# FaceMaskSSDMobileNetV3_FINAL_HEROKU_DEPLOYED

October 16, 2021

## 1 Start

[ ]:
```
pip install -U git+https://github.com/albumentations-team/albumentations
```

```
Collecting git+https://github.com/albumentations-team/albumentations
  Cloning https://github.com/albumentations-team/albumentations to /tmp/pip-req-
build-q5pwd9zy
  Running command git clone -q https://github.com/albumentations-
team/albumentations /tmp/pip-req-build-q5pwd9zy
Requirement already satisfied: numpy>=1.11.1 in /usr/local/lib/python3.7/dist-
packages (from albumentations==1.1.0) (1.19.5)
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages
(from albumentations==1.1.0) (1.4.1)
Requirement already satisfied: scikit-image>=0.16.1 in
/usr/local/lib/python3.7/dist-packages (from albumentations==1.1.0) (0.16.2)
Requirement already satisfied: PyYAML in /usr/local/lib/python3.7/dist-packages
(from albumentations==1.1.0) (3.13)
Requirement already satisfied: qudida>=0.0.4 in /usr/local/lib/python3.7/dist-
packages (from albumentations==1.1.0) (0.0.4)
Requirement already satisfied: opencv-python>=4.1.1 in
/usr/local/lib/python3.7/dist-packages (from albumentations==1.1.0) (4.1.2.30)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from
qudida>=0.0.4->albumentations==1.1.0) (3.7.4.3)
Requirement already satisfied: scikit-learn>=0.19.1 in
/usr/local/lib/python3.7/dist-packages (from
qudida>=0.0.4->albumentations==1.1.0) (0.22.2.post1)
Requirement already satisfied: opencv-python-headless>=4.0.1 in
/usr/local/lib/python3.7/dist-packages (from
qudida>=0.0.4->albumentations==1.1.0) (4.5.3.56)
Requirement already satisfied: PyWavelets>=0.4.0 in
/usr/local/lib/python3.7/dist-packages (from scikit-
image>=0.16.1->albumentations==1.1.0) (1.1.1)
Requirement already satisfied: imageio>=2.3.0 in /usr/local/lib/python3.7/dist-
packages (from scikit-image>=0.16.1->albumentations==1.1.0) (2.4.1)
Requirement already satisfied: networkx>=2.0 in /usr/local/lib/python3.7/dist-
packages (from scikit-image>=0.16.1->albumentations==1.1.0) (2.6.3)
Requirement already satisfied: matplotlib!=3.0.0,>=2.0.0 in
```

```
/usr/local/lib/python3.7/dist-packages (from scikit-
image>=0.16.1->albumentations==1.1.0) (3.2.2)
Requirement already satisfied: pillow>=4.3.0 in /usr/local/lib/python3.7/dist-
packages (from scikit-image>=0.16.1->albumentations==1.1.0) (7.1.2)
Requirement already satisfied: python-dateutil>=2.1 in
/usr/local/lib/python3.7/dist-packages (from matplotlib!=3.0.0,>=2.0.0->scikit-
image>=0.16.1->albumentations==1.1.0) (2.8.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/usr/local/lib/python3.7/dist-packages (from matplotlib!=3.0.0,>=2.0.0->scikit-
image>=0.16.1->albumentations==1.1.0) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.7/dist-packages (from matplotlib!=3.0.0,>=2.0.0->scikit-
image>=0.16.1->albumentations==1.1.0) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-
packages (from matplotlib!=3.0.0,>=2.0.0->scikit-
image>=0.16.1->albumentations==1.1.0) (0.10.0)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages
(from cycler>=0.10->matplotlib!=3.0.0,>=2.0.0->scikit-
image>=0.16.1->albumentations==1.1.0) (1.15.0)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-
packages (from scikit-learn>=0.19.1->qudida>=0.0.4->albumentations==1.1.0)
(1.0.1)
```

[ ]:
```
from google.colab import drive
drive.mount("/content/gdrive")
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call
drive.mount("/content/gdrive", force_remount=True).
```

[ ]:
```
!nvidia-smi
```

```
Thu Oct 14 15:21:18 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.74       Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE…  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   46C    P0    28W / 250W |      0MiB / 16280MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
```

```
|===============================================================================|
|  No running processes found                                                   |
+-------------------------------------------------------------------------------+
```

## 2 Modules

[ ]:
```
# image
import cv2
import matplotlib.pyplot as plt
import seaborn as sns

# DL
import torch
import torchvision
from torch.utils.data import Dataset, DataLoader
from torch import nn as nn

# file
from glob import glob
import pandas as pd
import numpy as np
import os
import xml.etree.ElementTree as ET
from sklearn.model_selection import train_test_split
from tqdm import tqdm
import json

# augmentation
import albumentations as A
```

## 3 Data Extraction

[ ]:
```
!mkdir csvFolder
```

### 3.1 Data 1

https://www.kaggle.com/andrewmvd/face-mask-detection

[ ]:
```

```

```
!curl --header 'Host: storage.googleapis.com' --user-agent 'Mozilla/5.0 (X11;␣
↪Ubuntu; Linux x86_64; rv:93.0) Gecko/20100101 Firefox/93.0' --header 'Accept:
↪ text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
↪webp,*/*;q=0.8' --header 'Accept-Language: en-US,en;q=0.5' --referer 'https:/
↪/www.kaggle.com/' --header 'DNT: 1' --header 'Upgrade-Insecure-Requests: 1'␣
↪--header 'Sec-Fetch-Dest: document' --header 'Sec-Fetch-Mode: navigate'␣
↪--header 'Sec-Fetch-Site: cross-site' --header 'Sec-Fetch-User: ?1' 'https://
↪storage.googleapis.com/kaggle-data-sets/667889/1176415/bundle/archive.zip?
↪X-Goog-Algorithm=GOOG4-RSA-SHA256&X-Goog-Credential=gcp-kaggle-com%40kaggle-161607.
↪iam.gserviceaccount.
↪com%2F20211014%2Fauto%2Fstorage%2Fgoog4_request&X-Goog-Date=20211014T112923Z&X-Goog-Expires
↪--output 'archive.zip'
```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  397M  100  397M    0     0  46.2M      0  0:00:08  0:00:08 --:--:-- 48.9M
```

[ ]:
```
!unzip /content/archive.zip -d /content/data1
```

[ ]:
```
!rm /content/archive.zip
```

[ ]:
```
dataOneImagesPath = "/content/data1/images/"
dataOneAnnotationPath = "/content/data1/annotations/"
```

[ ]:
```
import xml.etree.ElementTree as ET


#convert the object annotation from XML file to a dataframe
def xml_to_df(path):

    xml_list = []
    for xml_file in glob(path + '/*.xml'):
        tree = ET.parse(xml_file)
        root = tree.getroot()
        for member in root.findall('object'):

            ## chnage object index number, chnages with xml file

            xmin = float(member[5][0].text)
            ymin = float(member[5][1].text)
            xmax = float(member[5][2].text)
            ymax = float(member[5][3].text)

            img_width = float(root.find('size')[0].text)
            img_height = float(root.find('size')[1].text)
            aspectRatio = img_width / img_height
```

```
            box_width = (xmax-xmin)
            box_height = (ymax-ymin)
            bboxArea = box_width * box_height



            if xmax <= img_width and ymax <= img_height and bboxArea > 400:

                value = (dataOneImagesPath + root.find('filename').text,
                        member[0].text,
                        xmin,
                        ymin,
                        xmax,
                        ymax,
                        )

                xml_list.append(value)

    column_name = ['filename',
                   'class', 'xmin', 'ymin', 'xmax', 'ymax']


    xml_df = pd.DataFrame(xml_list, columns=column_name)
    return xml_df


df1 = xml_to_df(dataOneAnnotationPath)

df1.to_csv(os.path.join("/content/csvFolder", "data1.csv"), index=False)
```

[ ]:
```
df1.shape
```

[ ]:
```
(2346, 6)
```

[ ]:
```
df1['class'].value_counts()
```

[ ]:
```
with_mask               1926
without_mask             337
mask_weared_incorrect     83
Name: class, dtype: int64
```

### 3.2 Data 2

https://www.kaggle.com/wobotintelligence/face-mask-detection-dataset

[ ]:

