

This tutorial is used to teach how to create your own datasets and training model.

1. MaskRCNN

This model use a form of datasets called "coco-datasets" , it's structure is looked like this:

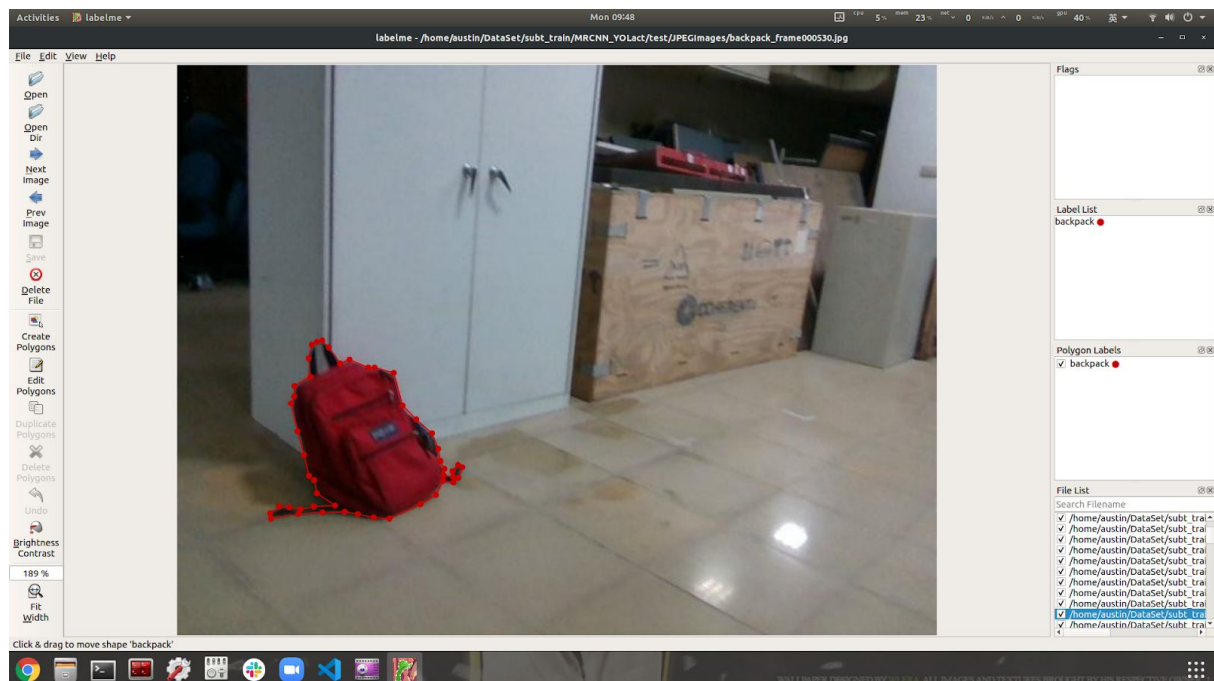
```
austin@austin:~/DataSet/subt_train/MRCNN_YOLact/subt_new$ tree -L 2
.
├── subt_test
│   └── JPEGImages
├── subt_train
│   └── JPEGImages
├── test.json
└── train.json

4 directories, 2 files
```

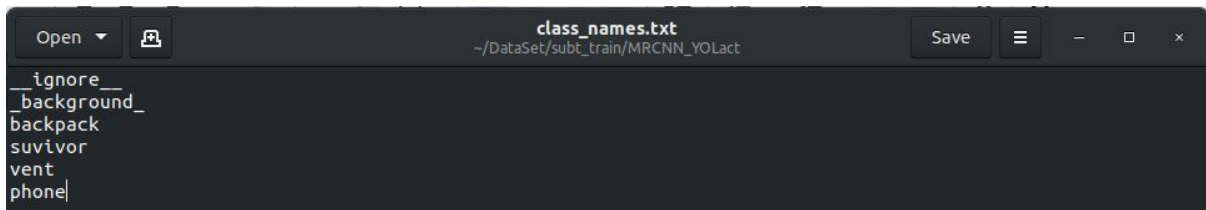
The folder "subt_train" and "subt_test" contain the RGB image folder "JPEGImages" for training and testing , train.json and test.json are the json file that have the information of each RGB image , for example , the location of the mask.

