Computer Vision - Spring 25

Assignment 3

Deadline : 8 March 2025 11:55 P.M.

Instructors: Prof Ravi Kiran Sarvadevabhatla
and Prof Makarand Tapaswi

February 21, 2025

1 General Instructions

- Your assignment must be implemented in Python.
- While you're allowed to use LLM services for assistance, you must explicitly declare in comments the prompts you used and indicate which parts of the code were generated with the help of LLM services. **If you haven't specified the prompts, then 50% of your marks will be deducted.**
- Plagiarism will only be taken into consideration for code that is not generated by LLM services. Any code generated with the assistance of LLM services should be considered as a resource, similar to using a textbook or online tutorial.
- The difficulty of your viva or assessment will be determined by the percentage of code in your assignment that is not attributed to LLM services. If during the viva you are unable to explain any part of the code, that code will be considered as plagiarized.
- Clearly label and organize your code, including comments that explain the purpose of each section and key steps in your implementation.
- Properly document your code and include explanations for any non-trivial algorithms or techniques you employ.
- Ensure that your files are well-structured, with headings, subheadings, and explanations as necessary. Your assignment will be evaluated not only based on correctness but also on the quality of code, the clarity of explanations, and the extent to which you've understood and applied the concepts covered in the course.
- Make sure to test your code thoroughly before submission to avoid any runtime errors or unexpected behavior.

- **Report all the analysis, comparison and any metrics in the notebook or a separate report that is part of the submission itself. No external links to cloud storage files, wandb logs or any other alternate will be accepted as part of your submission. Only the values and visualizations as part of your commits will be graded.**

2 Oriented Bounding Boxes through Faster RCNN (200 Marks)

You have been provided with an implementation of Faster R-CNN in your GitHub Classroom. Your task is to understand the codebase and extend it to predict oriented bounding boxes instead of the standard axis-aligned bounding boxes. For all the experiments that follow for this problem, you have to use this dataset from here. Please visualize the dataset before starting. The training and inference files along with the dataloader are provided.

2.1 Understanding Faster RCNN (50 Marks)

Faster R-CNN consists of two main components: the **Region Proposal Network (RPN)** and the **ROI Head**. The RPN is responsible for generating region proposals that may contain objects. To get started, for this subpart, you need to train the model for axis aligned bounding boxes only. Look at **train.py** for getting started.

For this subpart, you are expected to understand the code, and do a set of visualizations for different modules in the code. These visualizations will depend on a lot of hyperparameters. You are expected to understand the role of each of them, and do the same set of following visualizations for **atleast 3** different sets of hyperparameters.

1. For the RPN, please understand the **RegionProposalNetwork** and the **RPNHead** class. Visualize the following for a fixed validation batch during training.

    - **Visualizing predicted objectness map** : Visualize the evolution of objectness score heatmap for every feature map level coming from the backbone for this batch of images. Visualize it using a video over different validation iterations at different timesteps of training.
    - **Visualizing predicted object proposals** : Create an animation demonstrating how object proposals change over iterations. Essentially for every image in this batch, visualize the bounding box proposals that ultimately go to the ROI Head, on the image itself, for successive validation iterations, and play that as an animation.
    - **Visualizing bounding box assignments** : During the training of RPN, we sample both positive and negative anchors corresponding to the ground truth, that have a high and a low overlap with the ground truth boxes. Visualize the positive anchors with green color and negative anchors with red color on the image itself. While visualizing, limit the number of positive and negative anchors to 10. Also, visualize the complete set of anchors for all the predicted feature maps of the image.

(35 Marks)

2. For the ROI Heads, please understand the **ROIHeads** class. Visualize the following for the same validation batch during training:

   - The ROI head ouptuts the classification scores and the bounding box regression terms, on taking the ROI of the object proposals coming from the RPN. Visualize the difference between the proposal given by the RPN and the final bounding box predicted predicted by the ROIHeads on the same image, at successive iterations of training by plotting both of these with a different color. Visualize the classification score as well on top of the predicted bounding box.

