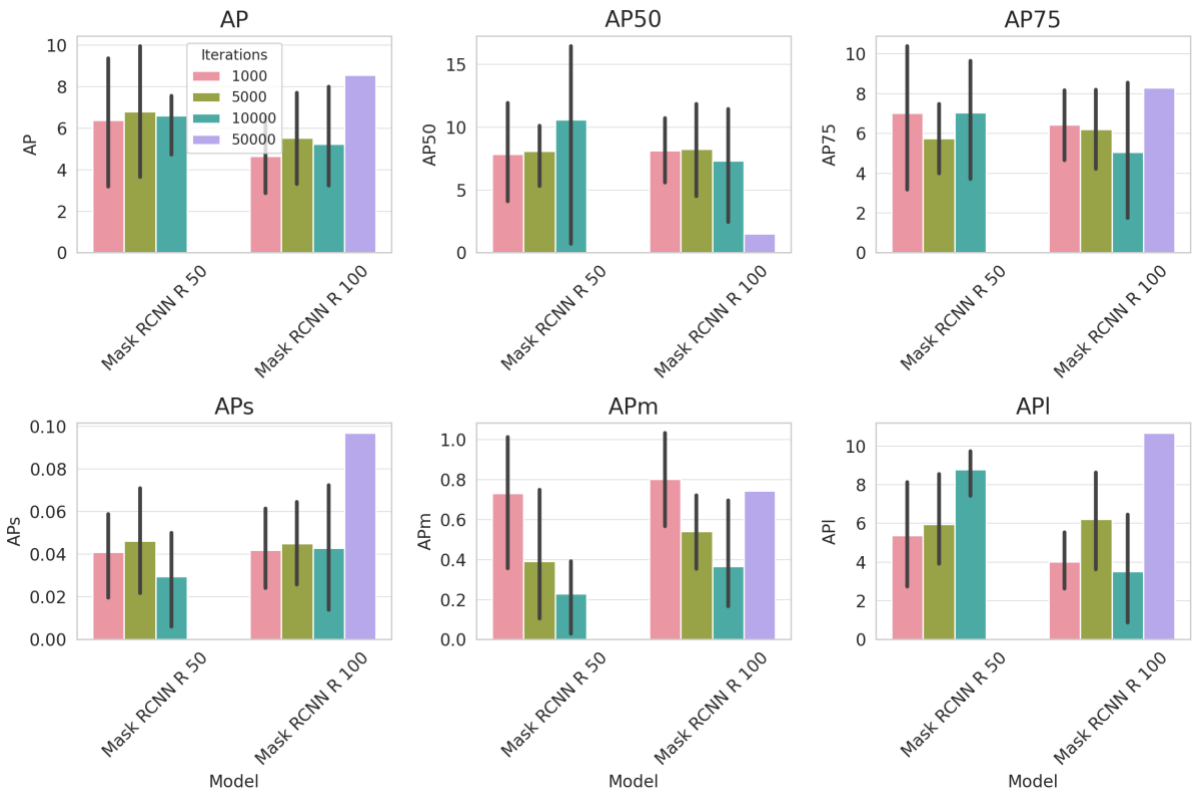
Streamlit App

I created the layout and backend programming for a Streamlit application. My contribution included architecting, creating, and integrating the model's backend features.

Things That Did Not Work

I also attempted to implement MMDetection models, particularly Hierarchical Task Cascading (HTC), but faced a lot of challenges due to library compatibility issues. The primary obstacles were with the mmdet, mmcv, and mmdet-full libraries, essential for MMDetection's architecture and functionalities.

Results and Summary



In my experiments, I hyperparameter tuned the following parameters for object detection models:

- IMS_PER_BATCH: 4, 8, 16
- BASE_LR: 0.00025, 0.01, 0.001, 0.0025, 0.00025, 0.00001, 0.000025
- MAX_ITER : 1,000; 5,000; 10,000; 50,000; 100,000
- LR_SCHEDULER_NAME: "WarmupMultiStepLR"
- CHECKPOINT_PERIOD: 20,000, 10,000
- BATCH_SIZE_PER_IMAGE: 64, 128, 256, 512

The comparison between two models, r50 and r100, was conducted across various performance metrics, as depicted in the provided figure. These metrics include Average Precision (AP), AP at 50% Intersection over Union (AP50), AP at 75% Intersection over Union (AP75), AP small (APs), AP medium (APm), and AP large (APl). The models were evaluated at different iterations: 1,000; 5,000; 10,000; 50,000; 100,000 to understand the impact of training duration on model performance.

Conclusions

- The r50 model generally outperforms the r100 model, suggesting that the former may be more efficient for our object detection tasks.
- All models demonstrated poor performance on small objects, as indicated by the low APs scores. This suggests that further refinement is needed in model training or architecture to enhance the detection of small objects.
- The impact of increased iterations on performance was notable but did not significantly improve the detection of small objects (APs).

Code Contribution

Total lines of code = 549 (Stremlit + train)
Modified lines = ~90
Added lines = ~50

Percentage = 91.98%

References

1. He, K., Gkioxari, G., Dollár, P., & Girshick, R. B. (2017). Mask R-CNN. arXiv preprint arXiv:1703.06870. Retrieved from http://arxiv.org/abs/1703.06870
2. ChatGPT by OpenAI.
3. Facebook Research. (2023). Detectron2. GitHub repository. Retrieved from https://github.com/facebookresearch/detectron2
4. Analytics Vidhya. (2021). Your Guide to Object Detection with Detectron2 in PyTorch. Retrieved from https://www.analyticsvidhya.com/blog/2021/08/your-guide-to-object-detection-with-detectron2-in-pytorch/
5. Justin To Data. (n.d.). Streamlit Python Tutorial. Retrieved from https://www.justintodata.com/streamlit-python-tutorial/