To create coco-datasets , first you need to collect the RGB image of all classes you need , and use the tool called "labelme" to label the object , just type "labelme" at the terminal and click "Open Dir" to open your image folder which you want to label :

```
austin@austin:~$ labelme
```



After lableing all image and saving , your image folder will contain json file belonging to each image. Before creating the "coco-datasets" you need to create a txt file that list the class :



Then we can create the "coco-datasets" by using this command :

"python3 labelme2coco.py --labels "class name txt file dir" "image folder dir" "generating folder dir"

```
austin@austin:~$ python3 labelme2coco.py --labels "class name txt file dir" "image folder dir" "generating folder dir"
```

And then will get a json file like "train.json" or "test.json" and a folder that contain RGB image like "subt_train" or subt_test".

Do above for training data and testing data.

After creating datasets , you can use the our colab notebook to train MaskRCNN.

Change the direction of the datasets to your own :

```
[ ] from detectron2.data.datasets import register_coco_instances

register_coco_instances('subt_train', {},
                       '/content/subt-urban-coco-dataset/SubT_urban_train.json',
                       '/content/subt-urban-coco-dataset/SubT_urban_train')
register_coco_instances('subt_val', {},
                       '/content/subt-urban-coco-dataset/SubT_urban_val.json',
                       '/content/subt-urban-coco-dataset/SubT_urban_val')
```

