Mask R-CNN

06/21/2021 meeting slides

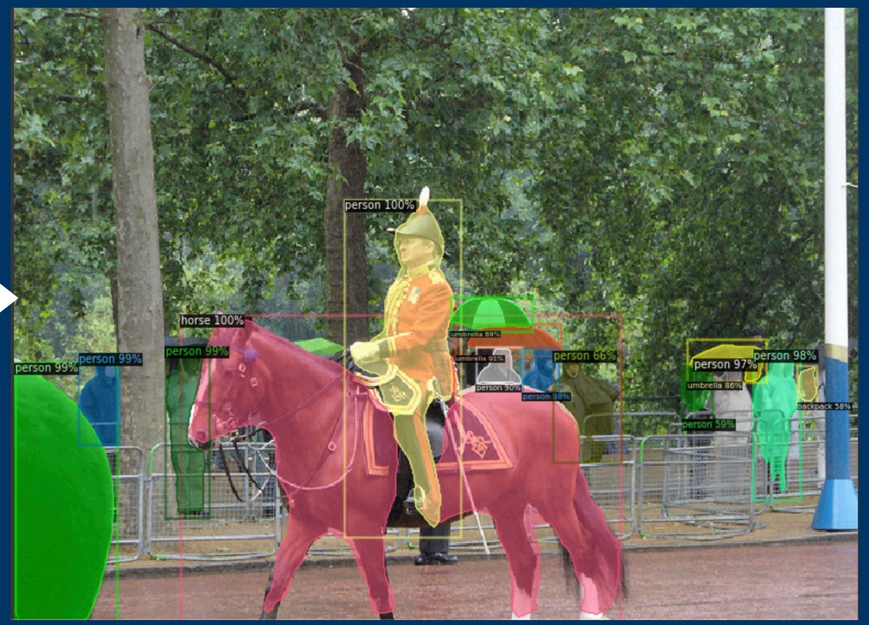
Indu Panigrahi

Overview

Image

Mask R-CNN

Bounding boxes, class labels, + masks



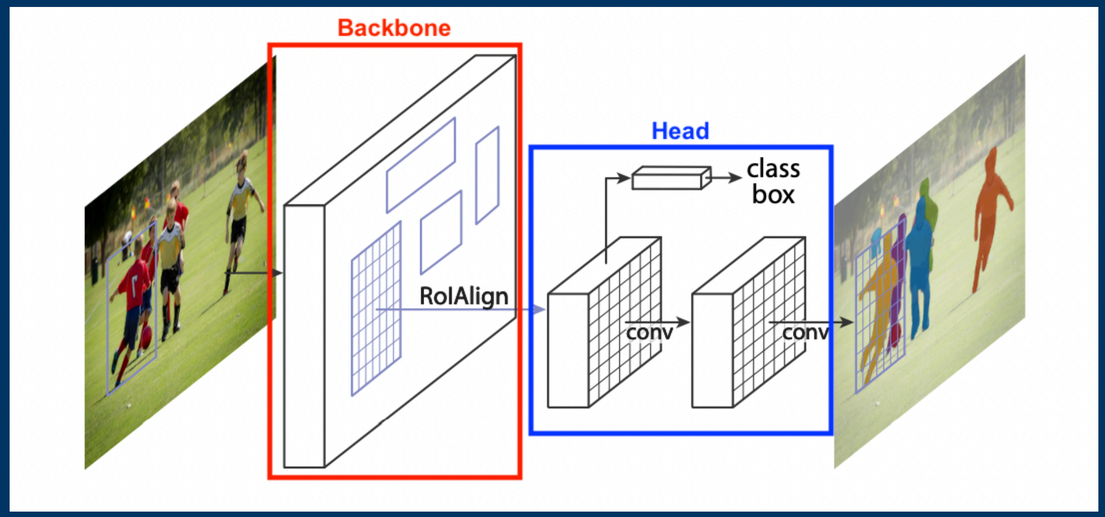
Images: https://colab.research.google.com/drive/16jcaJoc6bCFAQ96jDe2HwtXj7BMD_-m5

