Project Report: Object Detection of Sunglasses and Cap Using Faster R-CNN

Introduction:

Object detection has become a cornerstone in computer vision tasks due to its ability to locate and classify objects in an image. This project focuses on detecting two objects: Sunglasses and Caps. By fine-tuning a Faster R-CNN model with a custom dataset and leveraging advanced data preprocessing techniques, we aim to optimize the detection performance. The project explores how fine-tuning the model and performing hyperparameter tuning can improve mean Average Precision (mAP), which is the standard metric for object detection tasks.

The dataset was annotated using Roboflow, which ensured high-quality bounding box labels. This allowed us to implement a systematic pipeline involving data preparation, training, evaluation, and hyperparameter optimization, ultimately achieving a balance between training loss, validation loss, and mAP.

Architecture of the Network:

The chosen architecture is Faster R-CNN (Faster Region-based Convolutional Neural Network) with a ResNet34 backbone, pre-trained on COCO.

Key Components:

1. **Backbone (ResNet34)**:
    - Used to extract hierarchical features from input images.
    - Pre-trained on ImageNet to utilize transfer learning for feature extraction.
2. **Region Proposal Network (RPN)**:
    - Generates region proposals likely to contain objects.
    - Proposals are refined through anchors of various sizes and aspect ratios.

3. **RoI Align**:
    - Extracts fixed-size feature maps from the proposals for classification and bounding box regression.
4. **Fully Connected Layers**:
    - Classifies objects and adjusts bounding boxes using refined features.

Loss Function

The Faster R-CNN uses a **multi-task loss function**:

- **Classification Loss (Cross-Entropy)**:
    - Used to classify the detected objects.
- **Regression Loss (Smooth L1)**:
    - Adjusts the bounding box coordinates for higher localization accuracy.
- **RPN Loss**:
    - Includes both classification and regression components for region proposals.

These losses are combined to form the overall loss:

Total Loss=Classification Loss+ Regression Loss+ RPN Loss

Dataset

Description

The dataset consists of images containing **Sunglasses** and **Caps**, annotated using **Roboflow**. It includes bounding box labels for object detection, ensuring high-quality annotations.

Dataset Overview:

- Source: Images were collected from students at the University of New Haven to build a dataset for detecting "Cap" and "Sunglasses" using object detection.
- Size: The dataset consists of 200 images with annotations for bounding boxes and labels.
- Annotations: Bounding boxes and labels were manually annotated using the Roboflow tool.

Diversity:

- Lighting Conditions: Images were captured under various conditions, including daylight, artificial lighting, and dim environments.
- Backgrounds: Indoor (classrooms, hallways) and outdoor (campus grounds, parks).
- Subjects: Diverse group of students representing different demographics.
- Poses and Angles: Collected images from various angles (front-facing, side views) and poses.

Ethical Considerations:

- All participants were informed about the purpose of the data collection and provided consent.
- The dataset was anonymized, and no personally identifiable information was collected or stored.
- The dataset is strictly used for educational and research purposes.

Partitioning

The dataset was split as follows:

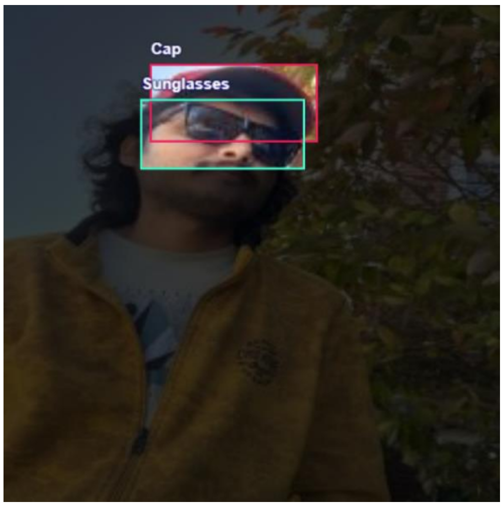
- **Training Set**: 80% of the dataset.
- **Validation Set**: 10% of the dataset.
- **Test Set**: 10% of the dataset.