```
!curl --header 'Host: storage.googleapis.com' --user-agent 'Mozilla/5.0 (X11;␣
↪Ubuntu; Linux x86_64; rv:93.0) Gecko/20100101 Firefox/93.0' --header 'Accept:
↪ text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
↪webp,*/*;q=0.8' --header 'Accept-Language: en-US,en;q=0.5' --referer 'https:/
↪/www.kaggle.com/' --header 'DNT: 1' --header 'Alt-Used: storage.googleapis.
↪com' --header 'Upgrade-Insecure-Requests: 1' --header 'Sec-Fetch-Dest:␣
↪document' --header 'Sec-Fetch-Mode: navigate' --header 'Sec-Fetch-Site:␣
↪cross-site' --header 'Sec-Fetch-User: ?1' 'https://storage.googleapis.com/
↪kaggle-data-sets/710024/1246711/bundle/archive.zip?
↪X-Goog-Algorithm=GOOG4-RSA-SHA256&X-Goog-Credential=gcp-kaggle-com%40kaggle-161607.
↪iam.gserviceaccount.
↪com%2F20211014%2Fauto%2Fstorage%2Fgoog4_request&X-Goog-Date=20211014T112951Z&X-Goog-Expires
↪--output 'archive.zip'
```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 2558M  100 2558M    0     0  54.2M      0  0:00:47  0:00:47 --:--:-- 68.4M
```

[ ]:
```
!unzip /content/archive.zip -d /content/data2
```

[ ]:
```
!rm /content/archive.zip
```

[ ]:
```
dataTwoImagesPath = "/content/data2/Medical mask/Medical mask/Medical Mask/
↪images"
dataTwoAnnotationPath = "/content/data2/Medical mask/Medical mask/Medical Mask/
↪annotations"
```

[ ]:
```
def dataTwoJson2DF(path):

    rowList = []

    for file in glob(os.path.join(path, "*")):


        f = open(file)
        jsonDict = json.load(f)

        filename = os.path.join(dataTwoImagesPath, jsonDict['FileName'])
        noOfAnno = jsonDict['NumOfAnno']
        Annotations = jsonDict['Annotations']
        for i in range(noOfAnno):
            classname = Annotations[i]['classname']
            bbox = Annotations[i]['BoundingBox']
            xmin = float(bbox[0])
            ymin = float(bbox[1])
            xmax = float(bbox[2])
            ymax = float(bbox[3])
```

```
            value = (
                filename, classname, xmin, ymin, xmax, ymax
            )

            boxArea = (ymax-ymin) * (xmax-xmin)

            if boxArea > 400.0:
                rowList.append(value)


    column_name = ['filename', 'class', 'xmin', 'ymin', 'xmax', 'ymax']

    return pd.DataFrame(rowList, columns=column_name)

# convert data2 to df

df2 = dataTwoJson2DF(dataTwoAnnotationPath)
```

[ ]:
```
df2.shape
```

[ ]:
```
(15178, 6)
```

[ ]:
```
df2["class"].replace({"face_with_mask_incorrect": "mask_weared_incorrect",
                          "face_with_mask": "with_mask",
                          "mask_surgical":"removeCat",
                          "mask_colorful":"removeCat",
                          "face_no_mask":"without_mask",
                          "gas_mask":"removeCat",

                          "balaclava_ski_mask":"removeCat",
                          "turban":"removeCat",
                          "hood":"removeCat",
                          "other":"removeCat",
                          "goggles":"removeCat",
                          "scarf_bandana":"removeCat",
                          "hair_net":"removeCat",
                          "sunglasses":"removeCat",
                          "hat":"removeCat",
                          "face_other_covering":"removeCat",
                          "hijab_niqab":"removeCat",
                          "face_shield":"removeCat",
                          "eyeglasses":"removeCat",
                          "helmet":"removeCat",

                          },
```

```
                          inplace=True)
```

[ ]:
```
df2 = df2[df2['class'] != "removeCat"]
```

[ ]:
```
df2.to_csv("/content/csvFolder/data2.csv", index=False)
```

[ ]:
```
df2['class'].value_counts()
```

[ ]:
```
with_mask               4167
without_mask            1562
mask_weared_incorrect    148
Name: class, dtype: int64
```

## 4 Merge Dataset

[ ]:
```
df1['class'].value_counts()
```

[ ]:
```
with_mask               1926
without_mask             337
mask_weared_incorrect     83
Name: class, dtype: int64
```

[ ]:
```
df2['class'].value_counts()
```

[ ]:
```
with_mask               4167
without_mask            1562
mask_weared_incorrect    148
Name: class, dtype: int64
```

[ ]:
```
DF = df1.append(df2, ignore_index=True)
```

[ ]:
```
DF.shape
```

[ ]:
```
(8223, 6)
```

[ ]:
```
DF['class'].value_counts()
```

[ ]:
```
with_mask               6093
without_mask            1899
mask_weared_incorrect    231
Name: class, dtype: int64
```

## 5 Import Scripts

[ ]:
```
!git clone https://github.com/pytorch/vision.git
```

```
Cloning into 'vision'…
remote: Enumerating objects: 42731, done.
remote: Counting objects: 100% (8207/8207), done.
remote: Compressing objects: 100% (2126/2126), done.
remote: Total 42731 (delta 6356), reused 7562 (delta 5921), pack-reused 34524
Receiving objects: 100% (42731/42731), 65.11 MiB | 28.59 MiB/s, done.
Resolving deltas: 100% (32907/32907), done.
```

[ ]:
```
# import a python file as a module in Notebook

!ln -s ./vision/references/detection/engine.py engine.py
!ln -s ./vision/references/detection/train.py train.py
!ln -s ./vision/references/detection/utils.py utils.py
!ln -s ./vision/references/detection/coco_utils.py coco_utils.py
!ln -s ./vision/references/detection/group_by_aspect_ratio.py␣
↪group_by_aspect_ratio.py
!ln -s ./vision/references/detection/presets.py presets.py
!ln -s ./vision/references/detection/coco_eval.py coco_eval.py
!ln -s ./vision/references/detection/transforms.py transforms.py
```

[ ]:
```
# to use pyprotocols
!pip install -q pycocotools
```

## 6 Train Eval Split

[ ]:
```
from sklearn.model_selection import train_test_split
```

[ ]:
```
filenames = DF['filename'].unique().tolist()

TRAIN_IMAGES, VALID_IMAGES = train_test_split(filenames, test_size=0.2)

len(TRAIN_IMAGES), len(VALID_IMAGES)
```

[ ]:
```
(3406, 852)
```

[ ]:
```
def get_dataframe_from_Images(DF, images):
    for i, image in enumerate(images):
        if i == 0:
            first_df = DF[DF['filename']==image]
            return_df = first_df
        else:
            temp_df = DF[DF['filename']==image]
```

```
            return_df = return_df.append(temp_df, ignore_index=True)

    return return_df

TRAIN_DF = get_dataframe_from_Images(DF, TRAIN_IMAGES)
VALID_DF = get_dataframe_from_Images(DF, VALID_IMAGES)
```