   (15 Marks)

Note - You can make these visualizations by either storing the checkpoints explicity during training and then visualizing, or alternatively saving the visualized images during training itself. You can use **OpenCV** for making the animations and the videos.

2.2 Extending Faster R-CNN for Oriented Bounding Boxes (100 Marks)

For predicting oriented boxes instead of axis aligned boxes, you will need to modify the ROIHead class as well as the network head itself for predicting the angles along with the classification scores and the bounding boxes.

Note - You don't need to change the RPN class and can assume that the proposals themselves are axis aligned.

1. Modify the code to predict oriented bounding boxes:

   - Modify the dataset class for reading angles as well. Preprocess these and visualize the annotations for a fixed set of images.
   - Extend the head to predict an additional angle parameter along with class scores and bounding box regression.
   - Implement two different approaches for angle prediction:
     - Direct angle regression.
     - Multi-bin classification for discretized angle prediction. Here you will need to discretize the angles based on a fixed step size.

   Essentially, angle prediction will yield an additional loss term. Add this term to the existing loss terms by using a corresponding weight factor for this. You will need to tune the weight factor here.

2. Adjust the evaluation metrics:

- Modify the Mean Average Precision (mAP) calculation to account for oriented bounding boxes.
- Log Precision, Recall, and mAP throughout training.

3. Tune the hyperparameters of the model:
   - The loss weight for the angle prediction. Try 3 different weights from {0.1, 1, 10}. Compare the loss graphs for these 3 for a few epochs (say 10), and then use the weight giving a lower loss value.
   - Try atleast 2 different bin sizes for multi-bin classification for the angle prediction.
   - Try with and without image flipping augmentation. You may optionally add other augmentations as well.

2.3 Analysis (50 Marks)

1. Evaluate the models:
   - Compute mAP at different IoU thresholds for oriented bounding boxes.
   - Analyze Precision and Recall metrics.
   - Perform a qualitative analysis of detection results.
2. Visualize results:
   - For a subset of validation images, visualize predictions overlaid with ground truth.
3. Analyze model performance:
   - Identify and discuss failure cases.
   - Architectural changes to improve oriented bounding box detection.
   - Alternative methods to enhance accuracy.
   - Convergence behavior and challenges faced.

Note: This problem encourages exploration of advanced techniques. Your grade will be based on:

- Technical soundness of modifications.
- Depth of analysis and insights.

3 Fruit Detection and Counting (100 Marks)

3.1 Dataset Preprocessing and Visualization (25 Marks)

1. Load the dataset provided here. The dataset contains images along with corresponding segmentation mask annotations of fruits. Process the segmentation masks to extract bounding box coordinates. (5 Marks)
2. Create a function `masks_to_boxes(mask)` that converts segmentation masks to bounding box annotations. Implement appropriate techniques to handle overlapping instances and edge cases. (10 Marks)
3. Implement visualization functions to display:
   - Original images with overlaid bounding box annotations
   - Segmentation masks converted to bounding boxes
   - Side-by-side comparison of original masks and generated boxes

   Display examples of each visualization type for at least 5 different images from the dataset. (10 Marks)

3.2 Dataset and Model Implementation (40 Marks)

1. Implement a PyTorch Dataset class `FruitDetectionDataset` that:
   - Loads images and corresponding annotations
   - Converts masks to bounding boxes on-the-fly or loads pre-computed boxes
   - Implements appropriate data augmentation techniques (consider rotations, flips, and color jittering)
   - Returns properly formatted input-target pairs for Faster R-CNN training

   (15 Marks)
2. Implement Faster R-CNN with the following specifications (you may adapt and reuse relevant encoder implementations from Question 1):
   - Use ResNet-34 as the backbone feature extractor (the encoder implementation from Question 1 can be modified and reused for this purpose)
   - Initialize the backbone with pre-trained weights
   - Configure appropriate anchor scales and aspect ratios specific to fruit detection
   - Include proper normalization and training configurations
   - If reusing code from Question 1, clearly document the modifications made to adapt it for object detection

Provide a detailed architecture diagram showing all components, highlighting any parts adapted from Question 1. (25 Marks)

