CS 771 HW3

Arshad Kazi, Bhuyashi Deka, Sadman Sakib

November 2024

1 Understanding FCOS

1) What is the output of the backbone (i.e., the input to FPN)?
The output of backbone are the features maps from the last 3 convolutional blocks.

2) How levels are there in the FPN output? And how does the FPN generates these output feature maps?
There are a total of 5 levels of output from FPN. FPN takes 3 levels of feature maps from Resnet backbone. The main idea of FPN is to make the high resolution features aware of the low resolution features. The high resolution features contain local features while the low resolution features have a higher receptive field thus containing global features. The FPN takes these deep features and up samples and adds to high resolution features followed by convolution to adjust the number of channels. It generates 2 more feature maps by further convolving it in order to obtain more deeper features.

3) How does FCOS assign positive / negative samples during training? What are the loss functions used in the training?
FCOS has 3 outputs for each FPN head, classification, regression and centerness. The classificaton head classifies every pixel of feature to the class it belongs to. Classification loss is calculated for every pixel across all the heads. The pixels with background class (0 class) is treated as negative samples and remaining pixels are considered as positive samples (that belong to any class apart from background)

4) How does FCOS decode objects at inference? What are the necessary post-processing steps?
FCOS first calculates the objective score for pixel by multiplying classification score with centerness score followed by a square root. Then it finds those pixels that are beyond a certain threshold. Based on this, it selects the topK boxes from from the regression head which has the highest objective score. These selected boxes are then decoded into boxes by calculating x1, y1, x2, y2 using the l,t,r,b. The points are scaled up with stride and clamped within the image boundaries. The outputs from all the levels of the pyramid are aggregated

together. A class wise non max suppression is applied to these boxes. Finally anther threshold is applied to keep the total number of boxes in the image within a threshold.

2 Model Inference

The forward function in the classification head calculates the score for each class at every point from the FPN features. Similarly, regression head outputs the (l, t, r, b)-tuple and centerness value for each point in the FPN features. From these outputs, the predicted values are decoded as answered in question 4 above.

2.1 Pretrained Inference

- **Time Taken for inference**: 3.5 mins (2GB NVIDIA Persistance GPU. Batchsize = 4)
- **mAP @ 50% of TestData:** = 0.603

Figure 1 shows some sample detections with the pretrained model.

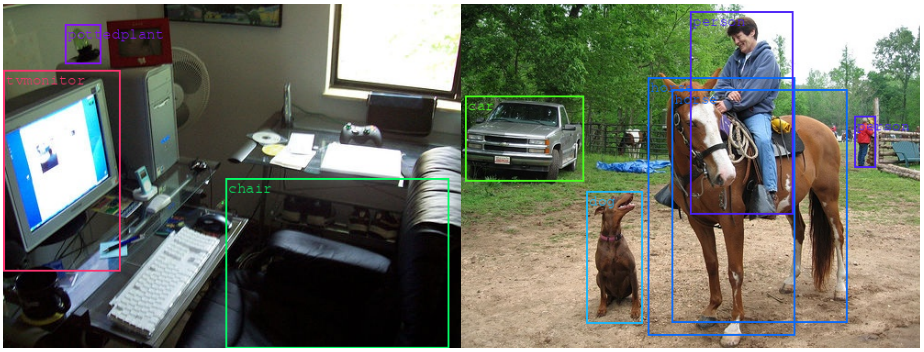

Figure 1: Sample detections from the pretrained FCOS model

3 Model Training

The forward pass of the training involves forward pass through the the model followed by the loss computation. The loss computation involves 2 major steps, (1) Ground truth generation and loss computation.

1) Ground Truth Generation: Ground truth is generated for all pyramid levels for classification, regression, and centerness. The boxes are assigned to different pyramid levels based on their size (regression range). To improve efficiency, we first compute the ground truth at the image-level resolution for all three outputs: classification, regression, and centerness. For classification ground truth, we iterate over the boxes in decreasing order of size and assign

a label to each pixel near the center of a box based on the box's label. Similarly, regression and centerness values are calculated for each box. Finally, using indexing, we calculate the ground truth for each pyramid level.

2) Loss Computation: The classification loss is computed for all pixels across all pyramid levels, using focal loss. The regression and centerness losses are calculated only for pixels where the classification ground truth is non-zero. For regression, the model employs the GIoU loss, while for centerness, BCE cross-entropy is used. All losses are normalized by the number of positive samples. The ground truth points are downsampled using stride as the downsampling factor.