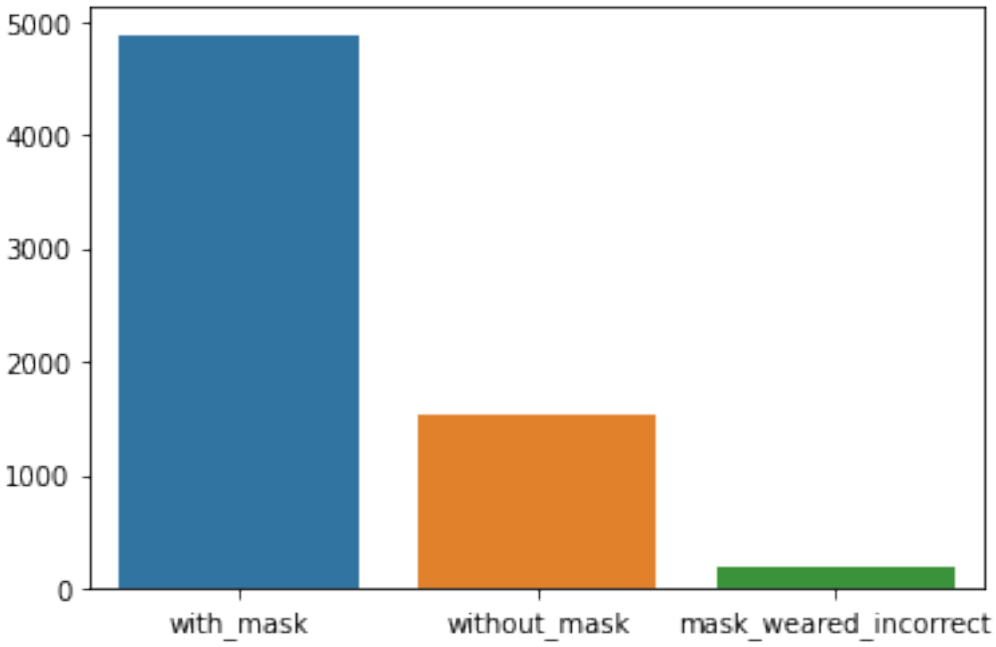
[ ]:
```
TRAIN_DF['filename'].unique().shape, VALID_DF['filename'].unique().shape
```

[ ]:
```
((3406,), (852,))
```

[ ]:
```
sns.barplot(TRAIN_DF['class'].value_counts().index, TRAIN_DF['class'].
↪value_counts().values)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:
Pass the following variables as keyword args: x, y. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f17d44cc3d0>
```

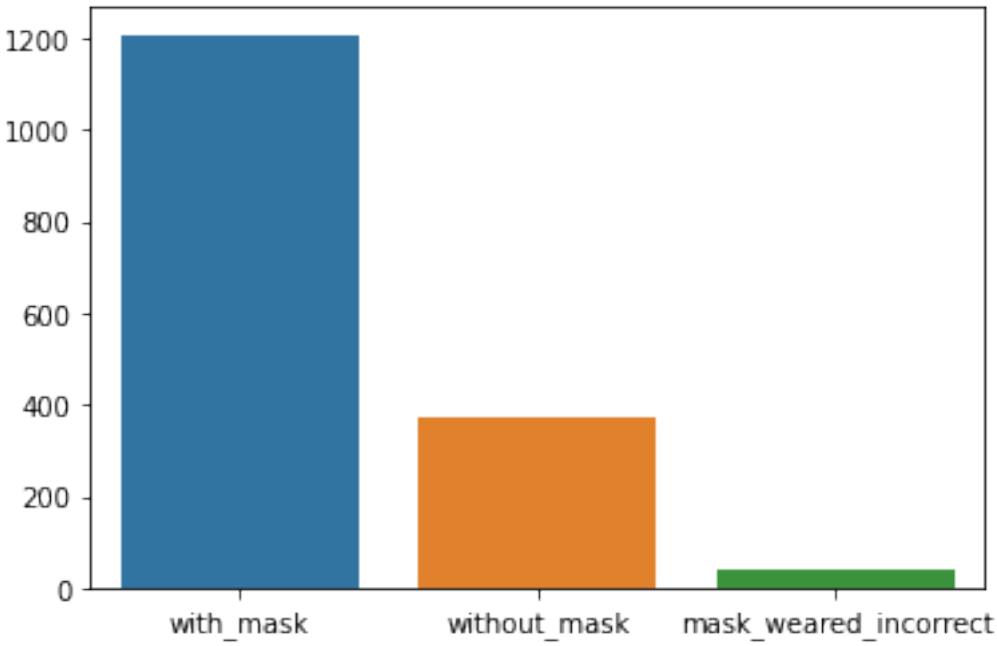



[ ]:
```
sns.barplot(VALID_DF['class'].value_counts().index, VALID_DF['class'].
↪value_counts().values)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:
```

```
Pass the following variables as keyword args: x, y. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f17d340ab10>
```




## 7 Visualize Dataset

[ ]:
```
from albumentations.pytorch.transforms import ToTensorV2
```

[ ]:
```
# A.Resize(512, 512, always_apply=True)

def get_aug_pipeline():

    transforms = []
    # transforms.append( )
    transforms.append(ToTensorV2(p=1))

    return A.Compose(
        transforms, bbox_params=A.BboxParams(format='pascal_voc',␣
↪label_fields=['labels']),
    )
```

[ ]:
```
from matplotlib import pyplot as plt
import matplotlib.patches as patches


def show_output_with_bbox(filename, bboxes, labels, transform):

    # image = plt.imread(filename)

    image = cv2.imread(filename, cv2.IMREAD_ANYCOLOR) # cv2.IMREAD_COLOR
    image = cv2.cvtColor(image, cv2.COLOR_BGRA2RGB).astype(np.float32)
    image = image/255.0

    aug_pipeline = get_aug_pipeline()
    transformed = aug_pipeline(image=image, bboxes=bboxes, labels=labels)

    img = transformed['image']
    bboxes = torch.as_tensor(transformed['bboxes'])
    bboxes = bboxes.detach().numpy()
    labels = transformed['labels']


    print(img.shape)

    img_height = img.shape[1]
    img_width = img.shape[2]

    fig, ax = plt.subplots(figsize=(5,5))

    ax.imshow(img.permute(1,2,0).numpy())

    for bbox, class_name in zip(bboxes, labels):
        xmin = bbox[0]
        ymin = bbox [1]
        width = bbox[2] - xmin
        height = bbox[3] - ymin

        # xmin, ymin, width, height = [x for x in [(xcenter-(width/2)),␣
↪(ycenter-(height/2)), width, height]]
        rect = patches.Rectangle((xmin, ymin), width, height, linewidth=1,␣
↪edgecolor='r', facecolor='none')
        ax.add_patch(rect)
        ax.text(xmin, ymin, class_name, color='w')


    plt.show()
```

```
#################################################################################

trainFiles = TRAIN_DF['filename'].unique().tolist()
trainFiles = trainFiles[:50]

for filename in trainFiles:
    records = TRAIN_DF[TRAIN_DF['filename']==filename]

    bboxes = records[['xmin', 'ymin', 'xmax', 'ymax']].values
    labels = records[['class']].values
    labels = [x[0] for x in labels]
    area = (bboxes[:,2]-bboxes[:,0]) * (bboxes[:,3]-bboxes[:,1])
    print(area)

    show_output_with_bbox(filename, bboxes, labels, get_aug_pipeline())
```

[ ]:
```
# Image Verification

trainFiles = TRAIN_DF['filename'].unique().tolist()

for filename in trainFiles:

    if filename == "/content/data1/images/maksssksksss719.png":

        records = TRAIN_DF[TRAIN_DF['filename']==filename]

        bboxes = records[['xmin', 'ymin', 'xmax', 'ymax']].values
        labels = records[['class']].values
        labels = [x[0] for x in labels]
        area = (bboxes[:,2]-bboxes[:,0]) * (bboxes[:,3]-bboxes[:,1])
        print(area)

        show_output_with_bbox(filename, bboxes, labels, get_aug_pipeline())

        break
```

```
[ 616.  540.  900. 1512.]
torch.Size([3, 226, 400])
```