Mask R-CNN Architecture

- Goes through CNN and Region Proposal Network which proposes regions of interest (RoI) where objects might be located
    - Image enters CNN (usually ResNet) which then outputs feature maps
    - Sliding window used on the feature maps to select anchor boxes
    - Refine proposed boxes with bounding box regression and binary classification of anchor boxes (based on IoU with ground truth boxes)
- Each RoI goes through RoI Align and Fully Convolutional Network to predict the mask

More Info

- Backbone - network which takes image and outputs feature maps, see diagram
- Head - everything after RoI Align, see diagram



- IoU - calculated overlap between candidate box(es) and ground truth box(es)

Source: https://stats.stackexchange.com/questions/397767/how-do-backbone-and-head-architecture-work-in-mask-r-cnn/401107

Key Points

- Outputs masks (classifies pixels)
- Mask for each class rather than one mask containing all classes
- RoI Align maintains spatial information

Why / Possible Usage

- Pixel-level classification
- Instance segmented output to potentially feed into U-Net
- Seems to be most up to date

Detectron2 Implementation

- Library built by Facebook AI Research
- Detectron2-pretrainedmrcnn.ipynb on GitHub
  - I used the Colab Tutorial which uses Pytorch
  - Part 1: Use neural net pre-trained on COCO dataset to predict
  - Part 2: Train same net on Labrador images then predict

Setup with Pre-trained network

Then, we create a detectron2 config and a detectron2 `DefaultPredictor` to run inference on this image.

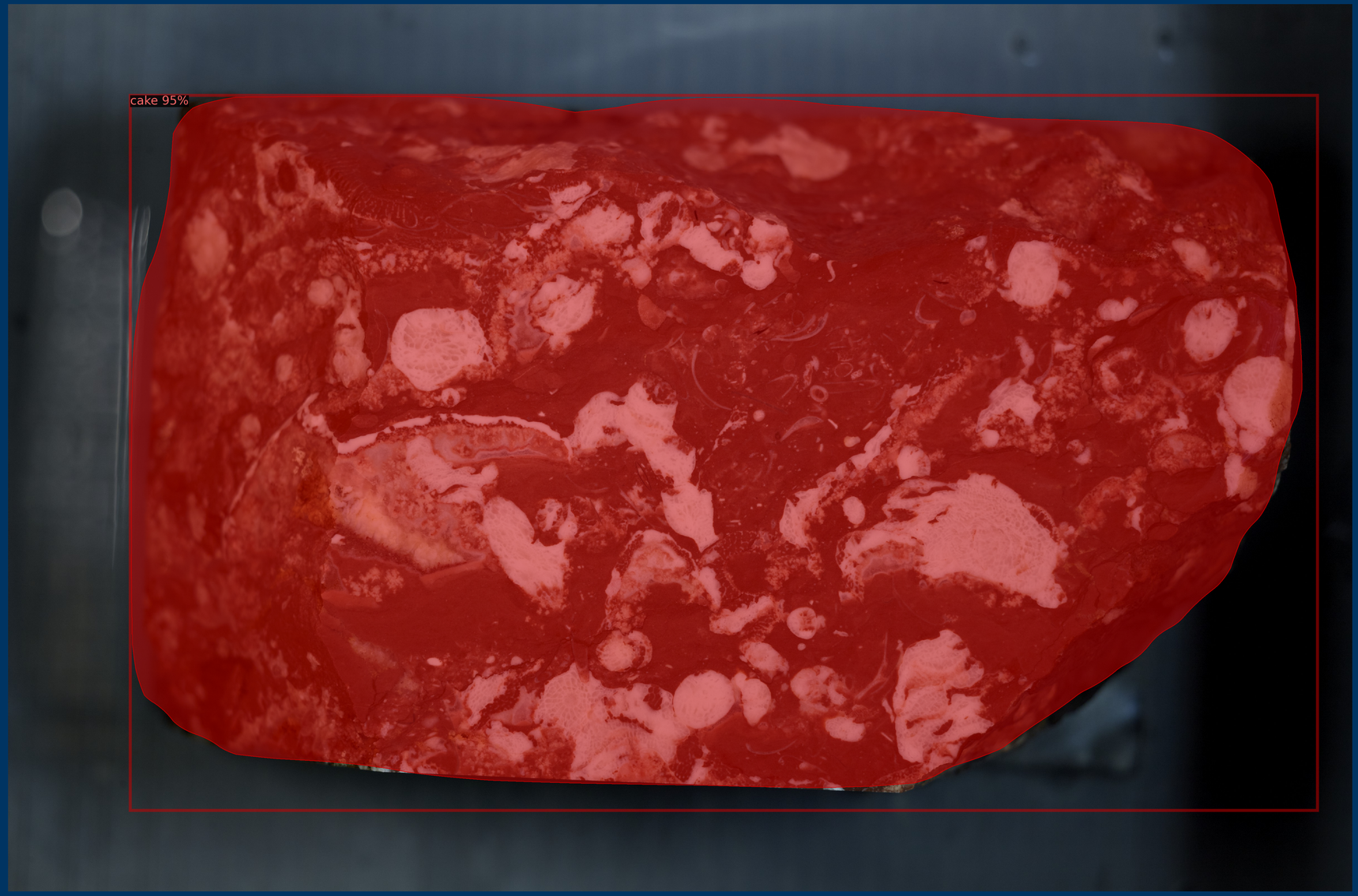
```
[8]  cfg = get_cfg()
     # add project-specific config (e.g., TensorMask) here if you're not running a model in detectron2's core library
     cfg.merge_from_file(model_zoo.get_config_file("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml"))
     cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.5  # set threshold for this model
     # Find a model from detectron2's model zoo. You can use the https://dl.fbaipublicfiles... url as well
     cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml")
     predictor = DefaultPredictor(cfg)
     outputs = predictor(im)
```

```
[ ]  # look at the outputs. See https://detectron2.readthedocs.io/tutorials/models.html#model-output-format for specification
     print(outputs["instances"].pred_classes)
     print(outputs["instances"].pred_boxes)
```

```
     tensor([55], device='cuda:0')
     Boxes(tensor([[ 436.7532,  326.5394, 4683.5166, 2883.0769]], device='cuda:0'))
```

