2019/12/19
0780828
Alisher Mukashev (穆安里)

HOMEWORK #4

Introduction

There are various techniques that are used in computer vision tasks. **Instance segmentation** includes identification of boundaries of the objects at the detailed pixel level. For example, in the image below, there are 7 balloons at certain locations, and these are the pixels that belong to each one of the balloons.

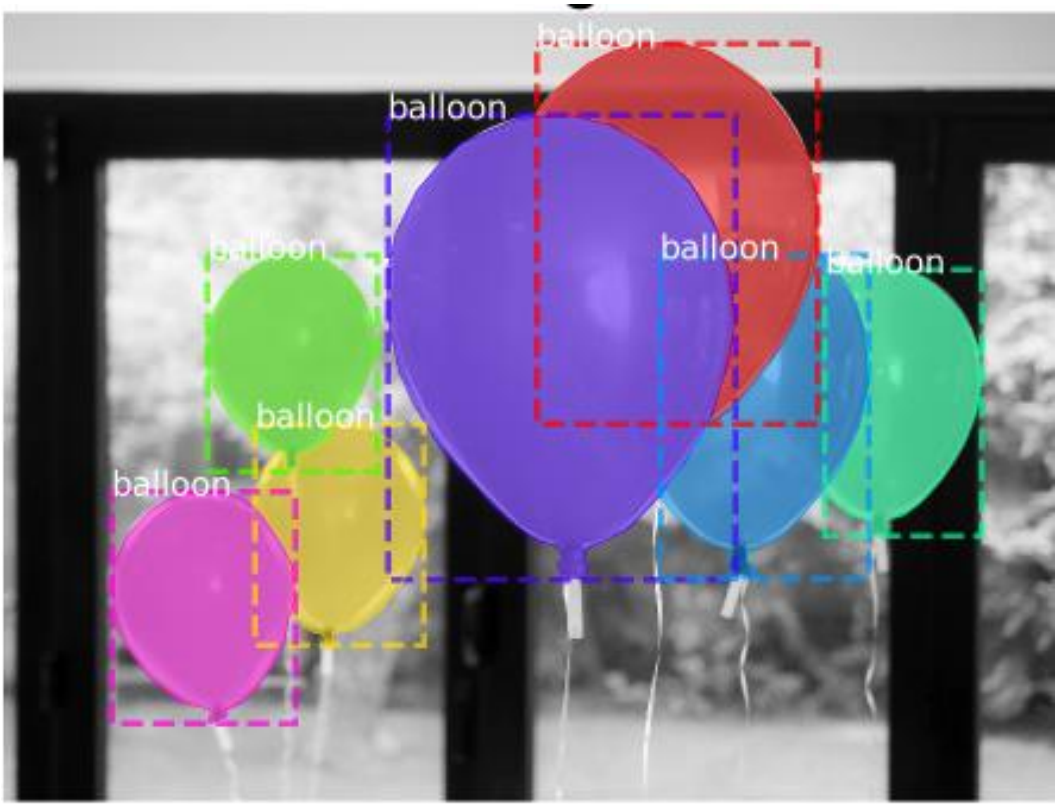

Figure 1: Image Segmentation

In this homework, I have developed a neural network for instance segmentation. Given Tiny VOC dataset contains 1349 training images and 100 test images. Example of a train image is shown in Figure 2:

Figure 2: Train image

I used Detectron2 for instance segmentation. Detecron2 originates from mask R-CNN [1]. The code based on Github repository [2].

Methodology

Preprocessing

To train the model we should use image and annotation data. The annotation data represented in COCO Annotation Format.

Model Architecture

In this homework, I am going to use mask R-CNN with ImageNet pretrained weights. Architecture of mask R-CNN is shown in Figure 3.

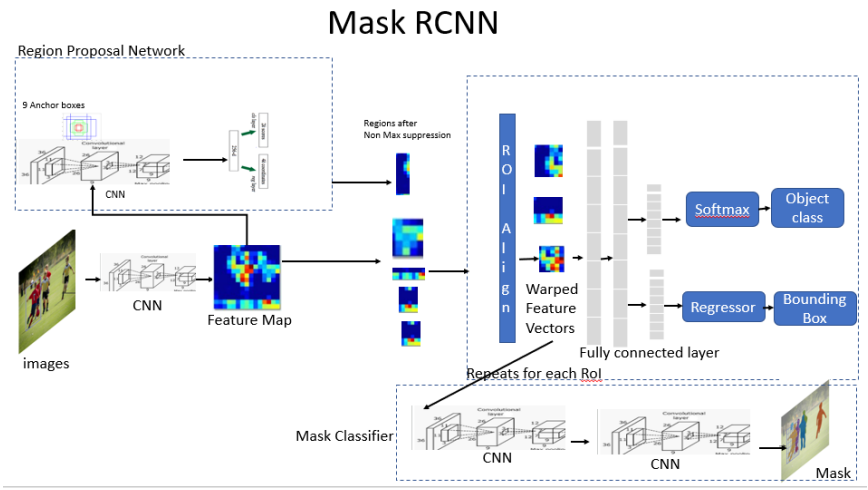

Figure 3: Mask R-CNN network architecture

Mask R-CNN is an instance segmentation technique which locates each pixel of every object in the image instead of the bounding boxes. It extends Faster R-CNN to pixel-level image segmentation. Mask R-CNN has two stages: region proposals and then classifying the proposals and generating bounding boxes and masks. It does so by using an additional fully convolutional network on top of a CNN based feature map with input as feature map and gives matrix with 1 on all locations where the pixel belongs to the object and 0 elsewhere as the output.