## 8 Dataset and Dataloader

[ ]:
```
import random
# random.seed(7)
```

[ ]:
```
from albumentations.pytorch.transforms import ToTensorV2

# https://albumentations.ai/docs/getting_started/bounding_boxes_augmentation/
# https://albumentations.ai/docs/api_reference/core/composition/


global class_int_dict
class_int_dict = {
    "background":0,
    "with_mask":1,
    "without_mask":2,
    "mask_weared_incorrect":3
}
```

[ ]:
```
def get_geometric_aug(aug):
    return A.Compose([A.Resize(320, 320, p=1.0),
                      aug,
                      A.Resize(320, 320, p=1.0),
                      ToTensorV2(p=1)],
                     bbox_params=A.BboxParams(format='pascal_voc', min_area=0.
 ↪0, min_visibility=0.0, label_fields=['labels']))

def get_noise_aug(aug):
    return A.Compose([A.Resize(320, 320, p=1.0),
```

```
                    aug,
                    ToTensorV2(p=1)],
                   bbox_params=A.BboxParams(format='pascal_voc', min_area=0.
↪0, min_visibility=0.0, label_fields=['labels']))

def get_resize_aug():
    return A.Compose([A.Resize(320, 320, p=1.0),
                    ToTensorV2(p=1)],
                    bbox_params=A.BboxParams(format='pascal_voc', min_area=0.
↪0, min_visibility=0.0, label_fields=['labels']))
```

[ ]:
```
class FacemaskDataset(Dataset):

    def __init__(self, DF, train):

        self.train = train
        self.DF = DF
        self.filenames = self.DF['filename'].unique()

        self.resize = get_resize_aug()
        self.rotate = get_geometric_aug(A.Rotate(limit = 10, p=1.0))
        self.transpose = get_geometric_aug(A.Transpose(p=1.0))
        self.h_flip = get_geometric_aug(A.HorizontalFlip(p=1.0))
        self.rand_flip = get_geometric_aug(A.Flip(p=1.0))
        self.center_crop_400 = get_geometric_aug(A.CenterCrop(height=400,␣
↪width=400, p=1.0))
        self.crop_pad_zoom_in_10 = get_geometric_aug(A.CropAndPad(percent=-0.
↪08, p=1))
        self.crop_pad_zoom_out_10 = get_geometric_aug(A.CropAndPad(percent=0.
↪10, p=1))

        self.gaussian_blur = get_noise_aug(A.GaussianBlur(blur_limit=(5, 7),␣
↪sigma_limit=0, always_apply=True, p=1))
        self.med_blur = get_noise_aug(A.MedianBlur(blur_limit=5,␣
↪always_apply=True, p=1))
        self.glass_blur = get_noise_aug(A.GlassBlur(p=1))
        self.mult_noise = get_noise_aug(A.MultiplicativeNoise(multiplier=(1.1,␣
↪2.1), p=1))
        self.gaussian_noise = get_noise_aug(A.GaussNoise(p=1))
        self.gray = get_noise_aug(A.ToGray(p=1))
        self.rand_contrast = get_noise_aug(A.
↪RandomBrightnessContrast(brightness_limit=0.2, contrast_limit=0.7, p=1))
        self.aa_sharpen = get_noise_aug(A.Sharpen(alpha=(0.1, 1), lightness=(0.
↪8, 1.0), p=1.0))

        self.transforms = None
```

```
    def __len__(self):
        return len(self.filenames)

    def load_images_boxes(self, index):

        image_id = self.filenames[index]

        records = self.DF[self.DF['filename'] == image_id]
        img = cv2.imread(image_id, cv2.IMREAD_ANYCOLOR) # cv2.IMREAD_COLOR
        img = cv2.cvtColor(img, cv2.COLOR_BGRA2RGB).astype(np.float32)
        img = img/255.0

        bboxes = records[['xmin', 'ymin', 'xmax', 'ymax']].values.tolist()
        class_labels = records[['class']].values.reshape(1, -1)[0]
        class_labels = [class_int_dict[i] for i in class_labels]

        return img, bboxes, class_labels

    def __getitem__(self, index):


        img, bboxes, class_labels = self.load_images_boxes(index)

        rand_number = np.round(random.random(), 3)

        # print(f"random number: {rand_number}")

        if self.train:

            if rand_number >= 0.65:
                # print("iaa_sharpen")
                self.transforms = self.aa_sharpen
                transformed = self.transforms(image=img, bboxes=bboxes,␣
↪labels=class_labels)

            elif rand_number >= 0.52:
                # print("gray")
                self.transforms = self.gray
                transformed = self.transforms(image=img, bboxes=bboxes,␣
↪labels=class_labels)

            elif rand_number >= 0.39:
                # print("h_flip")
                self.transforms = self.h_flip
```

```
                transformed = self.transforms(image=img, bboxes=bboxes,␣
↪labels=class_labels)

            elif rand_number >= 0.26:
                # print("gaussian_blur")
                self.transforms = self.gaussian_blur
                transformed = self.transforms(image=img, bboxes=bboxes,␣
↪labels=class_labels)

            elif rand_number >= 0.13:
                # print("rotate")
                self.transforms = self.rotate
                transformed = self.transforms(image=img, bboxes=bboxes,␣
↪labels=class_labels)

            elif rand_number >= 0.0:
                # print("med_blur")
                self.transforms = self.med_blur
                transformed = self.transforms(image=img, bboxes=bboxes,␣
↪labels=class_labels)

        else:
            self.transforms = self.resize
            transformed = self.transforms(image=img, bboxes=bboxes,␣
↪labels=class_labels)


        img = transformed['image']
        bboxes = torch.as_tensor(transformed['bboxes'], dtype=torch.float32)

        num_objs = len(bboxes)
        iscrowd = torch.zeros((num_objs,), dtype=torch.int64)
        class_label = torch.tensor(transformed['labels'])
        iscrowd = torch.zeros((num_objs,), dtype=torch.int64)

        target = {}
        target["labels"] = class_label # tensor
        target["boxes"] = bboxes # tensor
        target['image_id'] = torch.tensor([index]) # string
        target['iscrowd'] = iscrowd
        target['area'] = (bboxes[:, 3] - bboxes[:, 1]) * (bboxes[:, 2] -␣
↪bboxes[:, 0])

        return img, target
```

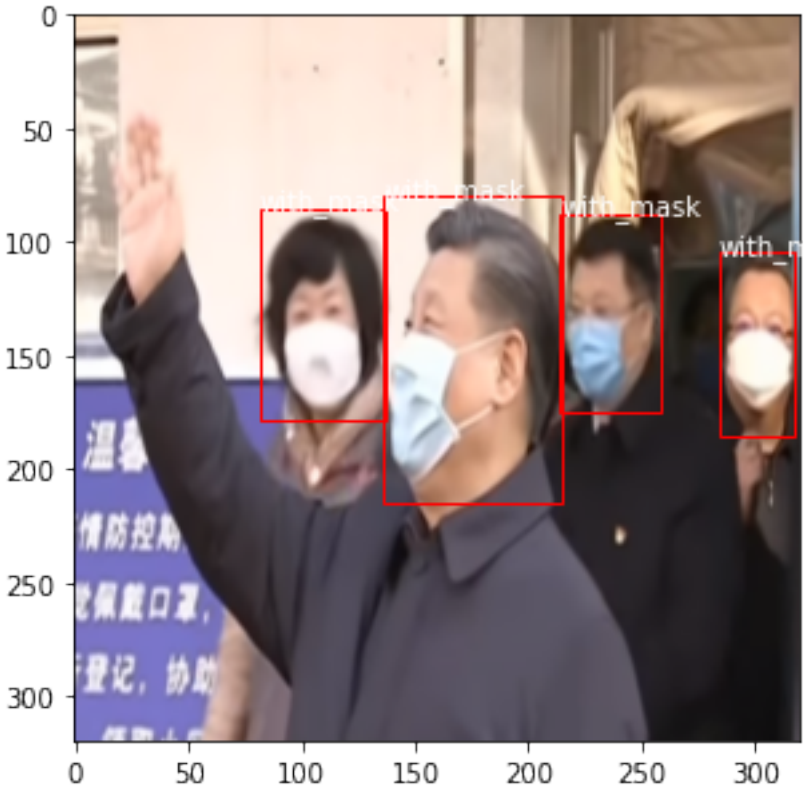
[ ]:
```
train_dataset = FacemaskDataset(DF=TRAIN_DF, train=True)

valid_dataset = FacemaskDataset(DF=VALID_DF, train=False)


def collate_fn(batch):
    return tuple(zip(*batch))

train_data_loader = DataLoader(
    train_dataset,
    batch_size = 30,
    shuffle = True,
    num_workers = 2,
    collate_fn = collate_fn
)


valid_data_loader = DataLoader(
    valid_dataset,
    batch_size = 20,
    shuffle = True,
    num_workers = 2,
    collate_fn = collate_fn
)
```

