IMAGE SEGMENTATION WITH REGION-BASED CNN

Deep Learning Module

Maher SEBAI, DSTI A18

TABLE OF CONTENTS

ABSTRACT

After scoring big achievement in image classification tasks (where the content of the image is treated as a whole), Convolutional Neural Networks or CNNs where instrumented to tackle new computer vision challenges. Image segmentation is such a challenge where the content of the image is segmented to reveal its semantic components. More specifically, the goal of semantic image segmentation is to label each pixel of an image with a corresponding class of what is being represented. Because we're predicting for every pixel in the image.

Region-based convolutional neural networks or regions with CNN features (R-CNNs) are a pioneering approach that applies deep models to object detection. In this report, we will present on of its latest development called **Mask R-CNN** (developed by **Ross Girshick**) that greatly enhanced state of the art results. We will quickly present the family of Region-based CNNs the algorithm inherit from, than we present Mask R-CNN model along with its benchmark result. Finally we finish with **using it's Keras implementation in a custom code to infer semantic segments using a personal photo and a pre-trained Mask R-CNN model.**

INTRODUCING REGION-BASED CNN

Object detection is a computer vision task that involves both localizing one or more objects within an image and classifying each object in the image. It is a challenging computer vision task that requires both successful object localization in order to locate and draw a bounding box around each object in an image, and object classification to predict the correct class of object that was localized.

An extension of object detection involves marking the specific pixels in the image that belong to each detected object instead of using coarse bounding boxes during object localization. This harder version of the problem is generally referred to as object segmentation or semantic segmentation.

The Region-Based Convolutional Neural Network, or R-CNN, is a family of convolutional neural network models designed for object detection, developed by Ross Girshick. There are perhaps four main variations of the approach, resulting in the current pinnacle called Mask R-CNN. The salient aspects of each variation can be summarized as follows:

- ***R-CNN:*** *Bounding boxes are proposed by the "selective search" algorithm, each of which is stretched and features are extracted via a deep convolutional neural network, such as AlexNet, before a final set of object classifications are made with linear SVMs.*
- ***Fast R-CNN:*** *Simplified design with a single model, bounding boxes are still specified as input, but a region-of-interest pooling layer is used after the deep CNN to consolidate regions and the model predicts both class labels and regions of interest directly.*
- ***Faster R-CNN:*** *Addition of a Region Proposal Network that interprets features extracted from the deep CNN and learns to propose regions-of-interest directly.*
- ***Mask R-CNN:*** *Extension of Faster R-CNN that adds an output model for predicting a mask for each detected object.*

The Mask R-CNN model introduced in the 2018 paper titled "Mask R-CNN" is the most recent variation of the family models and supports both object detection and object segmentation. The paper provides a nice summary of the model linage to that point:

> *The Region-based CNN (R-CNN) approach to bounding-box object detection is to attend to a manageable number of candidate object regions and evaluate convolutional networks independently on each RoI. R-CNN was extended to allow attending to RoIs on feature maps using RoIPool, leading to fast speed and better accuracy. Faster R-CNN advanced this stream by learning the attention mechanism with a Region Proposal Network (RPN). Faster R-CNN is flexible and robust to many*

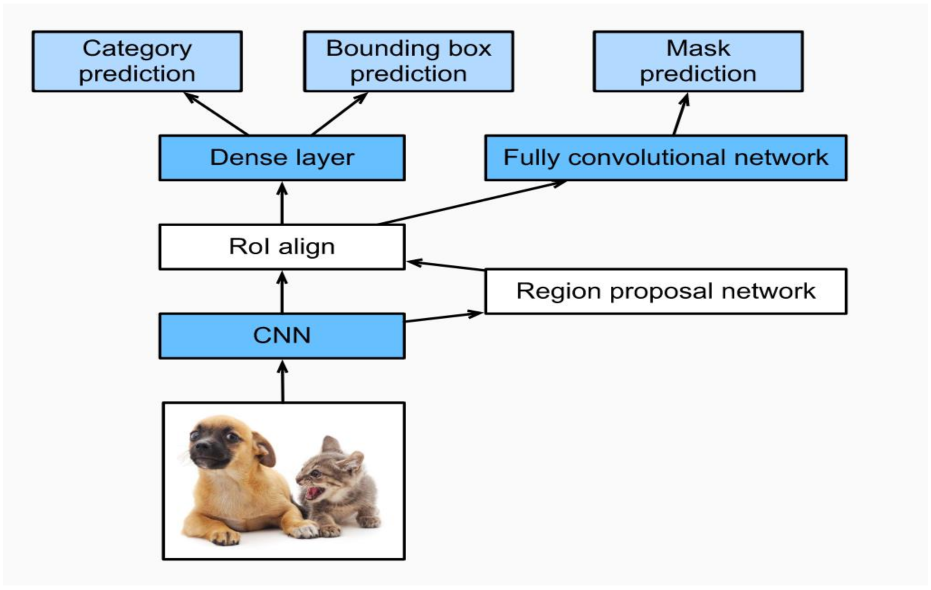
follow-up improvements, and is the current leading framework in several benchmarks.

