# Object Detection with Faster R-CNN

My Git Link:https://github.com/IaTsai/DL-HW2

## Introduction

In this project, I tackle the digit detection and recognition problem using object detection. The goal is to accurately localize and classify digits within real-world images, which may include varying scales, occlusion, and lighting conditions. We adopt the Faster R-CNN framework due to its well-established performance on similar tasks.

## Method

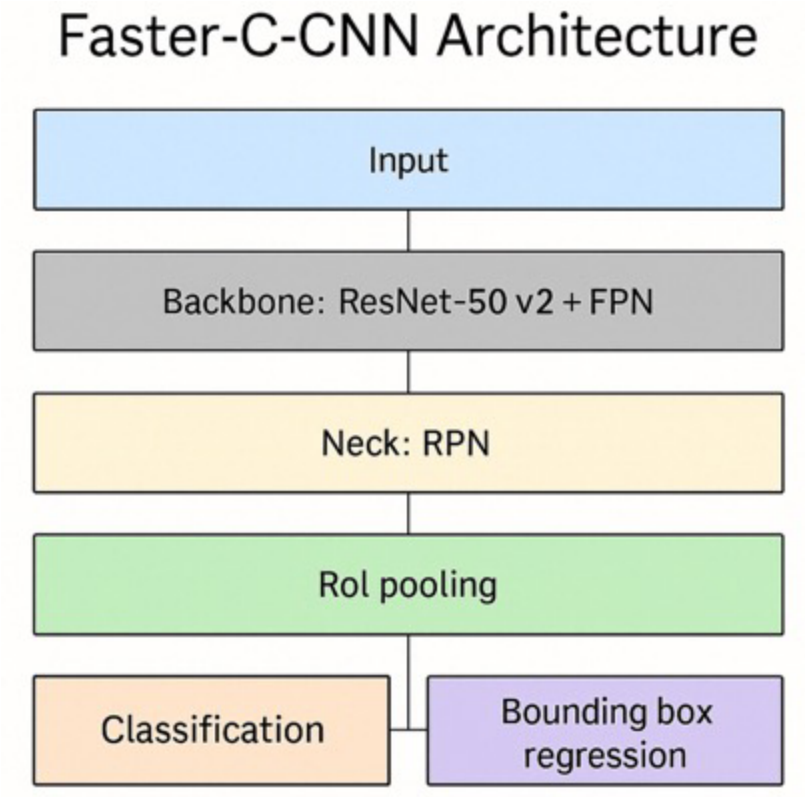
### Preprocessing

I used the provided COCO-format dataset and applied minimal preprocessing for our base experiments. The transform pipeline includes only ToTensor() for normalization. I later experimented with data augmentations such as ColorJitter, RandomHorizontalFlip, and Rotation.

### Model Architecture

I adopted **Faster R-CNN** as our main object detection framework, which includes:
Faster-C-CNN Architecture

- **Backbone**: ResNet-50 v2 with Feature Pyramid Network (FPN)
- **Neck (RPN)**: Region Proposal Network generates anchors at multiple scales and aspect ratios across the FPN outputs
- **Head**: A RoI pooling layer followed by two branches: classification (for 11 classes: 0~9 + background) and bounding box regression.




I also tested the following alternative backbones:

- ResNet101 + FPN
- ResNeXt50_32x4d + FPN
- ResNet50V2 + FPN + StepLR (Scheduler)
- ResNet50V2 + FPN + Data Augmentation
- ResNet50V2 + FPN + Anchor Resize
- ResNet50V2 + FPN + Freeze lower layers

All classification heads were replaced using:

```
in_features = model.roi_heads.box_predictor.cls_score.in_features
model.roi_heads.box_predictor = torchvision.models.detection.faster_rcnn.FastR
```

I used SGD with momentum=0.9, learning rate=0.005, and a batch size of 4.

## Results

**Model Comparison (Epoch 15, Full Training)**

| Model Version | Task1 mAP | Task2 Acc |
|---|---|---|
| ResNet50V2 + FPN | 0.37 | 0.77 |
| ResNet50V2 + FPN + StepLR | **0.39** | **0.84** |
| ResNet101 + FPN | 0.38 | 0.77 |
| ResNeXt50_32x4d + FPN | 0.37 | 0.76 |

**Data Augmentation (Epoch 3, Subset Testing)**

| Augmentation Variant | mAP |
|---|---|
| V0 (Baseline) | 0.358 |
| V1 (+ColorJitter) | 0.344 |
| V2 (+Horizontal Flip) | 0.098 |
| V3 (+Rotation 5°) | 0.338 |

Anchor resizing also showed mixed results depending on evaluation timing:

- After training: mAP ↑
- Offline evaluation: mAP ↓

I later ensured consistency by fixing random seeds and score thresholds.

## Additional Experiments and Analysis

### Why ResNet50V2 + FPN + StepLR?

- ResNet50V2 offers a balance between performance and speed.
- StepLR helps stabilize training and improved mAP by +0.02 and accuracy +0.07.
- Deeper backbones (ResNet101, ResNeXt) did not yield significant gains within 15 epochs.

### Hypothesis

- Freezing shallow layers helps avoid overfitting on low-level features.
- Scheduler improves long-term convergence stability.
- Data augmentation might require longer training to be effective.

### Limitations

- Subset testing may not generalize fully to full training.
- mAP variance observed when anchor settings change.

## References


- [1] Ren et al., *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*, NIPS 2015.
- [2] Torchvision Models: https://pytorch.org/vision/stable/models.html
- [3] COCO Evaluation: https://cocodataset.org/#detection-eval