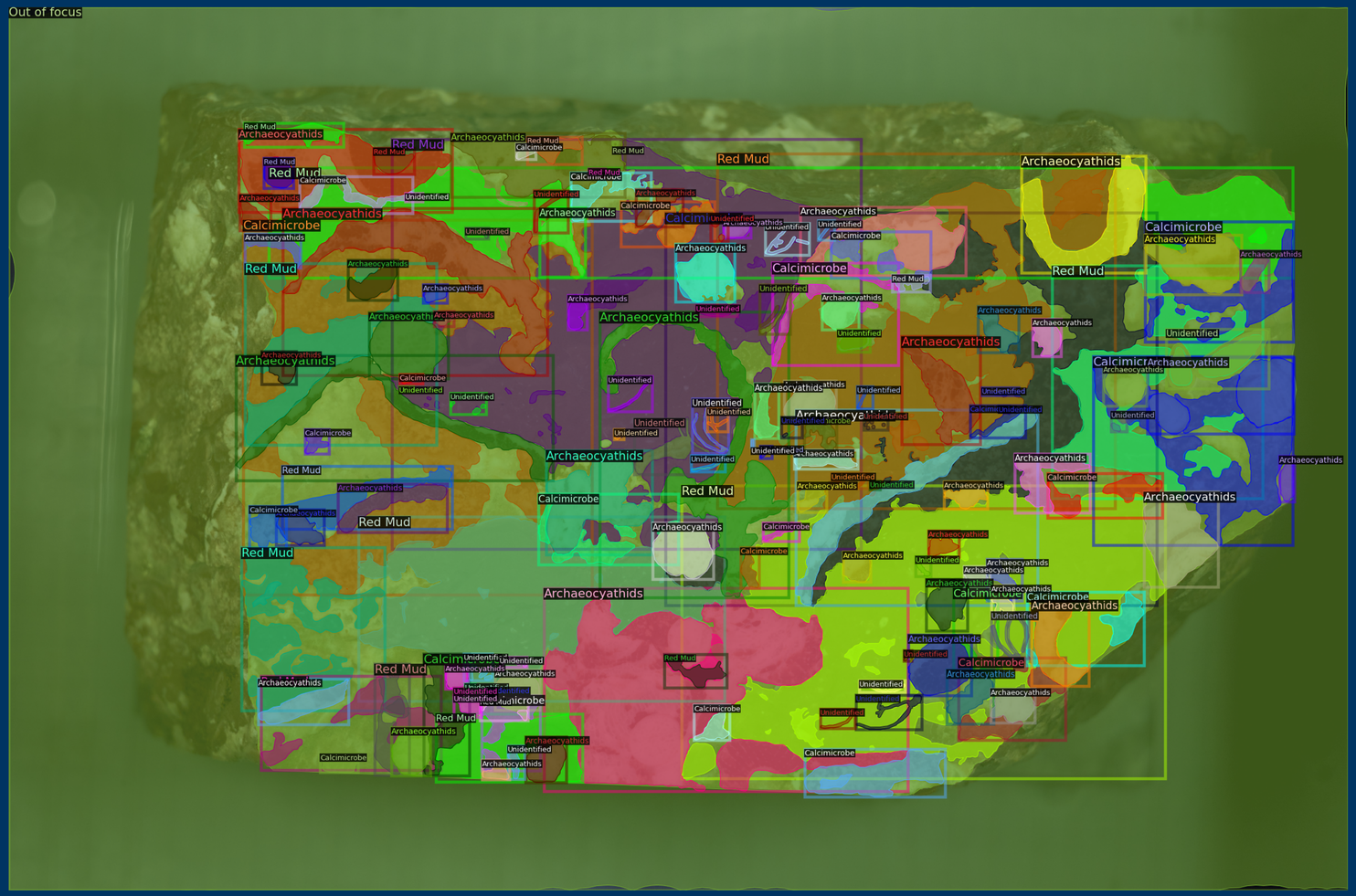
```
[9]  # We can use `Visualizer` to draw the predictions on the image.
     v = Visualizer(im[:, :, ::-1], MetadataCatalog.get(cfg.DATASETS.TRAIN[0]), scale=1.2)
     out = v.draw_instance_predictions(outputs["instances"].to("cpu"))
     cv2_imshow(out.get_image()[:, :, ::-1])
```