— Mask R-CNN [https://arxiv.org/abs/1703.06870], 2018.

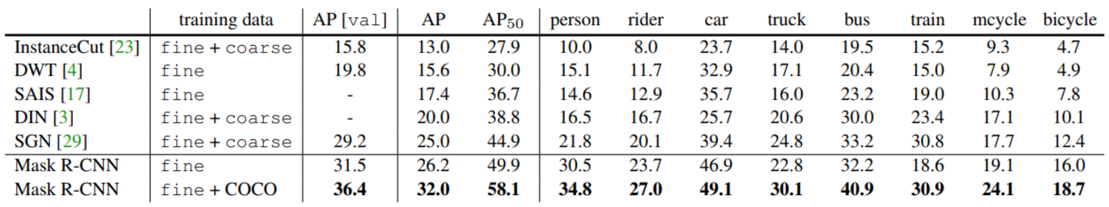
The family of methods may be among the most effective for object detection, achieving then state-of-the-art results on computer vision benchmark datasets. Although accurate, the models can be slow when making a prediction as compared to alternate models such as YOLO that may be less accurate but are designed for real-time prediction.

MASK R-CNN ARCHITECTURE AND BENCHMARK

Mask R-CNN is a modification to the Faster R-CNN model. Mask R-CNN models replace the RoI pooling layer with an RoI alignment layer. This allows the use of bilinear interpolation to retain spatial information on feature maps, making Mask R-CNN better suited for pixel-level predictions. The RoI alignment layer outputs feature maps of the same shape for all RoIs. This not only predicts the categories and bounding boxes of RoIs, but allows us to use an additional fully convolutional network to predict the pixel-level positions of objects.



The original paper reports the following state-of-the-art results on the famous **cityscapes dataset** (as of 2017, only recently outperformed by the newly released **Mask Scoring R-CNN**)



	training data	AP [`val`]	AP	AP_{50}	person	rider	car	truck	bus	train	mcycle	bicycle
InstanceCut [23]	`fine` + `coarse`	15.8	13.0	27.9	10.0	8.0	23.7	14.0	19.5	15.2	9.3	4.7
DWT [4]	`fine`	19.8	15.6	30.0	15.1	11.7	32.9	17.1	20.4	15.0	7.9	4.9
SAIS [17]	`fine`	-	17.4	36.7	14.6	12.9	35.7	16.0	23.2	19.0	10.3	7.8
DIN [3]	`fine` + `coarse`	-	20.0	38.8	16.5	16.7	25.7	20.6	30.0	23.4	17.1	10.1
SGN [29]	`fine` + `coarse`	29.2	25.0	44.9	21.8	20.1	39.4	24.8	33.2	30.8	17.7	12.4
Mask R-CNN	`fine`	31.5	26.2	49.9	30.5	23.7	46.9	22.8	32.2	18.6	19.1	16.0
Mask R-CNN	`fine` + COCO	**36.4**	**32.0**	**58.1**	**34.8**	**27.0**	**49.1**	**30.1**	**40.9**	**30.9**	**24.1**	**18.7**

Table 7. Results on Cityscapes `val` ('AP [`val`]' column) and `test` (remaining columns) sets. Our method uses ResNet-50-FPN.

PRACTICAL LAB: MASK R-CNN FOR INFERENCING

Detectron is Facebook AI Research's software system that implements state-of-the-art object detection algorithms, **including Mask R-CNN**. It is written in Python and powered by the Caffe2 deep learning framework. This is the original implementation of the paper authors.

