MASK RCNN

> **Introduction** Mask RCNN efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each in- stance. Mask RCNN i an extended version of Faster RCNN with aditional veature of predicting a box parallel to the existing bounding Box. Mask R-CNN 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. Top results in all three tracks of the COCO suite of challenges, including instance segmentation, bounding-box object detection, and person keypoint detection are shown. Without tricks, Mask R-CNN outperforms all existing, single-model entries on every task, including the COCO 2016 challenge winners.

Installing appropriate versions of Tensorflow and Keras to make RCNN work properly

```
!pip uninstall tensorflow -y
!pip uninstall keras -y
!pip install tensorflow-gpu==1.13.1
!pip install keras==2.0.8
!pip install h5py==2.10.0 --force-reinstall
```

```
Found existing installation: tensorflow 1.13.1
Uninstalling tensorflow-1.13.1:
  Successfully uninstalled tensorflow-1.13.1
Found existing installation: Keras 2.0.8
Uninstalling Keras-2.0.8:
  Successfully uninstalled Keras-2.0.8
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting tensorflow-gpu==1.13.1
  Downloading tensorflow_gpu-1.13.1-cp37-cp37m-manylinux1_x86_64.whl (345.0 MB)
     |████████████████████████████████| 345.0 MB 3.1 kB/s
Requirement already satisfied: astor>=0.6.0 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: keras-applications>=1.0.6 in /usr/local/lib/python3.7/di
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: six>=1.10.0 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: keras-preprocessing>=1.0.5 in /usr/local/lib/python3.7/d
Requirement already satisfied: grpcio>=1.8.6 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: tensorboard<1.14.0,>=1.13.0 in /usr/local/lib/python3.7/
Requirement already satisfied: tensorflow-estimator<1.14.0rc0,>=1.13.0 in /usr/local/li
Requirement already satisfied: protobuf>=3.6.1 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: absl-py>=0.1.6 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: gast>=0.2.0 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: h5py in /usr/local/lib/python3.7/dist-packages (from ker
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dis
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: mock>=2.0.0 in /usr/local/lib/python3.7/dist-packages (f
Installing collected packages: tensorflow-gpu
Successfully installed tensorflow-gpu-1.13.1
WARNING: The following packages were previously imported in this runtime:
  [tensorflow]
You must restart the runtime in order to use newly installed versions.
```

RESTART RUNTIME

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting keras==2.0.8
  Using cached Keras-2.0.8-py2.py3-none-any.whl (276 kB)
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: pyyaml in /usr/local/lib/python3.7/dist-packages (from k
Requirement already satisfied: scipy>=0.14 in /usr/local/lib/python3.7/dist-packages (f
Installing collected packages: keras
Successfully installed keras-2.0.8
WARNING: The following packages were previously imported in this runtime:
  [keras]
You must restart the runtime in order to use newly installed versions.
```

RESTART RUNTIME

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting h5py==2.10.0
  Using cached h5py-2.10.0-cp37-cp37m-manylinux1_x86_64.whl (2.9 MB)
Collecting six
  Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
```

```
  Collecting numpy>=1.7
    Using cached numpy-1.21.6-cp37-cp37m-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (
  Installing collected packages: six, numpy, h5py
    Attempting uninstall: six
```

I have taken Help from github to run Mask RCNN. here the github directory is cloned.

```
      Successfully uninstalled six-1.16.0
```

```
!git clone --quiet https://github.com/matterport/Mask_RCNN.git
```

```
  fatal: destination path 'Mask_RCNN' already exists and is not an empty directory.
```

```
    Attempting uninstall: h5py
```

```
import os
import sys
```

```
  ERROR: pip's dependency resolver does not currently take into account all the packages
```

```
!pwd
```

```
  /content
```

```
  Successfully installed h5py-2.10.0 numpy-1.21.6 six-1.16.0
```

pwd module provides access to the Unix user account and password database. Each entry stored in Unix user account and password database is reported as a tuple-like object whose attributes are similar as the members of passwd structure defined in pwd. It is used here to get the content from github.

