Mini Project 4

1st Amir Ali Pakatchian Fard
Electrical and Computer Engineering (ECE)
Student Number: 810199389

2nd Seyedeh Diba Ravanshid Shirazi
Electrical and Computer Engineering (ECE)
Student Number: 810199431

CONTENTS

***Abstract*—In this project we will use our knowledge from Mechatronics and Robotics course to complete the problems discussed in AI platform. At first we get familiar with what Yolo means and how it works and how each version differs from other versions. We try to analyze a dataset and try to implement YOLO for object detection. After that we get to do object segmentation using fastSAM. Also the grasp point generation is the bonus problem.**

***Index Terms*—YOLO, object segmentation, object detection, fastSAM**

I. PROBLEM 1: YOLO DIFFERENT VERSIONS

YOLO (You Only Look Once) has released several versions over time, each with improvements and changes compared to its predecessor. Below is an explanation of a significant difference in each YOLO version compared to the older version:

A. YOLOv1 to YOLOv2

Difference: YOLOv2, also known as YOLO9000, made improvements in model accuracy and speed. One of the significant changes was the use of Batch Normalization in all convolutional layers, which increased training stability and speed. Additionally, YOLOv2 employed Anchor Boxes for predicting bounding boxes, significantly improving object detection accuracy.

B. YOLOv2 to YOLOv3

Difference: YOLOv3 improved object detection capabilities by using the Darknet-53 architecture instead of Darknet-19. This version also used a multi-scale prediction system, which enhanced the detection accuracy for small objects. Moreover, YOLOv3 utilized Logistic Regression for predicting class scores, which improved model performance.

C. YOLOv3 to YOLOv4

Difference: YOLOv4 included a set of enhancement techniques such as Mosaic Data Augmentation, CSPDarknet53, Self-Adversarial Training, and Cross Stage Partial connections, which significantly increased model accuracy and speed. YOLOv4 also used optimization algorithms like CmBN and DropBlock regularization to improve training stability.

D. YOLOv4 to YOLOv5

Difference: YOLOv5 featured numerous structural changes and optimizations compared to its predecessors. One of the most important differences was the use of PyTorch instead of Darknet as the base framework, which provided greater flexibility and ease of use. Additionally, YOLOv5 achieved significant improvements in inference speed and object detection accuracy.

E. YOLOv5 to YOLOv6

Difference: YOLOv6 offered better performance in terms of speed and accuracy through further optimizations in architecture and learning algorithms. One notable improvement in YOLOv6 was the use of more advanced techniques in Data Augmentation and Regularization, enhancing model stability and accuracy.

F. YOLOv6 to YOLOv7

Difference: YOLOv7 introduced new architectures and further optimizations in the network structure, offering more improvements in object detection accuracy and speed. One significant change in YOLOv7 was the use of new techniques in the network Backbone for better and more efficient feature extraction.

G. Conclusion

These differences highlight the continuous advancements in YOLO technology, improving the performance and applicability of these models in various object detection applications.

II. Problem 2: mAP score

The mean Average Precision (mAP) is a commonly used evaluation metric in object detection tasks to assess the performance of a model. The mAP score provides a single number that summarizes the precision-recall tradeoff across different classes and is calculated as follows:

A. Precision and Recall

- **Precision** is the ratio of true positive detections to the total number of positive detections (both true positives and false positives).
- **Recall** is the ratio of true positive detections to the total number of actual positive instances (true positives and false negatives).

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

B. Precision-Recall Curve

- For each class, a precision-recall curve is plotted by varying the confidence threshold of the detection model.
- The x-axis represents recall, and the y-axis represents precision.

C. Average Precision (AP)

- The area under the precision-recall curve is calculated to obtain the Average Precision (AP) for each class.
- Interpolated precision values are often used to ensure a monotonically decreasing precision-recall curve. The interpolated precision at a certain recall level r is defined as the maximum precision observed for any recall level greater than or equal to r.

$$\text{AP} = \sum_{n} (R_n - R_{n-1}) P_n$$

where P_n and R_n are the precision and recall at the n-th threshold.

D. Mean Average Precision (mAP)

- The mAP is computed by averaging the AP values across all classes.

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i$$

where N is the number of classes and AP_i is the average precision for class i.