Instead of the original implementation, we are going to use a **Tensorflow+Keras Third-party implementation** developed by **Matterport**.

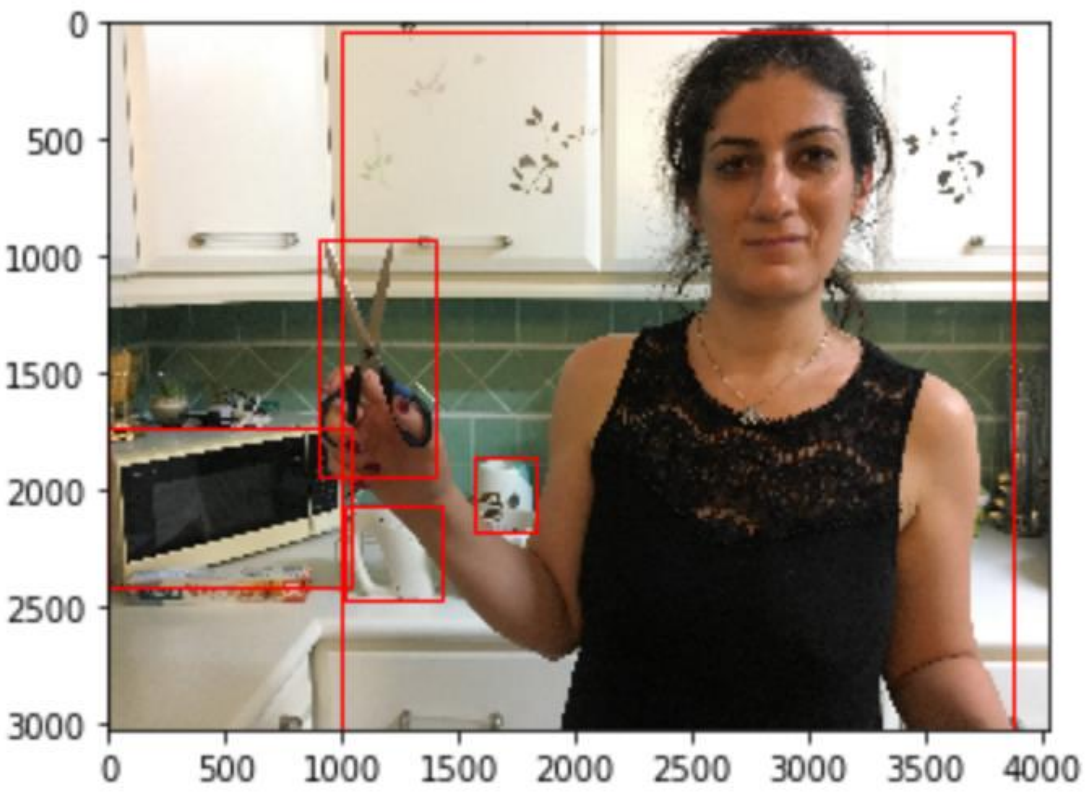
The used github source code was used to:

- *Install the model python package*
- *Download a pretrained Mask R-CNN model (named mask_rcnn_coco.h5 and around 250MB of size)*
- *Develop a small inference program using a personal photo (**my wife in the kitchen around various objects labelled in the MS COCO dataset**)*