[ ]:
```
for i in range(len(train_data_loader)):
    train_dataset[i]
```

[ ]:
```
from matplotlib import pyplot as plt
import matplotlib.patches as patches


class_dict = {
    0:"background",
    1:"with_mask",
    2:"without_mask",
    3:"mask_weared_incorrect"
}


def show_output_with_bbox(img_tensor, bbox_tensor, labels_tensor):
    img = img_tensor.permute(1,2,0).detach().cpu().numpy()
    bboxes = bbox_tensor.detach().cpu().numpy()
    labels = labels_tensor.detach().cpu().numpy()


    fig, ax = plt.subplots(figsize=(5,5))
```

```
    ax.imshow(img)

    for bbox, label in zip(bboxes, labels):

        class_name = class_dict[label]
        xmin = bbox[0]
        ymin = bbox [1]
        width = bbox[2] - xmin
        height = bbox[3] - ymin

        rect = patches.Rectangle((xmin, ymin), width, height, linewidth=1,␣
↪edgecolor='r', facecolor='none')
        ax.add_patch(rect)
        ax.text(xmin, ymin, class_name, color='w')

        # print(f"class: {class_name}, area: {height*width}")


    plt.show()




################################################################################

for i in range(TRAIN_DF.shape[0]):
    image, target = train_dataset[i]
    bboxes = target['boxes']
    labels = target['labels']
    print(target['area'])

    show_output_with_bbox(image, bboxes, labels)

    if i == 10:
        break
```

```
tensor([ 5128.9180, 10574.6387,  3902.8801,  2738.6248])
```

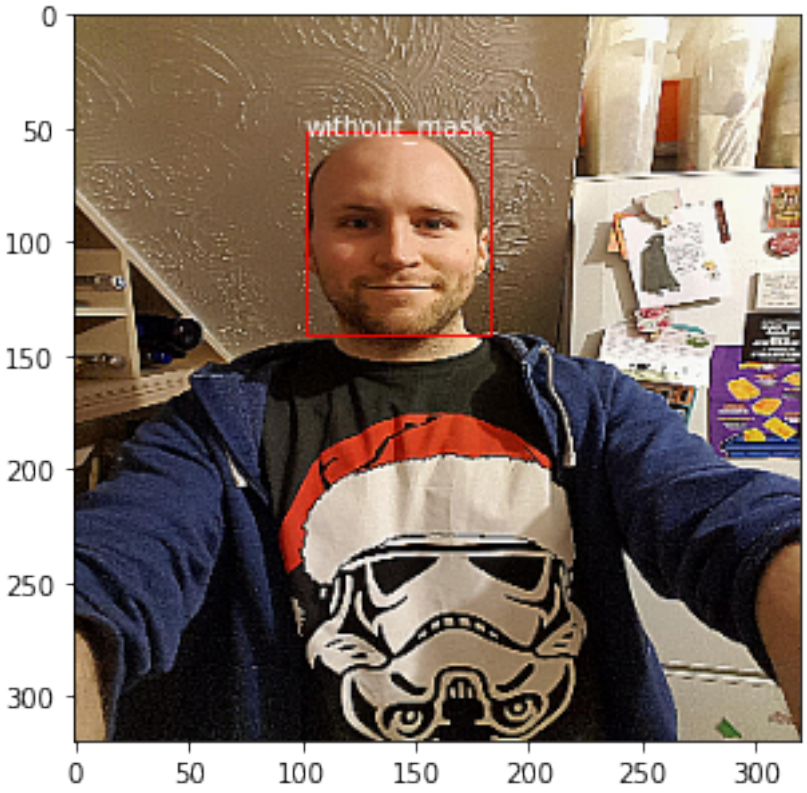



```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```

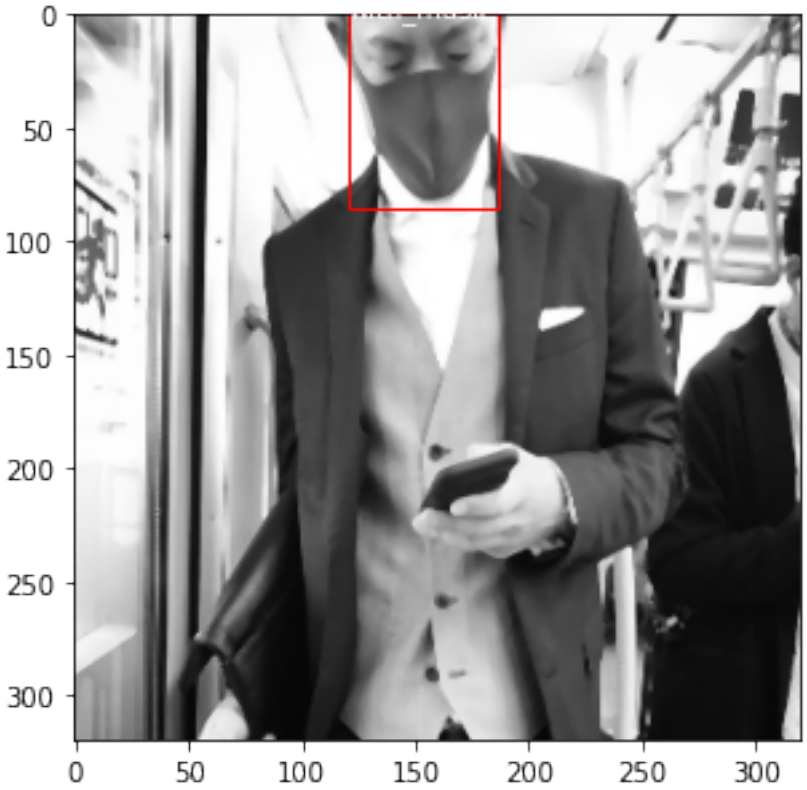
```
tensor([10915.2666,  3541.3977])
```