E. Steps to Calculate mAP

1) **Compute Precision and Recall**:
   - For each detected object, determine if it is a true positive or a false positive.
   - Calculate precision and recall at various confidence thresholds.
2) **Plot Precision-Recall Curve**:
   - Plot the precision against recall for different confidence thresholds.
3) **Calculate Average Precision (AP)**:
   - Compute the area under the precision-recall curve for each class.
4) **Calculate mAP**:
   - Average the AP scores across all classes to get the final mAP score.

Thus, the mAP score provides a single measure of the model's overall performance across different classes, balancing precision and recall.

III. Problem 3: Project dataset

A. Task 1

As it can be seen in Fig.1 some classes have more data than others but all of them have enough test train and valid data as seen in Fig. 2. Also if we should check the health of the dataset to ensure all images have been labeled and if their sizes are ok so we check it in Fig. 3.

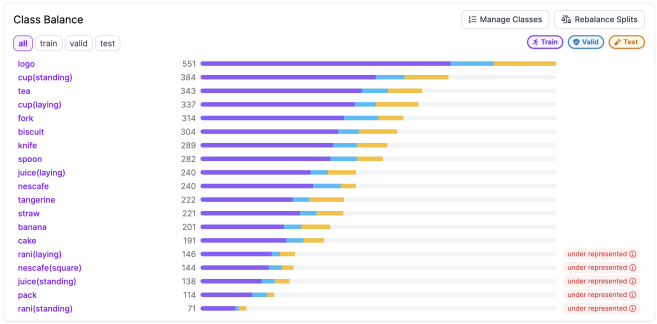

Fig. 1. dataset distribution

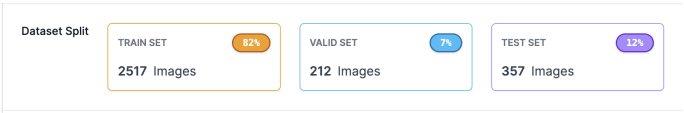

Fig. 2. dataset split

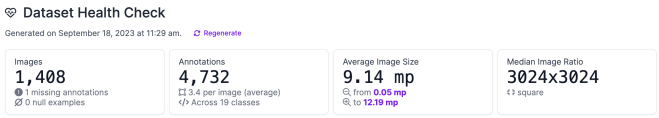

Fig. 3. health check

B. Task 2

To have richer data and because when we want to test our data in the real world, our images might have noise or the objects might rotate, we added some augmentation steps. These steps ensure that our model performs well on test images in real-world scenarios. In another way we use augmentation steps in the process of developing a dataset for several reasons:

1) **Increase Data Variety:** Augmentation artificially expands the size of the training dataset by creating modified versions of images, which helps the model generalize better.
2) **Improve Robustness:** By simulating real-world variations such as rotations, translations, and noise, augmentation makes the model more robust to these changes when encountered in real-world data.
3) **Prevent Overfitting:** Augmentation helps prevent the model from overfitting to the training data by exposing it to a wider variety of examples, making the model more capable of handling unseen data.
4) **Enhance Model Performance:** Through exposure to a diverse set of transformations during training, the model can learn to recognize objects under different conditions, thereby improving its accuracy and performance.
5) **Simulate Real-World Scenarios:** Augmentation can mimic various real-world scenarios (e.g., different lighting conditions, backgrounds, and object orientations), ensuring the model's applicability and effectiveness in practical applications.

Overall, data augmentation is a crucial technique in machine learning that enhances the quality and diversity of the dataset, leading to more effective and reliable models.

C. Task 3

First thing that we need to consider is the size of photos that need to be similar and resized to a certain number. This is done in the preprocessing step. After, we do the augmentation step where we can add noise and rotate the photos, crop them, do some image processing techniques on them, and flip the photos. Here we add noise and rotate the photos in 2 directions by only 15 degrees as seen in Fig. 4.

Fig. 4. augmentation step

These techniques are enough for our purpose. Adding noise is crucial because it's very likely to be experienced while working in real. Also the products may rotate and are not always in the same direction so we should be able to understand and learn these differences. In general, as said before, data augmentation is a crucial technique in machine learning that enhances the quality and diversity of the dataset, leading to more effective and reliable models

IV. Problem 4: Object detection

From here the questions are answered in the notebook itself.