Getting Root Directory for the model

```
# Root directory of the project
ROOT_DIR = os.path.abspath("/content/Mask_RCNN")
sys.path.append(ROOT_DIR)
```

Downloading the MASk RCNN model from github

```
!wget https://github.com/matterport/Mask_RCNN/releases/download/v2.0/mask_rcnn_coco.h5
```

```
  --2022-06-17 20:19:11--  https://github.com/matterport/Mask_RCNN/releases/download/v2.0
  Resolving github.com (github.com)... 140.82.112.4
  Connecting to github.com (github.com)|140.82.112.4|:443... connected.
  HTTP request sent, awaiting response... 302 Found
  Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/
  --2022-06-17 20:19:11--  https://objects.githubusercontent.com/github-production-releas
  Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.
  Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108
  HTTP request sent, awaiting response... 200 OK
  Length: 257557808 (246M) [application/octet-stream]
  Saving to: ‘mask_rcnn_coco.h5.1’
```

```
mask_rcnn_coco.h5.1 100%[===================>] 245.63M  4.65MB/s    in 24s

2022-06-17 20:19:36 (10.0 MB/s) - ‘mask_rcnn_coco.h5.1’ saved [257557808/257557808]
```

Coco dataset Details

Dataset consists of:

- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints
- Recognition in context
- Object segmentation As we want to implement the model on Coco Dataset so saving the dataset in directory.

```
sys.path.append(os.path.join(ROOT_DIR, "samples/coco/"))  # To find local version
import coco
config = coco.CocoConfig()
COCO_DIR = "/content/Mask_RCNN/Coco_Data"
```

MASK RCNN Faster R-CNN has two outputs for each candidate object, a class label and a bounding-box offset; to this there is addition of a third branch that outputs the object mask.

```
dataset = coco.CocoDataset()
dataset.load_coco(COCO_DIR, "minival", 2014, auto_download=True)
dataset.prepare()
```

```
Will use images in /content/Mask_RCNN/Coco_Data/val2014
Will use annotations in /content/Mask_RCNN/Coco_Data/annotations/instances_minival2014.
loading annotations into memory...
Done (t=0.68s)
creating index...
index created!
```

Parts of Mask-RCNN These are some parts of Mask RCNN (i) theconvolutional backbone architecture used for feature extraction over an entire image, and (ii) the network head for bounding-box recognition (classification and regression) and mask prediction that is applied separately to each RoI.

ResNet and ResNeXt networks of depth 50 or 101 layers are evaluated. This backbone with ResNet-50, for example, is denoted by ResNet-50-C4. This is a common choice used. For the network architectures of previous work is used with additional fully convolutional mask prediction branch. Specifically the Faster R-CNN box heads from the ResNet is extended. the head on the ResNet-C4 backbone includes the 5-th stage of ResNet (namely, the 9-layer 'res5'), which is compute intensive. For FPN, the backbone already includes res5 and thus allows for a more efficient head that uses fewer filters

```
sys.path.append(os.path.join(ROOT_DIR, "mrnn"))
```