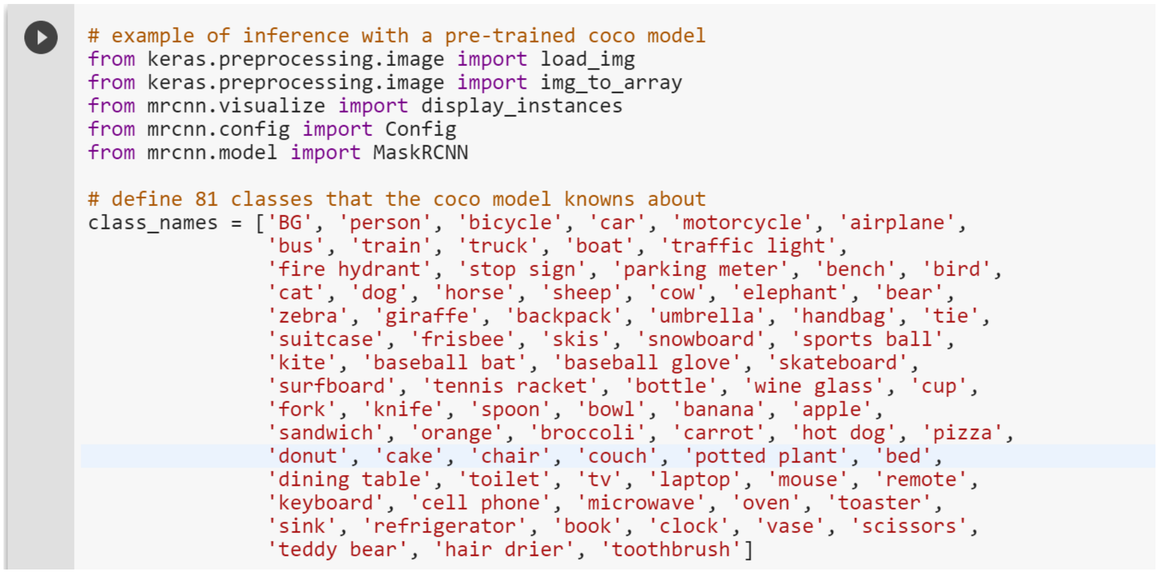
https://github.com/matterport/Mask_RCNN

Based on Colab Jupyter notebook, the first experiment is to produce bounding boxed around the identified objects.

```
[ ]  from keras.preprocessing.image import load_img
     from keras.preprocessing.image import img_to_array
     from mrcnn.config import Config
     from mrcnn.model import MaskRCNN
     from matplotlib import pyplot
     from matplotlib.patches import Rectangle

     # draw an image with detected objects
     def draw_image_with_boxes(filename, boxes_list):
          # load the image
          data = pyplot.imread(filename)
          # plot the image
          pyplot.imshow(data)
          # get the context for drawing boxes
          ax = pyplot.gca()
          # plot each box
          for box in boxes_list:
               # get coordinates
               y1, x1, y2, x2 = box
               # calculate width and height of the box
               width, height = x2 - x1, y2 - y1
               # create the shape
               rect = Rectangle((x1, y1), width, height, fill=False, color='red')
               # draw the box
               ax.add_patch(rect)
          # show the plot
          pyplot.show()

     # define the test configuration
     class TestConfig(Config):
          NAME = "test"
          GPU_COUNT = 1
          IMAGES_PER_GPU = 1
          NUM_CLASSES = 1 + 80

     # define the model
     rcnn = MaskRCNN(mode='inference', model_dir='./', config=TestConfig())
     # load coco model weights
     rcnn.load_weights('mask_rcnn_coco.h5', by_name=True)
     # load photograph
     img = load_img('IMG_1594.JPG')
     img = img_to_array(img)
     # make prediction
     results = rcnn.detect([img], verbose=0)
     # visualize the results
     draw_image_with_boxes('IMG_1594.JPG', results[0]['rois'])
```