Run the notebook and it will start training :

```
Training

[ ] os.makedirs(cfg.OUTPUT_DIR, exist_ok=True)

trainer = DefaultTrainer(cfg)
trainer.resume_or_load(False)
trainer.train()

Skip loading parameter 'roi_heads.mask_head.predictor.bias' to the model due to incompatible shapes: (80,) in the checkpoint but (5,) in the model! You might want to double check if this is expected.
[01/01 11:03:23 d2.engine.train_loop]: Starting training from iteration 0
/usr/local/lib/python3.8/dist-packages/detectron2/structures/masks.py:345: UserWarning: This overload of nonzero is deprecated:
  nonzero()
Consider using one of the following signatures instead:
  nonzero(*, bool as_tuple) (Triggered internally at /pytorch/torch/csrc/utils/python_arg_parser.cpp:802.)
  item = item.nonzero().squeeze(1).cpu().numpy().tolist()
[01/01 11:03:48 d2.utils.events]: eta: 10:48:39 iter: 19 total loss: 2.998 loss cls: 1.799 loss box reg: 0.4846 loss mask: 0.6947 loss_rpn_cls: 0.01079 loss_rpn_loc: 0.003485 time: 1.2727 data time: 0.11
[01/01 11:04:15 d2.utils.events]: eta: 11:02:02 iter: 39 total loss: 2.879 loss cls: 1.67 loss box reg: 0.4629 loss mask: 0.6913 loss_rpn_cls: 0.01163 loss_rpn_loc: 0.003423 time: 1.3136 data time: 0.11
[01/01 11:04:43 d2.utils.events]: eta: 11:27:28 iter: 59 total loss: 2.579 loss cls: 1.438 loss box reg: 0.4552 loss mask: 0.6852 loss_rpn_cls: 0.0105 loss_rpn_loc: 0.003998 time: 1.3448 data time: 0.11
[01/01 11:05:11 d2.utils.events]: eta: 11:38:17 iter: 79 total loss: 2.282 loss cls: 1.128 loss box reg: 0.4783 loss mask: 0.6745 loss_rpn_cls: 0.008376 loss_rpn_loc: 0.004194 time: 1.3612 data time: 0.11
[01/01 11:05:39 d2.utils.events]: eta: 11:39:44 iter: 99 total loss: 1.949 loss cls: 0.8132 loss box reg: 0.4484 loss mask: 0.6687 loss_rpn_cls: 0.009819 loss_rpn_loc: 0.004099 time: 1.3676 data time: 0.11
[01/01 11:06:08 d2.utils.events]: eta: 11:41:59 iter: 119 total loss: 1.74 loss cls: 0.6181 loss box reg: 0.4777 loss mask: 0.6415 loss_rpn_cls: 0.0121 loss_rpn_loc: 0.003763 time: 1.3893 data time: 0.11
[01/01 11:06:36 d2.utils.events]: eta: 11:45:46 iter: 139 total loss: 1.541 loss cls: 0.4764 loss box reg: 0.4271 loss mask: 0.6248 loss_rpn_cls: 0.01399 loss_rpn_loc: 0.004449 time: 1.3872 data time: 0.11
[01/01 11:07:04 d2.utils.events]: eta: 11:44:39 iter: 159 total loss: 1.565 loss cls: 0.4695 loss box reg: 0.4875 loss mask: 0.5972 loss_rpn_cls: 0.008732 loss_rpn_loc: 0.003914 time: 1.3871 data time: 0.11
[01/01 11:07:32 d2.utils.events]: eta: 11:43:52 iter: 179 total loss: 1.546 loss cls: 0.4649 loss box reg: 0.5153 loss mask: 0.5593 loss_rpn_cls: 0.01045 loss_rpn_loc: 0.003916 time: 1.3897 data time: 0.11
[01/01 11:08:01 d2.utils.events]: eta: 11:45:41 iter: 199 total loss: 1.446 loss cls: 0.4337 loss box reg: 0.4708 loss mask: 0.5404 loss_rpn_cls: 0.009195 loss_rpn_loc: 0.003173 time: 1.3945 data time: 0.11
[01/01 11:08:30 d2.utils.events]: eta: 11:48:11 iter: 219 total loss: 1.41 loss cls: 0.4039 loss box reg: 0.4832 loss mask: 0.5153 loss_rpn_cls: 0.008078 loss_rpn_loc: 0.003764 time: 1.4007 data time: 0.11
[01/01 11:08:59 d2.utils.events]: eta: 11:48:59 iter: 239 total loss: 1.447 loss cls: 0.4221 loss box reg: 0.537 loss mask: 0.4916 loss_rpn_cls: 0.006024 loss_rpn_loc: 0.00305 time: 1.4025 data time: 0.11
[01/01 11:09:27 d2.utils.events]: eta: 11:46:41 iter: 259 total loss: 1.366 loss cls: 0.3833 loss box reg: 0.5342 loss mask: 0.4773 loss_rpn_cls: 0.007008 loss_rpn_loc: 0.003412 time: 1.4020 data time: 0.11
[01/01 11:09:55 d2.utils.events]: eta: 11:46:47 iter: 279 total loss: 1.332 loss cls: 0.3718 loss box reg: 0.5203 loss mask: 0.4259 loss_rpn_cls: 0.005182 loss_rpn_loc: 0.003478 time: 1.4031 data time: 0.11
[01/01 11:10:23 d2.utils.events]: eta: 11:45:41 iter: 299 total loss: 1.332 loss cls: 0.3587 loss box reg: 0.5648 loss mask: 0.3842 loss_rpn_cls: 0.003527 loss_rpn_loc: 0.00337 time: 1.4016 data time: 0.11
[01/01 11:10:51 d2.utils.events]: eta: 11:44:29 iter: 319 total loss: 1.239 loss cls: 0.3361 loss box reg: 0.5333 loss mask: 0.3658 loss_rpn_cls: 0.00483 loss_rpn_loc: 0.004022 time: 1.4018 data time: 0.11
[01/01 11:11:19 d2.utils.events]: eta: 11:44:45 iter: 339 total loss: 1.174 loss cls: 0.3328 loss box reg: 0.5103 loss mask: 0.3313 loss_rpn_cls: 0.00619 loss_rpn_loc: 0.004473 time: 1.4028 data time: 0.11
```