Implementation Details: As in Fast R-CNN, an RoI is considered positive if it has IoU with a ground-truth box of at least 0.5 and negative otherwise. The mask loss Lmask is defined only on positive RoIs. The mask target is the intersection between an RoI and its associated ground-truth mask. Images are resized such that their scale (shorter edge) is 800 pixels. Each mini-batch has 2 images per GPU and each image has N sampled RoIs, with a ratio of 1:3 of positive to negatives. N is 64 for the C4 backbone and 512 for FPN. We train on 8 GPUs (so effective minibatch size is 16) for 160k iterations, with a learning rate of 0.02 which is decreased by 10 at the 120k iteration. We use a weight decay of 0.0001 and a momentum of 0.9. The RPN anchors span 5 scales and 3 aspect ratios. For convenient ablation, RPN is trained separately and does not share features with Mask R-CNN, unless specified. For every entry in this paper, RPN and Mask R-CNN have the same backbones and so they are shareable.

```
)/coco.py evaluate --dataset="/content/Mask_RCNN/Coco_Data" --model="/content/mask_rcnn_coco.
```

```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/framework/dtypes.py:526: Fu
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/usr/local/lib/python3.7/dist-packages/tensorflow/python/framework/dtypes.py:527: Fu
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/usr/local/lib/python3.7/dist-packages/tensorflow/python/framework/dtypes.py:528: Fu
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/usr/local/lib/python3.7/dist-packages/tensorflow/python/framework/dtypes.py:529: Fu
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/usr/local/lib/python3.7/dist-packages/tensorflow/python/framework/dtypes.py:530: Fu
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/usr/local/lib/python3.7/dist-packages/tensorflow/python/framework/dtypes.py:535: Fu
  np_resource = np.dtype([("resource", np.ubyte, 1)])
Using TensorFlow backend.
Command:  evaluate
Model:  /content/mask_rcnn_coco.h5
Dataset:  /content/Mask_RCNN/Coco_Data
```

```
Year:  2014
Logs:  /logs
Auto Download:  False

Configurations:
BACKBONE                       resnet101
BACKBONE_STRIDES               [4, 8, 16, 32, 64]
BATCH_SIZE                     1
BBOX_STD_DEV                   [0.1 0.1 0.2 0.2]
COMPUTE_BACKBONE_SHAPE         None
DETECTION_MAX_INSTANCES        100
DETECTION_MIN_CONFIDENCE       0
DETECTION_NMS_THRESHOLD        0.3
FPN_CLASSIF_FC_LAYERS_SIZE     1024
GPU_COUNT                      1
GRADIENT_CLIP_NORM             5.0
IMAGES_PER_GPU                 1
IMAGE_CHANNEL_COUNT            3
IMAGE_MAX_DIM                  1024
IMAGE_META_SIZE                93
IMAGE_MIN_DIM                  800
IMAGE_MIN_SCALE                0
IMAGE_RESIZE_MODE              square
IMAGE_SHAPE                    [1024 1024    3]
LEARNING_MOMENTUM              0.9
LEARNING_RATE                  0.001
LOSS_WEIGHTS                   {'rpn_class_loss': 1.0, 'rpn_bbox_loss': 1.0, 'mrcnn_
MASK_POOL_SIZE                 14
MASK_SHAPE                     [28, 28]
MAX_GT_INSTANCES               100
MEAN_PIXEL                     [123.7 116.8 103.9]
MINI_MASK_SHAPE                (56, 56)
NAME                           coco
NUM_CLASSES                    81
POOL_SIZE                      7
POST_NMS_ROIS_INFERENCE        1000
POST_NMS_ROIS_TRAINING         2000
PRE_NMS_LIMIT                  6000
ROI_POSITIVE_RATIO             0.33
RPN_ANCHOR_RATIOS              [0.5, 1, 2]
```

Main Results: Mask R-CNN is compared to the state-of-the-art methods in instance segmentation. All instantiations of our model outperform baseline variants of previous state-of-the-art models. This includes MNC and FCIS, the winners of the COCO 2015 and 2016 segmentation challenges, respectively. Without bells and whistles, Mask R-CNN with ResNet-101-FPN backbone outperforms FCIS+++, which includes multi-scale train/test, horizontal flip test, and online hard example mining (OHEM). While outside the scope of this work, its expected that many such improvements to be applicable to ours. Mask R-CNN achieves good results even under challenging conditions.

```
import Mask_RCNN.mrcnn.model as modellib
```

```
from Mask_RCNN.mrcnn import visualize
from  Mask_RCNN.samples.coco import coco


COCO_MODEL_PATH = "/content/mask_rcnn_coco.h5"
IMAGE_DIR = "/content/Mask_RCNN/images"


ROOT_DIR = "/content/Mask_RCNN"
import os
import skimage.io


MODEL_DIR = os.path.join(ROOT_DIR, "logs")


class InferenceConfig(coco.CocoConfig):
    # Set batch size to 1 since we'll be running inference on
    # one image at a time. Batch size = GPU_COUNT * IMAGES_PER_GPU
    GPU_COUNT = 1
    IMAGES_PER_GPU = 1

config = InferenceConfig()


# Create model object in inference mode.
model = modellib.MaskRCNN(mode="inference", model_dir=MODEL_DIR, config=config)

# Load weights trained on MS-COCO
model.load_weights(COCO_MODEL_PATH, by_name=True)


# COCO Class names
# Index of the class in the list is its ID. For example, to get ID of
# the teddy bear class, use: class_names.index('teddy bear')
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']
```