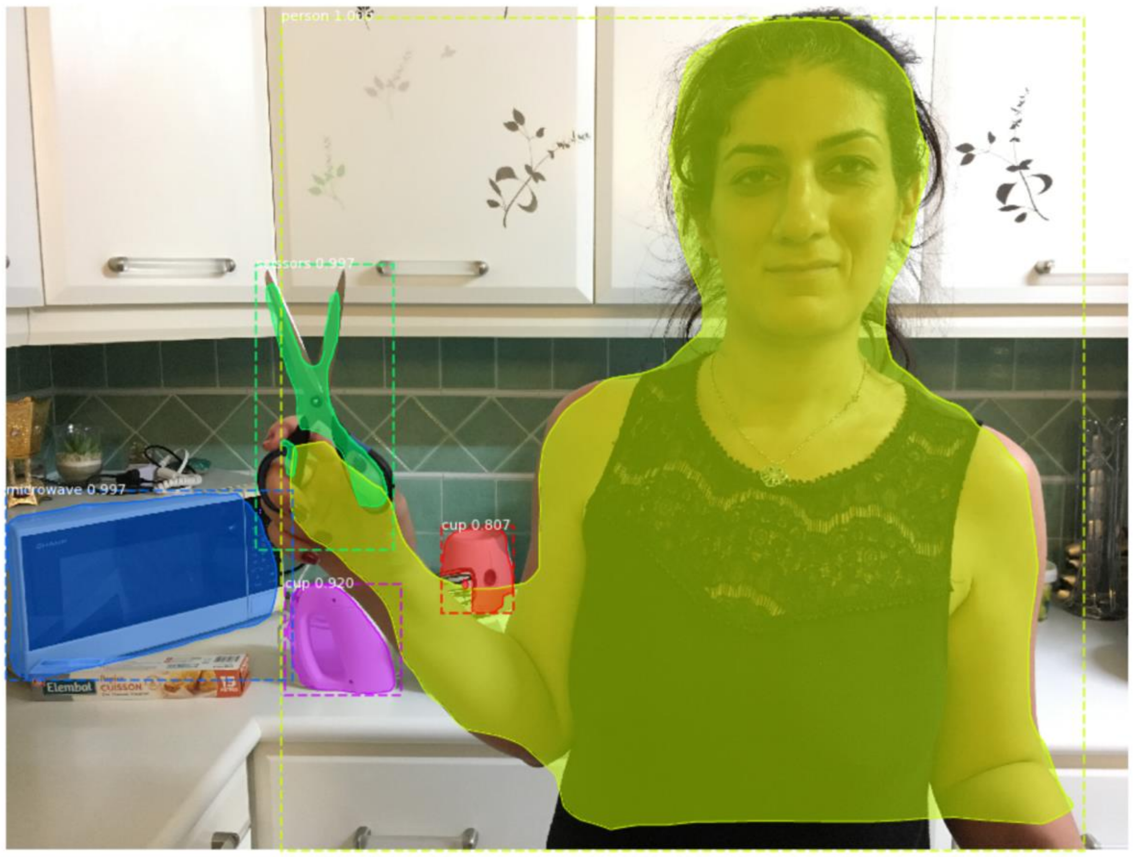


The model was able to identify Three RoIs (Region of Interest). The second experiment if to produce masks around the identified objects.



```
# example of inference with a pre-trained coco model
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from mrcnn.visualize import display_instances
from mrcnn.config import Config
from mrcnn.model import MaskRCNN

# define 81 classes that the coco model knowns about
class_names = ['BG', 'person', 'bicycle', 'car', 'motorcycle', 'airplane',
               'bus', 'train', 'truck', 'boat', 'traffic light',
               'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird',
               'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear',
               'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie',
               'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball',
               'kite', 'baseball bat', 'baseball glove', 'skateboard',
               'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup',
               'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple',
               'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza',
               'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed',
               'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote',
               'keyboard', 'cell phone', 'microwave', 'oven', 'toaster',
               'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors',
               'teddy bear', 'hair drier', 'toothbrush']
```

```
# define the test configuration
class TestConfig(Config):
     NAME = "test"
     GPU_COUNT = 1
     IMAGES_PER_GPU = 1
     NUM_CLASSES = 1 + 80

# define the model
rcnn = MaskRCNN(mode='inference', model_dir='./', config=TestConfig())
# load coco model weights
rcnn.load_weights('mask_rcnn_coco.h5', by_name=True)
# load photograph
img = load_img('IMG_1594.JPG')
img = img_to_array(img)
# make prediction
results = rcnn.detect([img], verbose=0)
# get dictionary for first prediction
r = results[0]
# show photo with bounding boxes, masks, class labels and scores
display_instances(img, r['rois'], r['masks'], r['class_ids'], class_names, r['scores'])
```



We now have a relatively precise mask surrounding the identified object along with its class and its probability (for instance it correctly identified the microwave with 0.997 probability, scissors with 0.997, but mistakenly labeled two kitchen applieces as being cups with respectively 0.92 and 0.807 probabilities)

SUMMARY

In this report we presented recent state of the art semantic segmentation deep learning method: Mask R-CNN. We first presented its lineage based on the family of Region based CNN. Then we introduced its architecture and reported results based on cityscapes dataset.

We finally used Matterport's pretrained Keras implementation and used it for semantic segmentation inferencing using a personal photo.