```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```

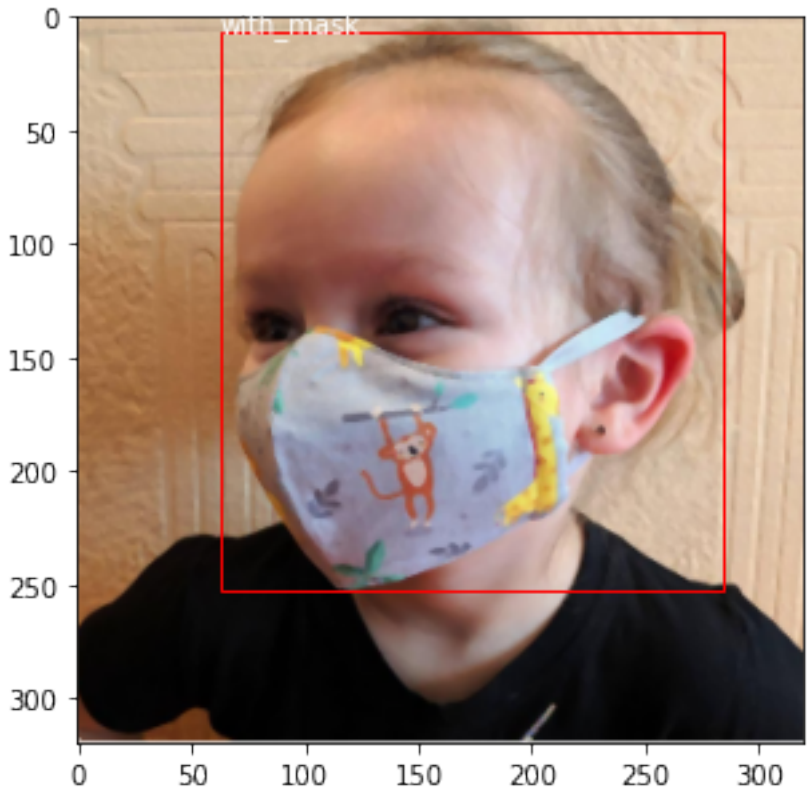
```
tensor([7242.7974])
```



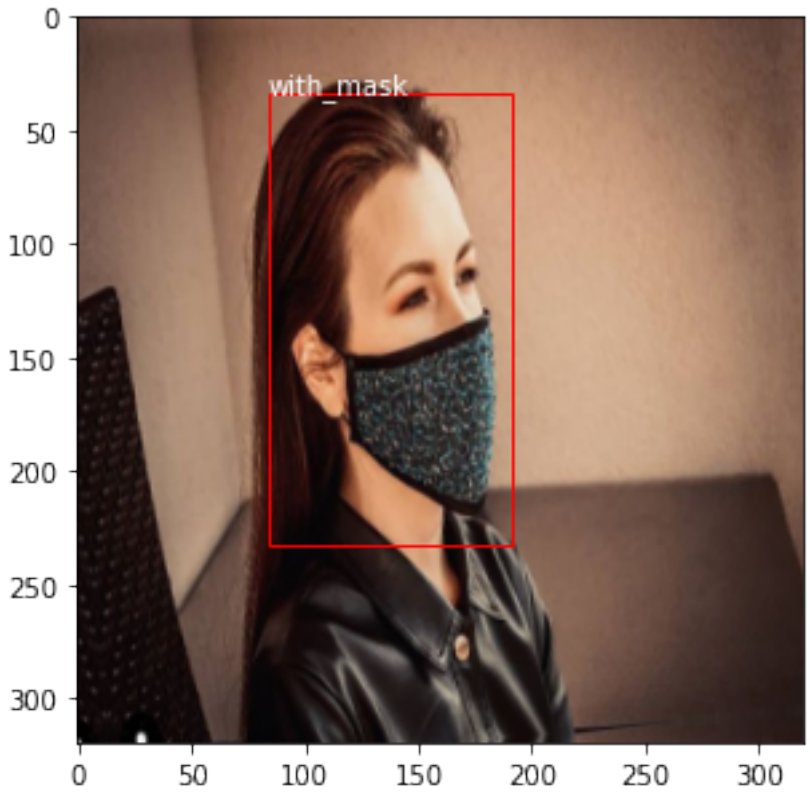

```
tensor([5629.1323])
```



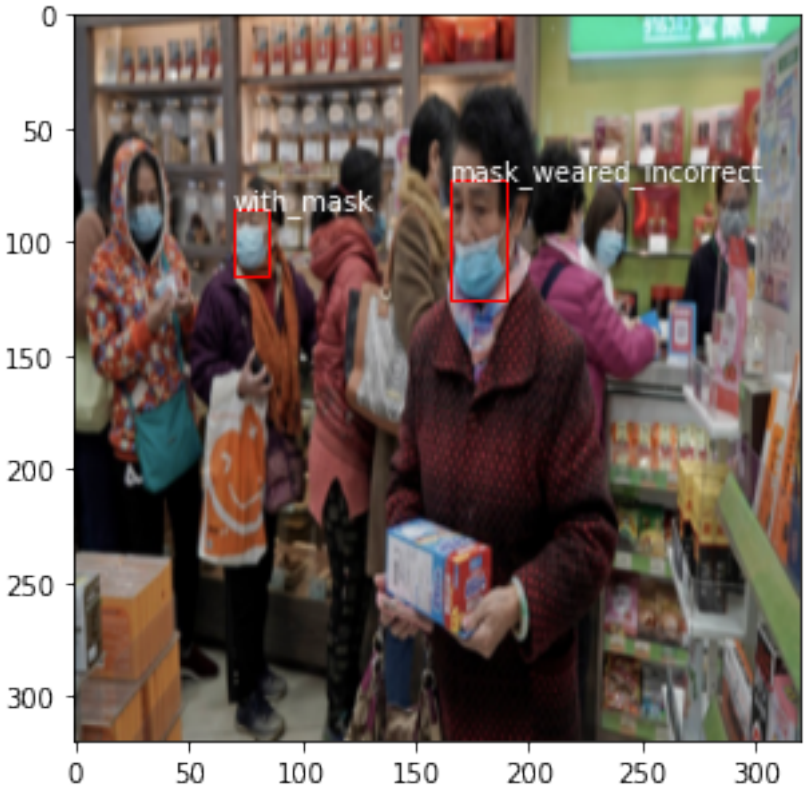

```
tensor([54700.5156])
```



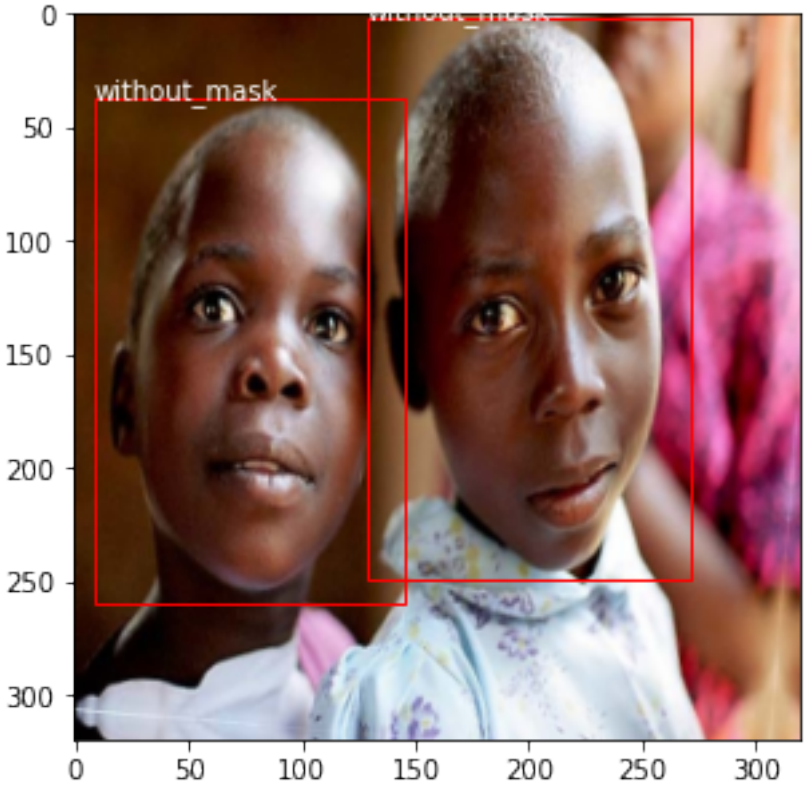

```
tensor([21445.1914])
```




```
tensor([1377.4155,  475.7522])
```