4 Results

4.1 ResNet18 Training

- **Time Taken for Training**: 1hr 5 mins (2GB NVIDIA Persistance GPU. Batchsize = 4)
- **mAP @ 50% of TestData:** = 0.574

Figure 2 shows the loss curve of FCOS training with Resnet18 backbone.

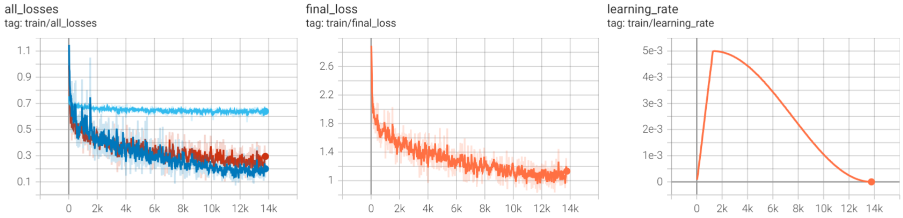

Figure 2: Loss curves for FCOS training with ResNet18 backbone

Figure 3 shows some sample detections with the trained model.

4.2 Resnet34 Training

- **Time Taken for Training**: 1hr 8 mins (2GB NVIDIA Persistance GPU. Batchsize = 4)
- **mAP @ 50% of TestData:** = 0.61

Figure 4 shows the loss curve of FCOS training with Resnet34 backbone. Figure 5 shows some sample detections with the trained model.

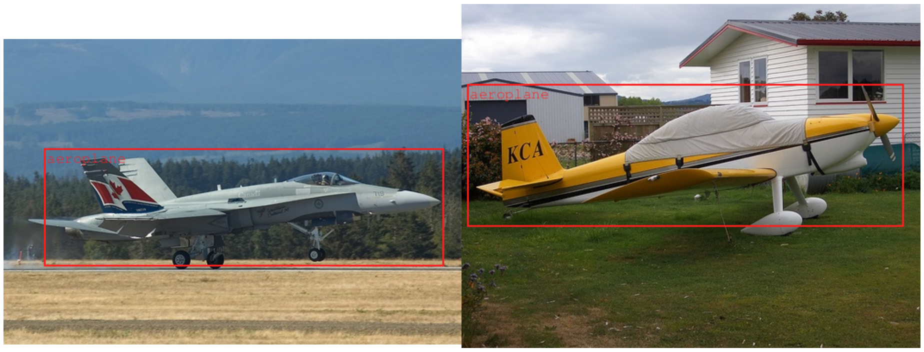

Figure 3: Sample detections from Trained FCOS model with ResNet18 backbone

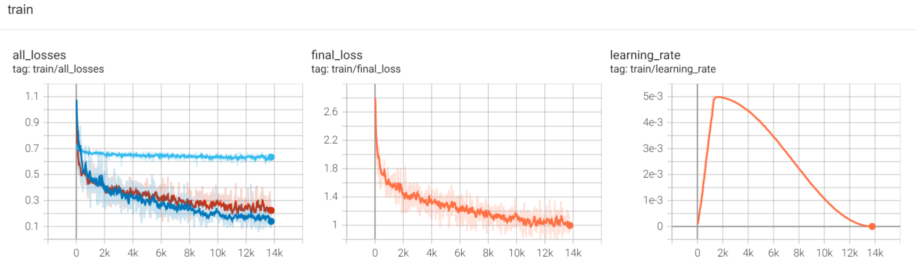

Figure 4: Loss curves for FCOS training with ResNet34 backbone

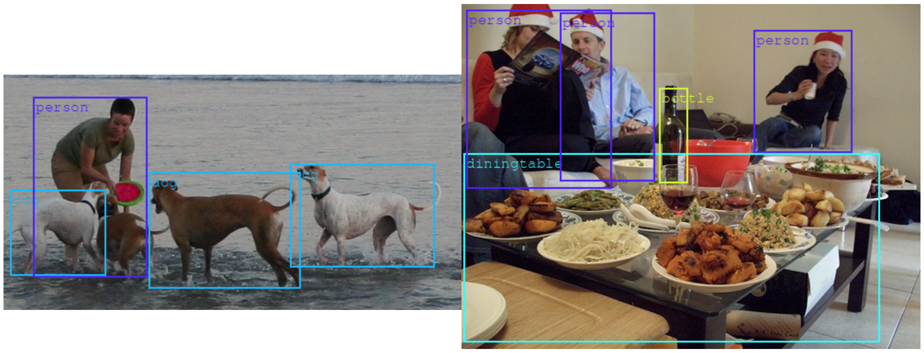

Figure 5: Sample detections from Trained FCOS model with ResNet34 backbone