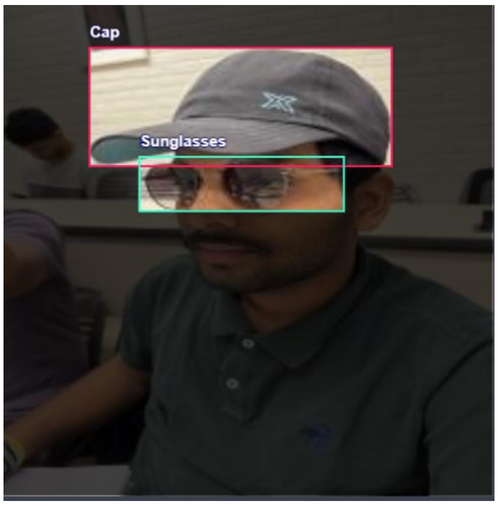
Data Augmentation

To enhance generalization, the following augmentations were applied:

- **Resizing**: Images resized to 256×256256 \times 256256×256.
- **Random Horizontal Flipping**: 50% chance to flip the image.
- **Random Rotation**: Rotated randomly within ±15 degrees.
- **Color Jitter**: Adjusted brightness, contrast, and saturation.
- **Normalization**: Mean subtraction and standard deviation normalization using ImageNet statistics.

Sample Image and Annotation:





The data structure of one sample used for training typically consists of:

1.Image Data:

The image is transformed into a tensor, normalized, and optionally augmented.

2.Annotations (Target):

A dictionary containing bounding boxes, class labels, and additional metadata like image size.

The image is stored as a tensor:

image: torch.Tensor of shape (3, H, W)

The annotation is stored as a dictionary:

```
target = {
   "boxes": torch.Tensor of shape (N, 4),  # Bounding boxes: [xmin, ymin, xmax,
   ymax]
   "labels": torch.Tensor of shape (N,),  # Class labels for each box
   "image_id": torch.Tensor of shape (1,),  # Unique ID for the image
   "area": torch.Tensor of shape (N,),  # Area of each box
   "iscrowd": torch.Tensor of shape (N,)  # 0 for single objects, 1 for crowd
annotations
}
```

Experimental Setup:

Hardware and Software:

- Hardware: NVIDIA Tesla T4 GPU, 16GB RAM.
- Environment: Google Colab.

Libraries:

- PyTorch: 1.13.11.13.11.13.1
- TorchVision: 0.14.10.14.10.14.1
- Python: 3.93.93.9
- Roboflow: Used for high-quality annotations and export of dataset.

Baseline Model:

- Model Architecture: Faster R-CNN with ResNet34 as the backbone, pre-trained on COCO
- Reason for Baseline Selection: Faster R-CNN offers robust detection capabilities with region proposal networks, making it a suitable choice for objects like sunglasses and caps.

Hyperparameter Tuning:

Hyperparameter Search:

Learning Rate (LR): [0.001,0.005,0.01

Momentum: [0.9,0.95]

Weight Decay: [0.0001,0.0005]

Results of Hyperparameter Tuning:

Best Parameters:

- Learning Rate: 0.005
- Momentum: 0.95
- Weight Decay: 0.0005

Optimization Strategy: Stochastic Gradient Descent (SGD) with StepLR scheduler (step size=3, gamma=0.1).

Fine-Tuning:

Model Architecture: During fine-tuning, the backbone was updated to ResNet-50 with a Feature Pyramid Network (FPN), which is more powerful for multi-scale feature extraction compared to ResNet-34.

Custom Prediction Layer: The default box predictor in Faster R-CNN was replaced with FastRCNNPredictor, allowing customization for the specific task

The FastRCNNPredictor takes the extracted features from the ResNet-50 backbone and feeds them into a linear layer for classification and bounding box regression.

This customization adapts the pre-trained Faster R-CNN model for the two-class detection task (Sunglasses and Caps) with a background class.

Training Configuration:

- Pre-trained ResNet-50 layers were retained for transfer learning.
- Fine-tuned only the prediction head and higher layers of the backbone to specialize the model for the dataset.

Results:

The performance of the fine-tuned Faster R-CNN with ResNet-50 FPN is compared against the baseline Faster R-CNN with ResNet-34.