Hyperparameters

The learning rate 0.00025 was selected. The batchsize 2 was chosen. The number of iteration 30000 was choosen.

```
cfg = get_cfg()
cfg.merge_from_file("./detectron2_repo/configs/COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml")
cfg.DATASETS.TRAIN = ("VOC",)
cfg.DATASETS.TEST = ()
cfg.DATALOADER.NUM_WORKERS = 2
cfg.SOLVER.IMS_PER_BATCH = 2
cfg.SOLVER.BASE_LR = 0.00025
cfg.SOLVER.WEIGHT_DECAY_BIAS = 0.01
cfg.SOLVER.CHECKPOINT_PERIOD = 3000
cfg.SOLVER.MAX_ITER = 30000
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 128
cfg.MODEL.ROI_HEADS.NUM_CLASSES = 20
```

Figure 4: Hyperparameters

Results

The program was written in Google Colab and was inferenced by TA of this class. The mean Average Precision (mAP) was used to evaluate the performance of obtained model.

Figure 5: Accuracy evaluation; map 0.47416

Use of regularization technique as weight decay helped to get better results. My Github code link [3].

Related work implementation

I've tried to base my code on different repositories. But all of them originate mask R-CNN. I also changed a weight decay to avoid overfitting. Thus I got these results:

Figure 6: Mask R-CNN with ResNet50 backbone

without weight decay, mAp = 0.42467

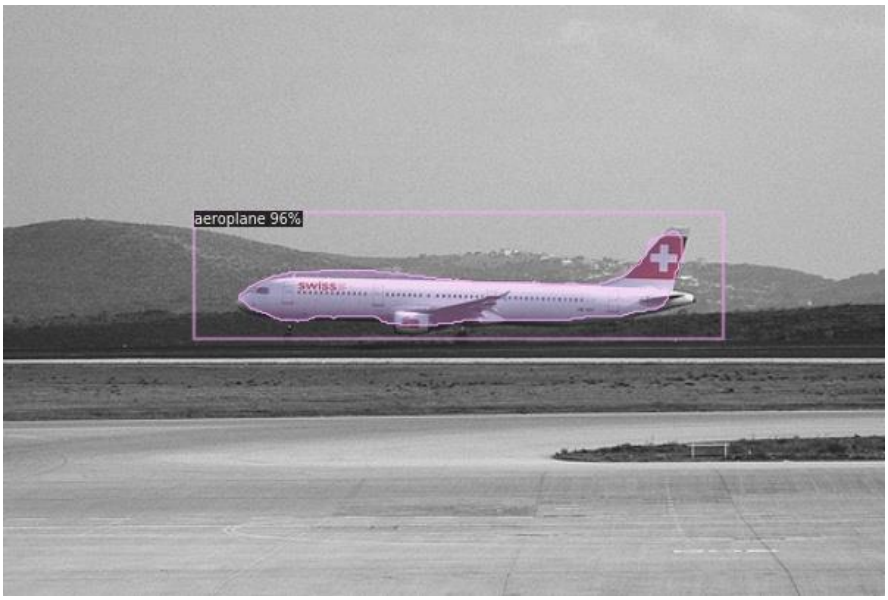

Figure 7: Mask R-CNN with ResNet50 backbone

weight decay = 0.1, mAp = 0.47416

I wanted to change the pretrained weights but I didn't have enough time for it. Instead of ResNet50 I've used ResNeXt-101-32x8d model trained with Caffe2 at FB but because the model is too large it took very long time. Anyway I saw good potential in the loss results.

I've also tried to use YOLACT [4] but I didn't succeed in this.

Summary

I think that for this moment mask R-CNN and YOLACT are the most powerful instance segmentation methods. But if we are talking about real time instance segmentation it is better if we use YOLACT. In other hand, different modifications of mask R-CNN can give us higher accuracy.

I think that I got higher detection accuracy but less mAp in Figure 6 because of overfitting.

References

1. https://arxiv.org/abs/1703.06870
2. https://github.com/facebookresearch/detectron2
3. https://github.com/alishsuper/Selected-Topics-in-Visual-Recognition-using-Deep-Learning/tree/master/HW4
4. https://github.com/dbolya/yolact