2. Yolact

This model also using the "coco-datasets" , you can use our colab notebook to train Yolact. After cloning the Yolact from github , click the file icon at the left side and click "Yolact-pytorch" -> "data" -> "config.py" , change the file direction at "custom_dataset = ..." to your own.

```
1 [!] git clone https://github.com/ARX-MCTU/Yolact-pytorch
Cloning into 'Yolact-pytorch':
remote: Enumerating objects: 444, done.
remote: Counting objects: 100% (444/444), done.
remote: Compressing objects: 100% (315/315), done.
remote: Total 444 (delta 144), reused 391 (delta 100)
Receiving objects: 100% (444/444), 12.85 MiB | 25.01
Resolving deltas: 100% (144/144), done.

2 Import Libraries
import sys
import os
sys.path.append('content/Yolact-pytorch')
import gdown
from zipfile import ZipFile
import logging

3 Install related module
[ ] !pip install tensorboardX
!pip install terminaltables

Collecting tensorboardX
  Downloading https://files.pythonhosted.org/packages/31/7d/10/94b/
Requirement already satisfied: protobuf>=3.8.0 in /usr
Requirement already satisfied: numpy in /usr/local/lib
Requirement already satisfied: six in /usr/local/lib
```

```
99 # ----- DATASETS ----- #
100 coco_dataset = Config({'name': 'COCO 2017',
101                        'train_images': '/home/alex/yolact_tmp/yolact_minimal/data/train2017',
102                        'train_info': '/home/alex/yolact_tmp/yolact_minimal/data/annotations/instances_train2017.json',
103                        'valid_images': '/home/alex/yolact_tmp/yolact_minimal/data/val2017',
104                        'valid_info': '/home/alex/yolact_tmp/yolact_minimal/data/annotations/instances_val2017.json',
105                        'class_names': COCO_CLASSES})
106
107 pascal_sbd_dataset = Config({'name': 'Pascal SBD 2012',
108                              'train_images': '/home/feiyu/Data/pascal_sbd/dataset/img',
109                              'train_info': '/home/feiyu/Data/pascal_sbd/dataset/img',
110                              'valid_images': '/home/feiyu/Data/pascal_sbd/dataset/pascal_sbd_train.json',
111                              'valid_info': '/home/feiyu/Data/pascal_sbd/dataset/pascal_sbd_val.json',
112                              'class_names': PASCAL_CLASSES})
113
114 custom_dataset = Config({'name': 'SubT',
115                           'train_images': '/home/alex/yolact_tmp/yolact_minimal/data/SubT_urban_train', # No need to add
116                           'train_info': '/home/alex/yolact_tmp/yolact_minimal/data/annotations/SubT_urban_train.json',
117                           'valid_images': '/home/alex/yolact_tmp/yolact_minimal/data/SubT_urban_val',
118                           'valid_info': '/home/alex/yolact_tmp/yolact_minimal/data/annotations/SubT_urban_val.json',
119                           'class_names': CUSTOM_CLASSES})
120
121 # ----- TRANSFORMS ----- #
122 resnet_transform = Config({'channel_order': 'RGB',
123                            'normalize': True,
124                            'subtract_means': False,
125                            'to_float': False})
126
127 # ----- BACKBONES ----- #
128 resnet101_backbone = Config({'name': 'ResNet101',
129                              'path': 'resnet101_reducedfc.pth',
130                              'type': 'ResNetBackbone',
131                              'args': ([3, 4, 23, 3]),
132                              'transform': resnet_transform,
133                              'selected_layers': [1, 2, 3]})
134
135 resnet50_backbone = resnet101_backbone.copy({'name': 'ResNet50',
136                                              'path': 'resnet50-19c8e357.pth',
137                                              'args': ([3, 4, 6, 3],)})
138
139 resnet101_coco_config = Config({
```

