Detecting Object Using Mask R-CNN on a custom dataset (Tiger)

Instructed by Prof. Delphine Maugars

Prepared By
Md Imtiaz Sultan
(PSRS)

December 31, 2022

1 Mask R-CNN

Mask R-CNN, sometimes known as Mask RCNN, is the most advanced Convolutional Neural Network (CNN) for instance segmentation, and image segmentation. Faster R-CNN, a region-based convolutional neural network, served as the foundation for Mask R-CNN. Mask R-CNN extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. It is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks, e.g., allowing us to estimate human poses in the same framework.

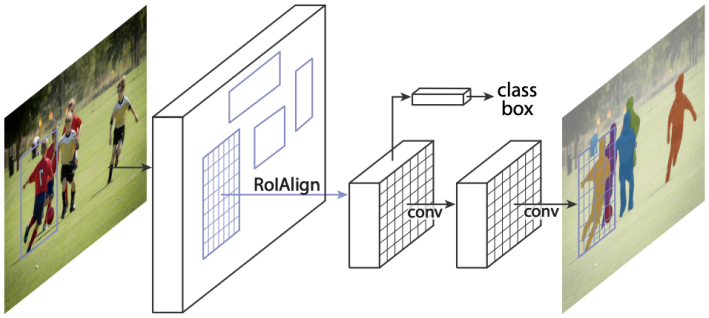

Figure 1: The Mask R-CNNframework

2 Working Principle of Mask R-CNN

The essential component of Fast/Faster R-CNN that is lacking from Mask R-CNN is the pixel-to-pixel alignment. The same two-step method, with an identical first stage, is used by Mask R-CNN (which is RPN). In the second stage, Mask R-CNN additionally produces a binary mask for each RoI in addition to class and box offset predictions. In contrast, the majority of modern systems rely on mask predictions for classification. Furthermore, the Faster R-CNN framework, which supports a large variety of customizable architecture designs, makes it easy to develop and train Mask R-CNN.

3 Working Procedures

3.1 Preparing Dataset

I used Tiger photos as a custom dataset for this project. The dataset I utilized was produced by VGG Image Annotator (VIA). The coordinates of every polygon I've drawn on my photographs are included in the annotation file, which is in.json format. This is how the.json file will appear:

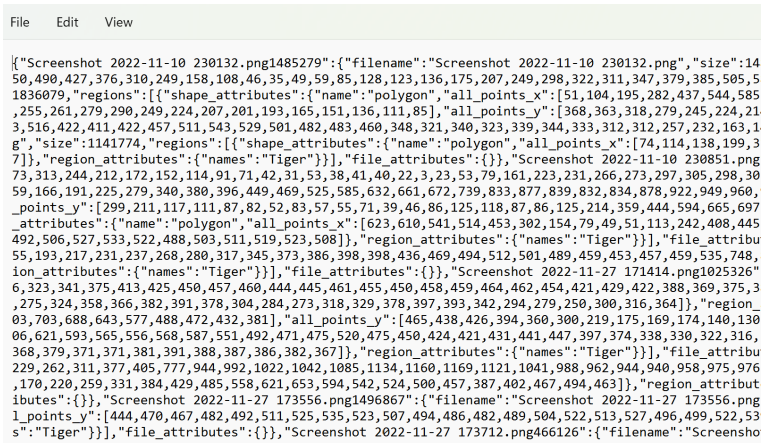

Figure 2: The .json files

3.2 Choosing the proper library versions & Hyperparameters

To implement the Mask RCNN architecture without getting any errors I used Tensorflow version 1.14.0 and Keras version 2.2.5. I used jupyter notebook to do the task on my Windows 11 device.

The number of classes is set to be total classes + 1 (for background). The default number of steps per epoch is 10, but we can raise it if we have access to more powerful computational resources. Since the detection threshold is 90%, all proposals with a confidence level of less than 0.9 will be disregarded.

Then I made a CustomDataset class instance for the training dataset. A second instance was made in a similar manner for the validation dataset. After that, I called the load custom() method and provided the directory name where our data is kept. Here 'layers' parameter is set to 'heads'. The model is running on 20 epochs and Starting at epoch 0 LR=0.001. After setting we start the training.

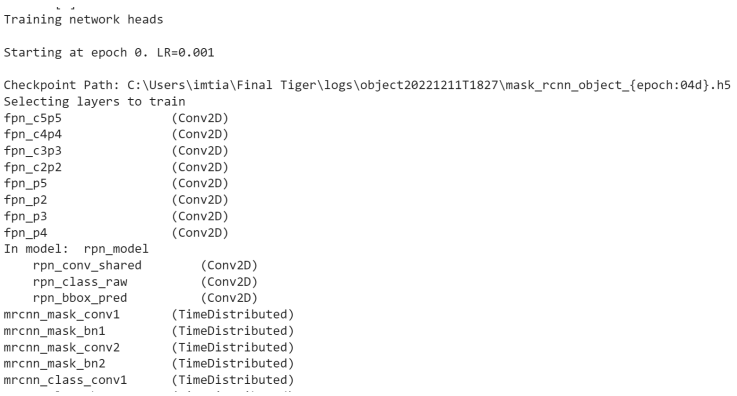

Figure 3: Traing the Model

4 Test

After the training process finishes we tested our trained model by uploading a photo of a tiger. And it detected the tiger properly with 99.8% accuracy.

Figure 4: Detecting Tiger

Later we randomly tested tiger images from our dataset and it also successfully detected the tiger with a very good percentage of accuracy.

Figure 5: Detecting Tiger randomly from the dataset

The project can be accessed from the following GitHub repository link: https://github.com/imtiaz-95/-imtiaz-95-Custom-Mask-R-CNN-Model-for-Detecting-Tiger.git