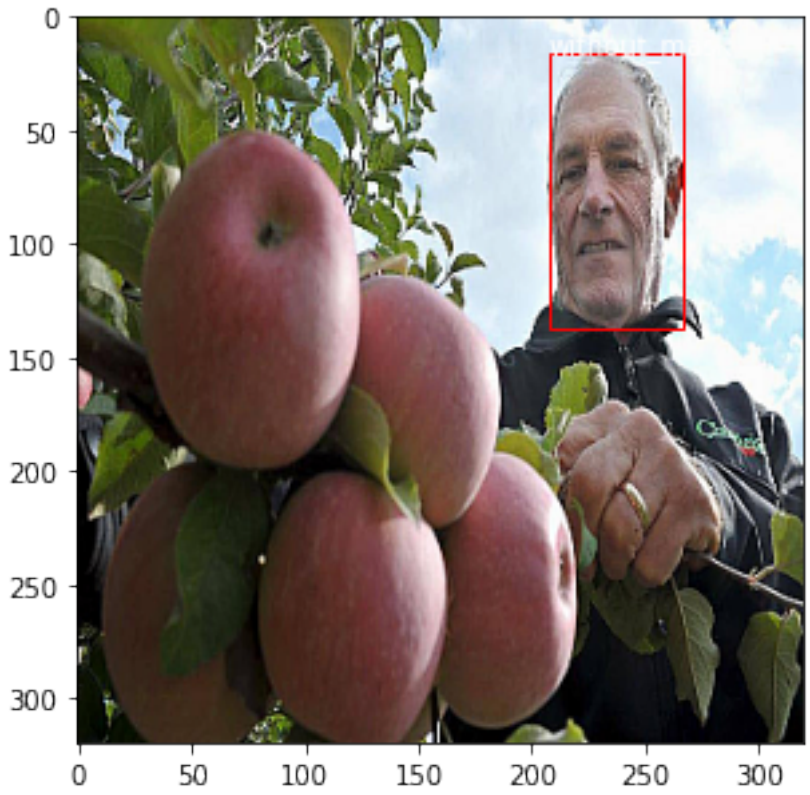



```
tensor([35510.5898, 30364.7734])
```



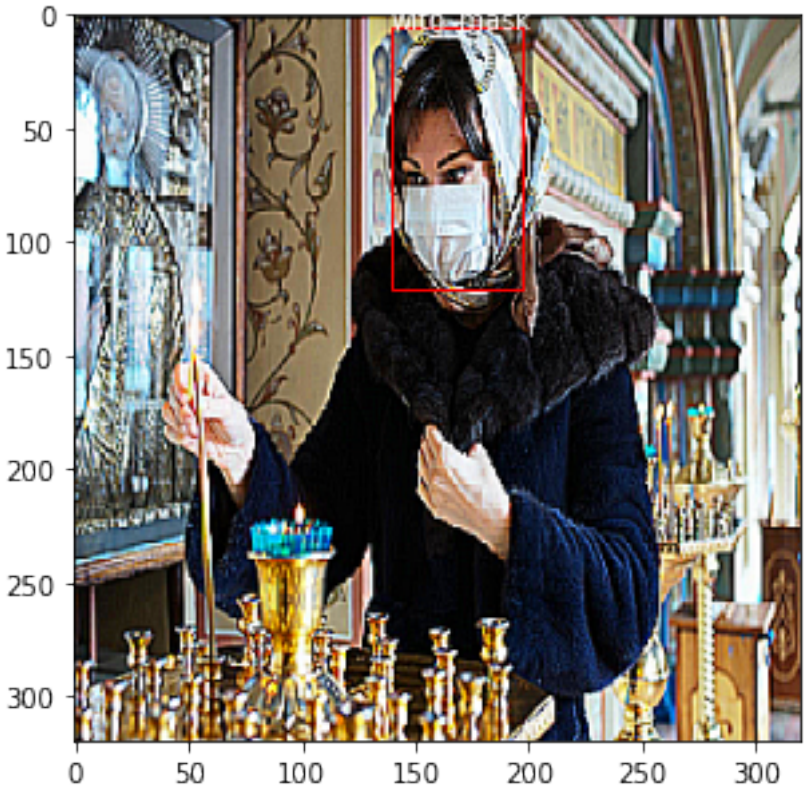

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```

```
tensor([7104.6934])
```

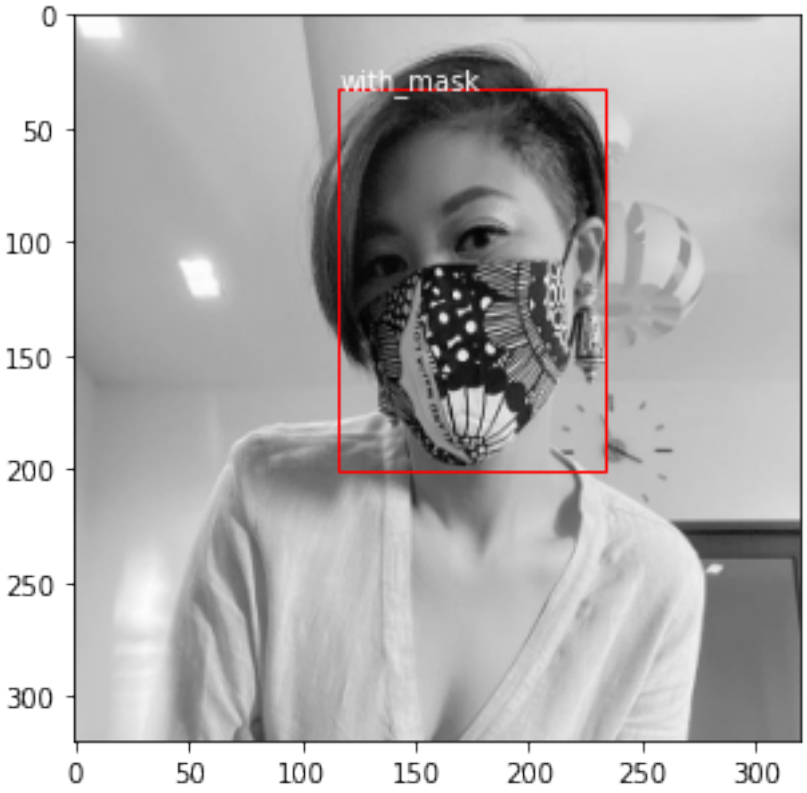



```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```

```
tensor([6751.7690])
```




```
tensor([19875.0781])
```




[ ]:
```
!nvidia-smi
```

```
Thu Oct 14 15:24:02 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.74       Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE…  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   46C    P0    28W / 250W |      0MiB / 16280MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

## 9 Device

[ ]:
```
def getDevice():
    return torch.device("cuda") if torch.cuda.is_available() else torch.
↪device("cpu")
```

## 10 Weighted Loss SSD

[ ]:
```
TRAIN_DF['class'].value_counts()
```

[ ]:
```
with_mask               5525
without_mask            1718
mask_weared_incorrect    206
Name: class, dtype: int64
```

[ ]:
```
total = 5525 + 1718 + 206
total
```

[ ]:
```
7449
```

[ ]:
```

```

[ ]:
```
global classWeight

classWeight = torch.tensor([1, 1.3, 4.3, 36.16], dtype=torch.float32,␣
↪device=getDevice())
```

[ ]:
```
from typing import Any, Dict, List, Optional, Tuple
from torch import nn
import torch.nn.functional as F
from collections import OrderedDict
import torch
import torchvision
import torchvision.models.detection._utils as det_utils
from torch import nn, Tensor
from torchvision.ops import boxes as box_ops
```