Run the notebook and it will start training :

```
Train
you need to change the path of datasets in data/config.py first.

python /content/Yolact-pytorch/train.py --config=res101_custom_config

mode = random.choice(self.sample_options)
18 | B: 5.259 | C: 18.685 | M: 5.637 | S: 2.580 | T: 24.081 | Lr: 1.18e-04 | t data: 0.004 | t forward: 0.376 | t total: 1.124 | ETA: 10 days, 2:29:36
19 | B: 4.816 | C: 8.713 | M: 5.329 | S: 1.692 | T: 20.550 | Lr: 1.36e-04 | t data: 0.005 | t forward: 0.387 | t total: 1.128 | ETA: 10 days, 4:53:28
20 | B: 4.713 | C: 7.754 | M: 5.253 | S: 1.266 | T: 18.986 | Lr: 1.54e-04 | t data: 0.004 | t forward: 0.396 | t total: 1.145 | ETA: 10 days, 6:59:10
40 | B: 4.602 | C: 7.089 | M: 5.254 | S: 1.038 | T: 17.972 | Lr: 1.72e-04 | t data: 0.004 | t forward: 0.397 | t total: 1.140 | ETA: 10 days, 9:17:10
50 | B: 4.513 | C: 6.649 | M: 5.205 | S: 0.868 | T: 17.235 | Lr: 1.90e-04 | t data: 0.004 | t forward: 0.415 | t total: 1.147 | ETA: 10 days, 11:43:13
60 | B: 4.440 | C: 6.387 | M: 5.163 | S: 0.770 | T: 16.679 | Lr: 2.08e-04 | t data: 0.004 | t forward: 0.424 | t total: 1.212 | ETA: 10 days, 13:57:39
70 | B: 4.379 | C: 6.037 | M: 5.115 | S: 0.700 | T: 16.251 | Lr: 2.26e-04 | t data: 0.004 | t forward: 0.425 | t total: 1.213 | ETA: 10 days, 16:24:41
80 | B: 4.350 | C: 5.820 | M: 5.094 | S: 0.641 | T: 15.994 | Lr: 2.44e-04 | t data: 0.004 | t forward: 0.440 | t total: 1.243 | ETA: 10 days, 19:04:17
90 | B: 4.309 | C: 5.648 | M: 5.051 | S: 0.599 | T: 15.607 | Lr: 2.62e-04 | t data: 0.004 | t forward: 0.432 | t total: 1.212 | ETA: 10 days, 21:09:39
100 | B: 4.257 | C: 5.356 | M: 4.971 | S: 0.535 | T: 15.119 | Lr: 2.80e-04 | t data: 0.004 | t forward: 0.409 | t total: 1.188 | ETA: 10 days, 22:13:00
110 | B: 4.189 | C: 4.759 | M: 4.894 | S: 0.312 | T: 14.154 | Lr: 2.98e-04 | t data: 0.004 | t forward: 0.428 | t total: 1.195 | ETA: 11 days, 0:05:37
120 | B: 4.163 | C: 4.482 | M: 4.832 | S: 0.259 | T: 13.736 | Lr: 3.16e-04 | t data: 0.004 | t forward: 0.430 | t total: 1.226 | ETA: 11 days, 3:11:32
130 | B: 4.112 | C: 4.295 | M: 4.783 | S: 0.237 | T: 13.427 | Lr: 3.34e-04 | t data: 0.004 | t forward: 0.425 | t total: 1.261 | ETA: 11 days, 4:58:40
140 | B: 4.112 | C: 4.154 | M: 4.704 | S: 0.230 | T: 13.200 | Lr: 3.52e-04 | t data: 0.004 | t forward: 0.432 | t total: 1.272 | ETA: 11 days, 6:26:05
150 | B: 4.077 | C: 4.042 | M: 4.649 | S: 0.221 | T: 12.988 | Lr: 3.70e-04 | t data: 0.004 | t forward: 0.412 | t total: 1.200 | ETA: 11 days, 7:37:24
160 | B: 4.053 | C: 3.942 | M: 4.588 | S: 0.208 | T: 12.791 | Lr: 3.88e-04 | t data: 0.004 | t forward: 0.435 | t total: 1.222 | ETA: 11 days, 8:28:21
170 | B: 3.999 | C: 3.833 | M: 4.449 | S: 0.199 | T: 12.480 | Lr: 4.06e-04 | t data: 0.004 | t forward: 0.420 | t total: 1.234 | ETA: 11 days, 8:23:06
180 | B: 3.917 | C: 3.731 | M: 4.324 | S: 0.197 | T: 12.170 | Lr: 4.24e-04 | t data: 0.004 | t forward: 0.424 | t total: 1.208 | ETA: 11 days, 7:27:27
190 | B: 3.849 | C: 3.627 | M: 4.224 | S: 0.186 | T: 11.887 | Lr: 4.42e-04 | t data: 0.004 | t forward: 0.400 | t total: 1.225 | ETA: 11 days, 6:34:45
200 | B: 3.805 | C: 3.557 | M: 4.115 | S: 0.176 | T: 11.653 | Lr: 4.60e-04 | t data: 0.005 | t forward: 0.431 | t total: 1.230 | ETA: 11 days, 6:38:56
210 | B: 3.680 | C: 3.469 | M: 4.005 | S: 0.170 | T: 11.324 | Lr: 4.78e-04 | t data: 0.004 | t forward: 0.418 | t total: 1.208 | ETA: 11 days, 6:48:28
220 | B: 3.615 | C: 3.386 | M: 3.912 | S: 0.172 | T: 11.085 | Lr: 4.96e-04 | t data: 0.005 | t forward: 0.424 | t total: 1.206 | ETA: 11 days, 6:29:59
230 | B: 3.536 | C: 3.297 | M: 3.794 | S: 0.173 | T: 10.799 | Lr: 5.14e-04 | t data: 0.004 | t forward: 0.428 | t total: 1.219 | ETA: 11 days, 6:42:10
```

