312540021 / Kezia Nathania (林紀霞)

Visual Recognition HW2 Report

Introduction

In this homework there are two tasks:

1. Predict class and bounding box of each digit in the image
2. Recognize the number of detected digits in the image

Method

1. Faster R-CNN

Faster R-CNN is a two-stage object detection framework that generates region proposals and classifies them. It starts with a convolutional backbone network (like ResNet) to extract feature maps from the input image. These features are passed to a Region Proposal Network (RPN), which predicts potential object regions (Region of Interests or RoIs) using anchor boxes and refines their locations. In the second stage, the proposed regions are aligned with the feature maps using ROIAlign, and each region is then classified into object categories and further refined in terms of bounding box coordinates. The entire architecture is trained end-to-end, allowing both region proposal and object detection to be learned simultaneously, making it more efficient and accurate than earlier methods.

2. Resnet50 Backbone

In Faster R-CNN, ResNet-50 is used as a backbone to extract deep feature maps from the input image. It is a 50-layer deep convolutional neural network that uses residual connections (skip connections) to help train very deep networks effectively. ResNet-50 is typically divided into several stages (conv1 to conv5), and in Faster R-CNN, the early layers extract low-level features while the deeper layers extract high-level semantic information. These features are then passed to the RPN for region proposal. When combined with a Feature Pyramid Network (FPN), ResNet-50 also provides multi-scale feature maps, improving the detection of objects of various sizes.

3. Resnet101 Backbone

In Faster R-CNN, ResNet-101 is used as a deep backbone network to extract rich feature maps from input images. It is a 101-layer convolutional neural network with residual (skip) connections, which help train very deep models by preventing vanishing gradients. Like ResNet-50, it processes the image through multiple stages (conv1 to conv5), capturing low-level and high-level features. Compared to ResNet-50, ResNet-101 has more layers, allowing it to learn more complex representations, which can improve detection accuracy, especially for challenging or fine-grained objects.

Combined with a Feature Pyramid Network (FPN), it also provides multi-scale features for better performance on objects of different sizes.

4. Custom Anchor

To improve digit detection, the anchor generator in Faster R-CNN is customized to better match the size and shape of digits in cropped images like those in the dataset. Since digits typically range from 50 to 500 pixels and are often centered, anchor sizes of 64, 96, 128, 256, and 384 are used. Aspect ratios are also customized to 0.5, 1.0, and 1.5 to handle digits that may be narrow or slightly wide. This setup helps the model propose regions that better fit the digit shapes, which can improve detection accuracy.

5. Non-Maximum Suppression (NMS)

To reduce redundant detections, we applied Non-Maximum Suppression (NMS) using the implementation provided by torchvision.ops.nms. This technique selects the bounding box with the highest confidence score and suppresses other overlapping boxes that have an Intersection over Union (IoU) greater than a specified threshold. In our case, we used an IoU threshold of 0.3, which means that any box overlapping more than 30% with a higher-scoring box is eliminated. This process is repeated iteratively, ensuring that only the most confident and non-overlapping detections are retained. As a result, NMS effectively filters out duplicate predictions while preserving accurate object localizations.

6. Thresholding

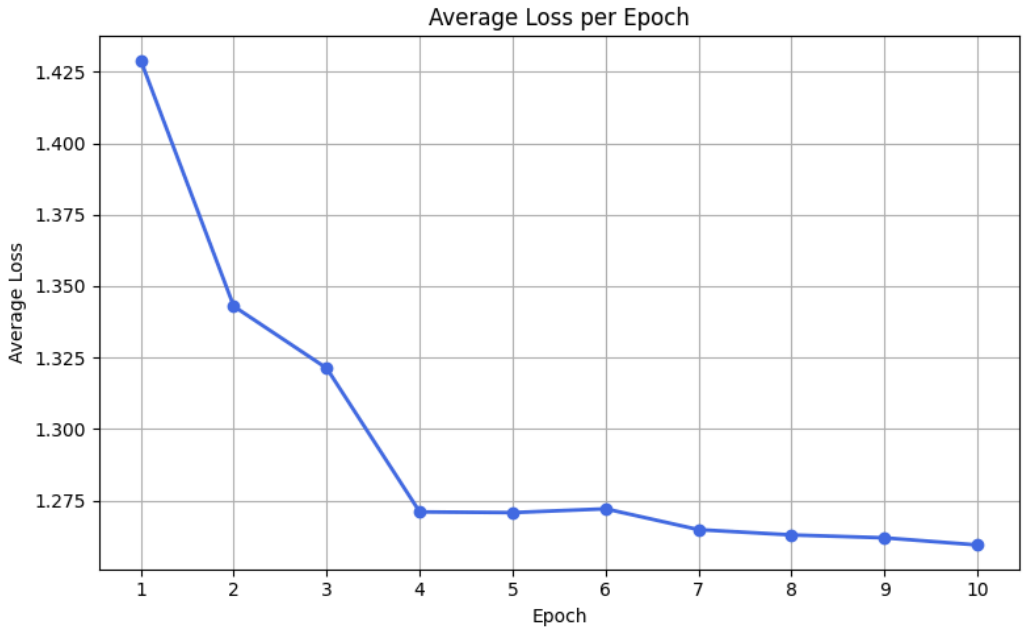
To ensure reliable predictions, the detected bounding boxes are filtered based on their confidence scores. Specifically, any bounding box with a score lower than 0.65 is excluded from the final output. This threshold helps reduce false positives by retaining only the boxes with higher certainty in their predictions.