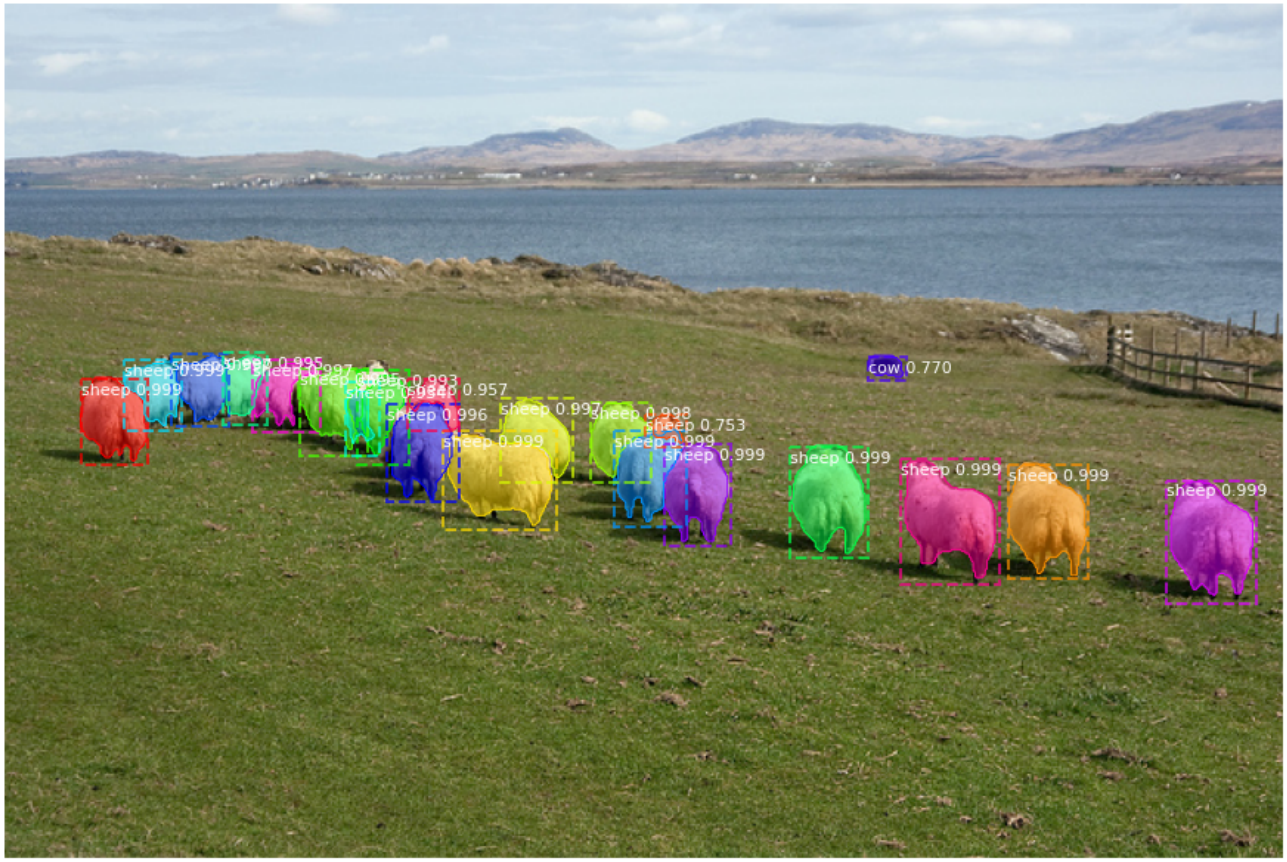
```
import random

# Load a random image from the images folder
file_names = next(os.walk(IMAGE_DIR))[2]
image = skimage.io.imread(os.path.join(IMAGE_DIR, random.choice(file_names)))

# Run detection
results = model.detect([image], verbose=1)

# Visualize results
r = results[0]
visualize.display_instances(image, r['rois'], r['masks'], r['class_ids'],
                            class_names, r['scores'])
```

```
Processing 1 images
image                    shape: (427, 640, 3)         min:    0.00000  max:  255.00000
molded_images            shape: (1, 1024, 1024, 3)    min: -123.70000  max:  143.10000
image_metas              shape: (1, 93)               min:    0.00000  max: 1024.00000
anchors                  shape: (1, 261888, 4)        min:   -0.35390  max:    1.29134
```



✓ 36s completed at 1:42 AM