3.3 Training and Detection Analysis (20 Marks)

1. Analyze the detection pipeline:

   - Justify your choice of anchor box configurations
   - Discuss the role of Non-Maximum Suppression in fruit detection
   - Explain strategies for handling overlapping fruits
   - Analyze the impact of IoU thresholds on detection performance

   (10 Marks)

2. Implement and compare different strategies for improving detection:

   - Test various data augmentation techniques
   - Experiment with different confidence thresholds
   - Analyze the impact of image resolution on detection accuracy

   Compare the effectiveness of each approach on the validation set. (10 Marks)

3.4 Analysis and Evaluation (10 Marks)

1. Evaluate your model using appropriate metrics:

   - Mean Average Precision (mAP) at different IoU thresholds
   - Precision-Recall curves
   - False Positive and False Negative analysis

   (10 Marks)

2. Analyze detection performance specifically focusing on:

   - Impact of occlusion on detection accuracy
   - Effect of fruit color variations and lighting conditions
   - Performance on densely clustered fruits versus isolated fruits
   - Model's behavior with fruits at different scales

   (5 Marks)

3.5 Generalization Analysis (5 Marks)

1. Test your model on external fruit images and analyze:

   - Performance on different fruit varieties
   - Impact of background variations
   - Detection accuracy under different lighting conditions

   (5 Marks)

2. Discuss potential modifications needed for:

   - Adapting the model to detect multiple fruit types
   - Improving robustness to environmental variations
   - Handling different growth stages and ripeness levels

   (5 Marks)

4 Human Parts Detection (100 Marks)

4.1 Dataset Processing and Analysis (30 Marks)

1. The modified dataset is available here. The dataset contains images with corresponding part segmentation masks for human body parts. Your first task is to:

   - Convert the segmentation masks to bounding box annotations
   - Create appropriate data structures to store the annotations
   - Split the dataset into training and validation sets

   Document your approach and justify your choices. (15 Marks)

2. Perform exploratory data analysis on the dataset:

   - Analyze the distribution of different parts
   - Visualize examples of converted bounding boxes overlaid on original images
   - Study the variations in scale, aspect ratio, and positioning of part annotations
   - Report any challenges or interesting patterns you observe in the data

   (15 Marks)

4.2 Model Selection and Implementation (40 Marks)

1. Choose an object detection architecture for fine-tuning. You may select from but are not limited to:

   - Faster R-CNN family
   - DETR family
   - YOLO family
   - RetinaNet

   Justify your choice of architecture based on the characteristics of the parts detection task. (10 Marks)

2. Implement your chosen approach:

   - Set up the model with appropriate modifications for parts detection
   - Design suitable anchor boxes or detection priors (if applicable)
   - Implement necessary data augmentation strategies
   - Configure the training pipeline with appropriate hyperparameters

   Document all modifications made to adapt the base architecture for parts detection. (30 Marks)

4.3 Training and Evaluation (20 Marks)

1. Train your model and maintain proper documentation of:

   - Training configurations and hyperparameters
   - Resource utilization and training time
   - Convergence behavior and any training challenges

   (10 Marks)

2. Evaluate your model using:

   - Mean Average Precision (mAP) at different IoU thresholds
   - Part-specific precision and recall metrics
   - Analysis of performance across different object categories
   - Qualitative analysis of detection results

   (10 Marks)

4.4 Analysis and Discussion (10 Marks)

1. Provide a detailed analysis of your model's performance:
   - Compare performance across different body parts and categories
   - Analyze failure cases and success cases
   - Discuss the impact of part size, occlusion, and viewpoint variation

   (5 Marks)

2. Suggest potential improvements:
   - Architectural modifications that could better suit parts detection
   - Alternative approaches to handling challenging cases
   - Ideas for incorporating part relationships or hierarchical structure

   (5 Marks)

Note: This is an open-ended problem. You are encouraged to explore creative solutions and modifications to standard object detection approaches. Your grade will be based on:

- Technical soundness of your approach
- Quality of implementation and documentation
- Depth of analysis and insights
- Innovation in addressing parts-specific challenges

Good luck with the assignment!