3. FCN

The datasets of the FCN model need two data type "RGB image" and "Mask image", its structure is looked like this :

```
austin@austin:~/DataSet/subt_train/FCN_train$ tree -L 2
.
├── test
│   ├── images
│   └── masks
└── train
    ├── images
    └── masks

6 directories, 0 files
```

RGB image (left) and Mask image (right) :



To create mask of image , After using lableme to generate json file for each image , make a folder that only contain these json file , and run the "genejson.sh" (you need to change "dir" and "path" to your own) to create mask , it will bw generate at the json file folder .

You can use our colab notebook to train FCN.

Change the direction of the datasets to your own :

Define path, directory training environment

```
[ ] # get data
FullPath = os.getcwd()
data_dir = os.path.join(FullPath + "/data/FCN_train")
if not os.path.exists(data_dir):
    print("Data not found!")
```

Run the notebook and it will start training :

```
train()
/usr/local/lib/python3.6/dist-packages/torch/optim/lr_scheduler.py:136: UserWarning: Detected call of `lr_scheduler.step()` before `optimizer.step()`. In PyTorch 1.1.0 and later, you should call them in the opposite order: `optimizer.step()` before `lr_scheduler.step()`. See https://pytorch.org/docs/stable/optim.html#how-to-adjust-learning-rate, for more information.
epoch1, iter0, loss: 0.78598895157135
epoch1, iter1, loss: 0.6943728871400479
epoch1, iter2, loss: 0.678621338052458
epoch1, iter3, loss: 0.6781897206306458
epoch1, iter4, loss: 0.6787481741905212
epoch1, iter5, loss: 0.6647931385520123
epoch1, iter6, loss: 0.6649534106254578
epoch1, iter7, loss: 0.663619875907898
epoch1, iter8, loss: 0.6579977869907488
epoch1, iter9, loss: 0.6676369389425354
epoch1, iter10, loss: 0.668355192565918
epoch1, iter11, loss: 0.6686330295562744
epoch1, iter12, loss: 0.6625357866287231
epoch1, iter13, loss: 0.6689653838528259
epoch1, iter14, loss: 0.6623077392578125
epoch1, iter15, loss: 0.6570571064848936
epoch1, iter16, loss: 0.6546377539634705
epoch1, iter17, loss: 0.6635231971748723
```