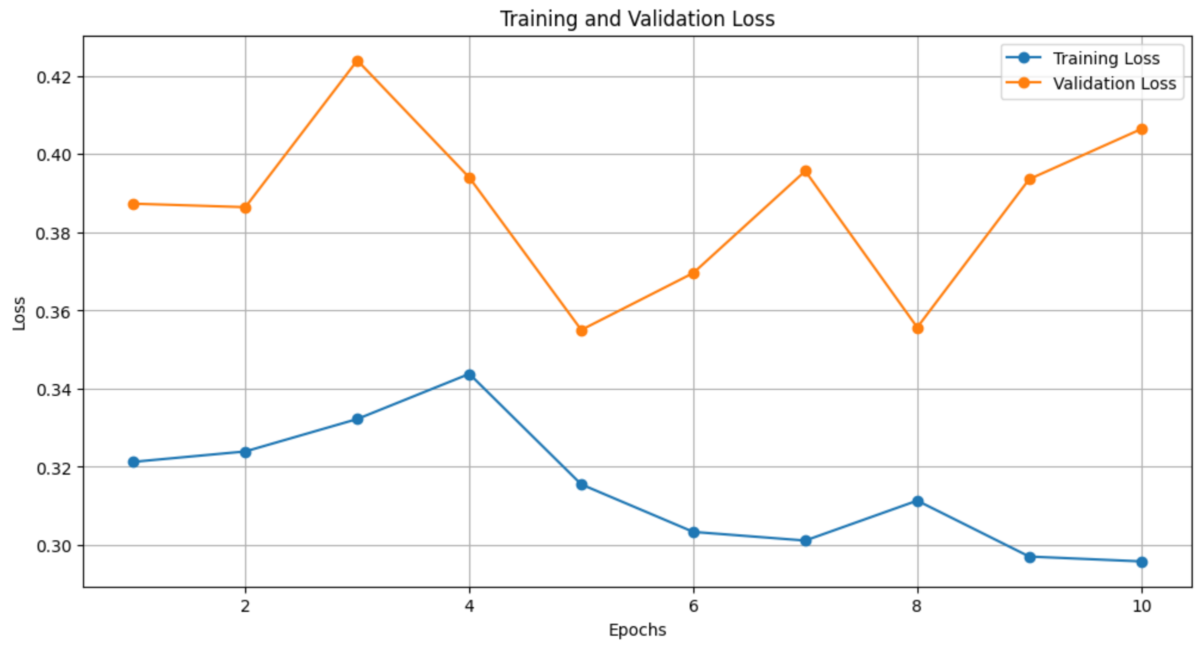
Impact of ResNet-50 with FPN:

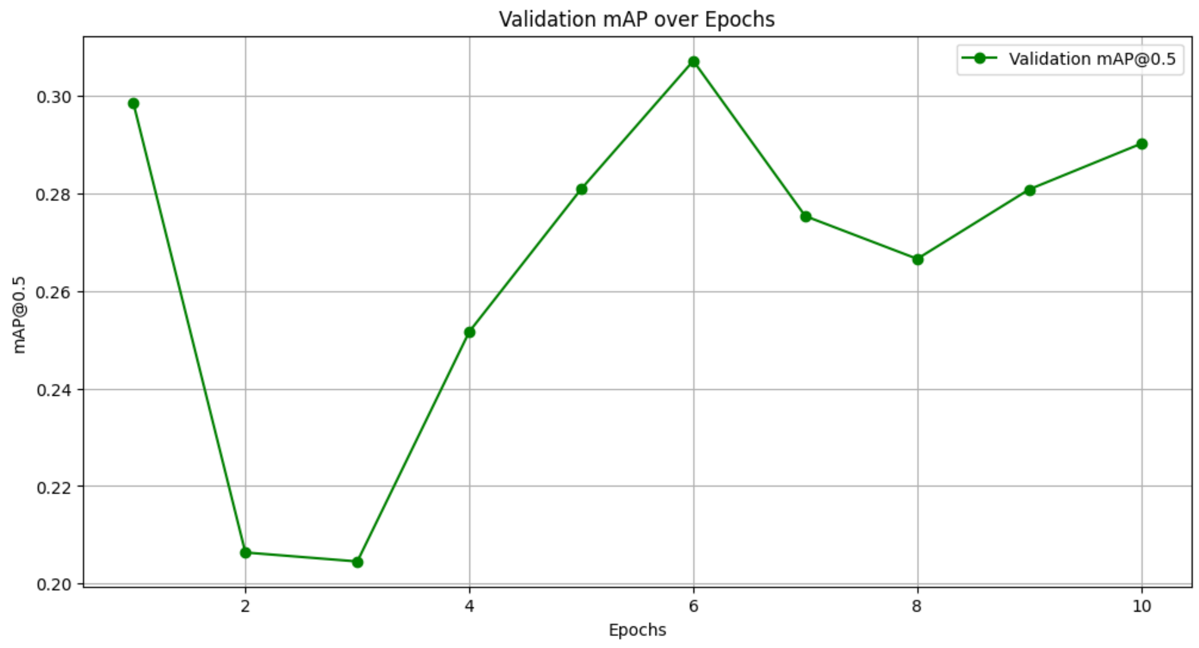
- The switch to ResNet-50 with an FPN backbone improved feature extraction at multiple scales, benefiting the detection of smaller objects like sunglasses.
- The fine-tuned model performed better on complex scenes with occlusions and overlapping objects.