7. Number Recognition (Task 2)

To identify the number shown in the image, the individual digits detected in Task 1 are first organized based on their horizontal positions. This is done by sorting the detected bounding boxes using their x1 coordinate, which represents the leftmost edge of each box. Sorting ensures that the digits are arranged from left to right, matching their natural reading order. After sorting, the class labels of the digits (i.e., their predicted numerical values) are sequentially concatenated to reconstruct the full number displayed in the image.

Results

1. Resnet50
    a. Training loss

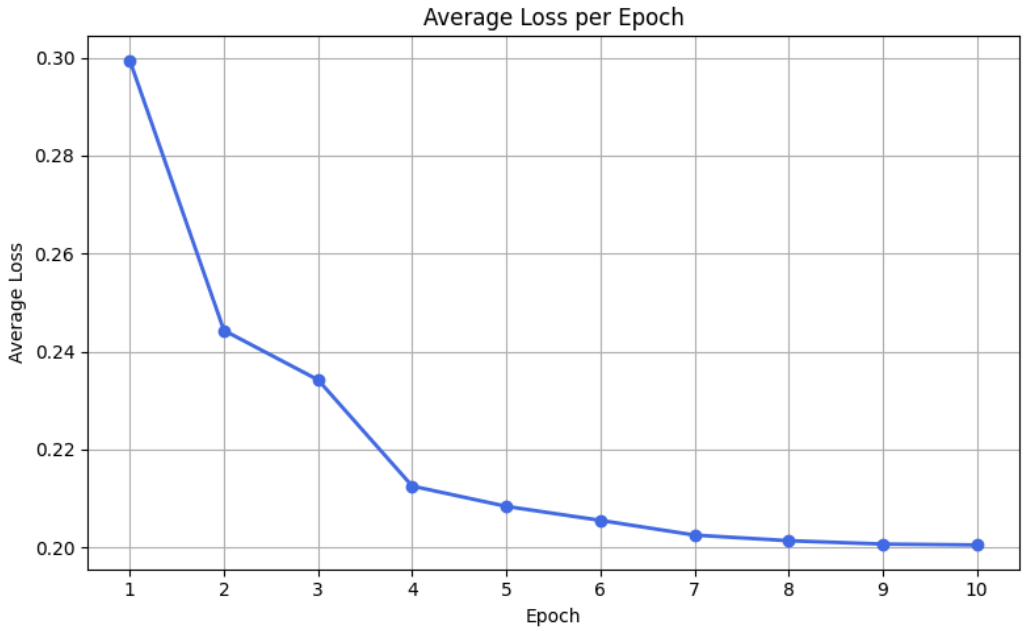


    b. Performance on Validation Set

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.401
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.783
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.353
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.445
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.636
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.439
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.468
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.468
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.458
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.504
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.639
Mean Average Precision (mAP): 0.4006
```

2. Resnet101

a. Training Loss



b. Performance on Validation Set

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.420
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.825
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.362
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.410
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.456
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.585
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.461
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.493
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.493
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.485
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.521
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.593
Mean Average Precision (mAP): 0.4196
```

3. Final Result

Resnet101 is chosen for the final result since it gives better training loss and validation mAP. Below are the mAP score and accuracy score on Faster R-CNN with Resnet101 as the backbone:

80	kezia	1	2025-04-16 16:47	268860	312540021	**0.35**	**0.71**

References

PyTorch websites:
https://pytorch.org/vision/0.19/_modules/torchvision/models/detection/faster_rcnn.html
https://pytorch.org/vision/main/models/generated/torchvision.models.detection.fasterrcnn_resnet50_fpn.html

GitHub Link

https://github.com/Kezia-Nathania/NYCU-Visual-Recognition-Spring-2025-HW2