Part 1: Prediction without additional training



Part 2: Register the dataset

```
[ ]  # if your dataset is in COCO format, this cell can be replaced by the following three lines:
     from detectron2.data.datasets import register_coco_instances
     register_coco_instances("labrador_train", {}, "/content/drive/MyDrive/Colab Notebooks/lab_train/labrador.json", "/content/drive/MyDrive/C
     register_coco_instances("labrador_val", {}, "/content/drive/MyDrive/Colab Notebooks/lab_test/labrador_test.json", "/content/drive/MyDrive
```



Part 2: Train

Now, let's fine-tune a COCO-pretrained R50-FPN Mask R-CNN model on the dataset. It takes ~6 minutes to train 300 iterations on Colab's K80 GPU, or ~2 minutes on a P100 GPU for the balloon dataset from the original tutorial.

```
from detectron2.engine import DefaultTrainer

cfg = get_cfg()
cfg.merge_from_file(model_zoo.get_config_file("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml"))
cfg.DATASETS.TRAIN = ("labrador_train",)
cfg.DATASETS.TEST = ()
cfg.DATALOADER.NUM_WORKERS = 2
cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml")  # Let training initialize from
cfg.SOLVER.IMS_PER_BATCH = 1
cfg.SOLVER.BASE_LR = 0.00025  # pick a good LR
cfg.SOLVER.MAX_ITER = 300    # 300 iterations seems good enough for this toy dataset; you will need to train longer for a practical datas
cfg.SOLVER.STEPS = []        # do not decay learning rate
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 512   # originally 128 which was faster, and good enough for the balloon dataset (default: 512
cfg.MODEL.ROI_HEADS.NUM_CLASSES = 6  # number of classes(see https://detectron2.readthedocs.io/tutorials/datasets.html#update-the-config-
# NOTE: this config means the number of classes, but a few popular unofficial tutorials incorrect uses num_classes+1 here.

os.makedirs(cfg.OUTPUT_DIR, exist_ok=True)
trainer = DefaultTrainer(cfg)
trainer.resume_or_load(resume=False)
trainer.train()
```

Part 2: Predict

Comments

- Could be helpful to find a net pre-trained on objects which are not commonplace
    - Satellite images?
- Detectron2 allows custom training loop
- Another option is https://github.com/matterport/Mask_RCNN which uses Tensorflow