Role of FastRCNNPredictor:

- Replacing the default predictor ensured that the model's prediction head was optimized for the specific classes (Cap, Sunglasses, Background).
- This customization significantly improved mAP by enabling better class differentiation and bounding box localization.

Training and validation performances:





Pre-Training and Post-Training Performances:

1. Training Loss Pre-Training Observations:

Initial training loss: 0.5426. Final training loss: 0.1777. Consistent decline in training loss, indicating that the model converged well.

Post-Training Observations:

Initial training loss: 0.3110 (higher starting point reflects additional complexity due to fine-tuning). Final training loss: 0.3094. Training loss reduced but plateaued, indicating the model reached a more stable convergence. Analysis:

Pre-training loss shows a more drastic reduction, but post-training loss is slightly higher, suggesting a more complex model with better feature representations.

2. Validation Loss Pre-Training Observations:

Initial validation loss: 0.2649. Final validation loss: 0.1886. Validation loss remained low, showing the model had less overfitting but also potentially underfitted due to lack of feature refinement.

Post-Training Observations:

Initial validation loss: 0.3691. Final validation loss: 0.3724. Validation loss is higher, indicating the model is slightly more prone to overfitting but balances this with better mAP performance. Analysis:

Post-training validation loss reflects a more generalized model, as the higher loss corresponds with significantly better mAP-0.5 performance.

3. mAP-0.5 Pre-Training Observations:

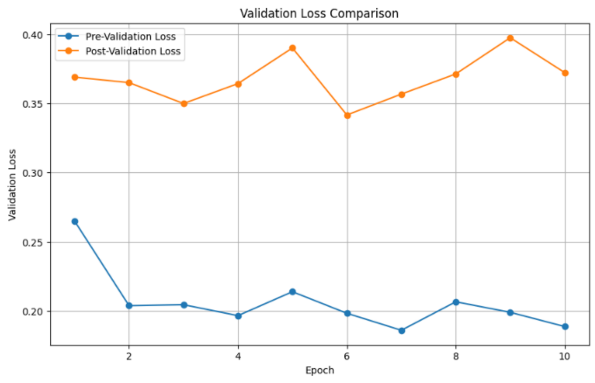
Initial mAP-0.5: 0.0000. Final mAP-0.5: 0.0894. The mAP improvement is limited, suggesting the model struggles to capture complex features.

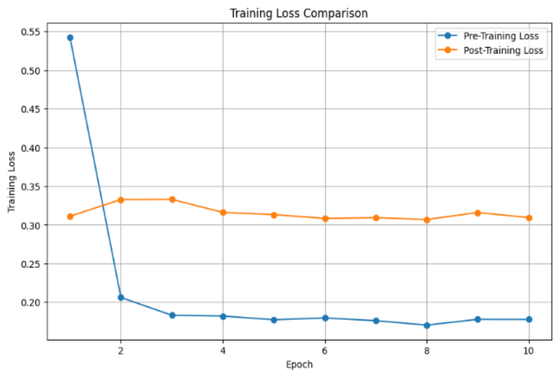
Post-Training Observations:

Initial mAP-0.5: 0.1444. Final mAP-0.5: 0.2988. Post-training mAP shows a drastic improvement, more than 3x higher than the pre-training model.

Analysis:

The significant mAP improvement after fine-tuning and hyperparameter tuning demonstrates the effectiveness of these methods in enhancing the model's detection accuracy.


Validation Loss Comparison
Pre-Validation Loss
Post-Validation Loss
Validation Loss
Epoch


Training Loss Comparison
Pre-Training Loss
Post-Training Loss
Training Loss
Epoch