[ ]:
```
# Weighted SSD Implementation


class WeightedLossSSD(nn.Module):
    """
    Implements SSD architecture from `"SSD: Single Shot MultiBox Detector"␣
↪<https://arxiv.org/abs/1512.02325>`_.
```

```
    The input to the model is expected to be a list of tensors, each of shape␣
↪[C, H, W], one for each
    image, and should be in 0-1 range. Different images can have different␣
↪sizes but they will be resized
    to a fixed size before passing it to the backbone.

    The behavior of the model changes depending if it is in training or␣
↪evaluation mode.

    During training, the model expects both the input tensors, as well as a␣
↪targets (list of dictionary),
    containing:
        - boxes (``FloatTensor[N, 4]``): the ground-truth boxes in ``[x1, y1,␣
↪x2, y2]`` format, with
          ``0 <= x1 < x2 <= W`` and ``0 <= y1 < y2 <= H``.
        - labels (Int64Tensor[N]): the class label for each ground-truth box

    The model returns a Dict[Tensor] during training, containing the␣
↪classification and regression
    losses.

    During inference, the model requires only the input tensors, and returns␣
↪the post-processed
    predictions as a List[Dict[Tensor]], one for each input image. The fields␣
↪of the Dict are as
    follows, where ``N`` is the number of detections:

        - boxes (``FloatTensor[N, 4]``): the predicted boxes in ``[x1, y1, x2,␣
↪y2]`` format, with
          ``0 <= x1 < x2 <= W`` and ``0 <= y1 < y2 <= H``.
        - labels (Int64Tensor[N]): the predicted labels for each detection
        - scores (Tensor[N]): the scores for each detection

    Args:
        backbone (nn.Module): the network used to compute the features for the␣
↪model.
            It should contain an out_channels attribute with the list of the␣
↪output channels of
            each feature map. The backbone should return a single Tensor or an␣
↪OrderedDict[Tensor].
        anchor_generator (DefaultBoxGenerator): module that generates the␣
↪default boxes for a
            set of feature maps.
        size (Tuple[int, int]): the width and height to which images will be␣
↪rescaled before feeding them
            to the backbone.
```

```
        num_classes (int): number of output classes of the model (excluding the␣
↪background).
        image_mean (Tuple[float, float, float]): mean values used for input␣
↪normalization.
            They are generally the mean values of the dataset on which the␣
↪backbone has been trained
            on
        image_std (Tuple[float, float, float]): std values used for input␣
↪normalization.
            They are generally the std values of the dataset on which the␣
↪backbone has been trained on
        head (nn.Module, optional): Module run on top of the backbone features.␣
↪Defaults to a module containing
            a classification and regression module.
        score_thresh (float): Score threshold used for postprocessing the␣
↪detections.
        nms_thresh (float): NMS threshold used for postprocessing the␣
↪detections.
        detections_per_img (int): Number of best detections to keep after NMS.
        iou_thresh (float): minimum IoU between the anchor and the GT box so␣
↪that they can be
            considered as positive during training.
        topk_candidates (int): Number of best detections to keep before NMS.
        positive_fraction (float): a number between 0 and 1 which indicates the␣
↪proportion of positive
            proposals used during the training of the classification head. It␣
↪is used to estimate the negative to
            positive ratio.
    """
    __annotations__ = {
        'box_coder': det_utils.BoxCoder,
        'proposal_matcher': det_utils.Matcher,
    }

    def __init__(self, backbone: nn.Module, anchor_generator: torchvision.
↪models.detection.anchor_utils.DefaultBoxGenerator,
                 size: Tuple[int, int], num_classes: int,
                 image_mean: Optional[List[float]] = None, image_std:␣
↪Optional[List[float]] = None,
                 head: Optional[nn.Module] = None,
                 score_thresh: float = 0.01,
                 nms_thresh: float = 0.45,
                 detections_per_img: int = 200,
                 iou_thresh: float = 0.5,
                 topk_candidates: int = 400,
                 positive_fraction: float = 0.25):
```

```
        super().__init__()

        self.backbone = backbone

        self.anchor_generator = anchor_generator

        self.box_coder = torchvision.models.detection._utils.
↪BoxCoder(weights=(10., 10., 5., 5.))

        if head is None:
            if hasattr(backbone, 'out_channels'):
                out_channels = backbone.out_channels
            else:
                out_channels = det_utils.retrieve_out_channels(backbone, size)

            assert len(out_channels) == len(anchor_generator.aspect_ratios)

            num_anchors = self.anchor_generator.num_anchors_per_location()
            head = torchvision.models.detection.ssd.SSDHead(out_channels,␣
↪num_anchors, num_classes)
        self.head = head

        self.proposal_matcher = det_utils.SSDMatcher(iou_thresh)

        if image_mean is None:
            image_mean = [0.485, 0.456, 0.406]
        if image_std is None:
            image_std = [0.229, 0.224, 0.225]
        self.transform = torchvision.models.detection.transform.
↪GeneralizedRCNNTransform(min(size), max(size), image_mean, image_std,
                                                  size_divisible=1,␣
↪fixed_size=size)

        self.score_thresh = score_thresh
        self.nms_thresh = nms_thresh
        self.detections_per_img = detections_per_img
        self.topk_candidates = topk_candidates
        self.neg_to_pos_ratio = (1.0 - positive_fraction) / positive_fraction

        # used only on torchscript mode
        self._has_warned = False

    @torch.jit.unused
    def eager_outputs(self, losses: Dict[str, Tensor],
                      detections: List[Dict[str, Tensor]]) -> Tuple[Dict[str,␣
↪Tensor], List[Dict[str, Tensor]]]:
        if self.training:
```

```
            return losses

        return detections

    def compute_loss(self, targets: List[Dict[str, Tensor]], head_outputs:␣
↪Dict[str, Tensor], anchors: List[Tensor],
                     matched_idxs: List[Tensor]) -> Dict[str, Tensor]:
        bbox_regression = head_outputs['bbox_regression']
        cls_logits = head_outputs['cls_logits']

        # Match original targets with default boxes
        num_foreground = 0
        bbox_loss = []
        cls_targets = []
        for (targets_per_image, bbox_regression_per_image,␣
↪cls_logits_per_image, anchors_per_image,
             matched_idxs_per_image) in zip(targets, bbox_regression,␣
↪cls_logits, anchors, matched_idxs):
            # produce the matching between boxes and targets
            foreground_idxs_per_image = torch.where(matched_idxs_per_image >=␣
↪0)[0]
            foreground_matched_idxs_per_image =␣
↪matched_idxs_per_image[foreground_idxs_per_image]
            num_foreground += foreground_matched_idxs_per_image.numel()

            # Calculate regression loss
            matched_gt_boxes_per_image =␣
↪targets_per_image['boxes'][foreground_matched_idxs_per_image]
            bbox_regression_per_image =␣
↪bbox_regression_per_image[foreground_idxs_per_image, :]
            anchors_per_image = anchors_per_image[foreground_idxs_per_image, :]
            target_regression = self.box_coder.
↪encode_single(matched_gt_boxes_per_image, anchors_per_image)
            bbox_loss.append(torch.nn.functional.smooth_l1_loss(
                bbox_regression_per_image,
                target_regression,
                reduction='sum'
            ))

            # Estimate ground truth for class targets
            gt_classes_target = torch.zeros((cls_logits_per_image.size(0), ),␣
↪dtype=targets_per_image['labels'].dtype,
                                            device=targets_per_image['labels'].
↪device)
            gt_classes_target[foreground_idxs_per_image] = \
                targets_per_image['labels'][foreground_matched_idxs_per_image]
```

```
            cls_targets.append(gt_classes_target)

        bbox_loss = torch.stack(bbox_loss)
        cls_targets = torch.stack(cls_targets)

        # Calculate classification loss
        num_classes = cls_logits.size(-1)
        cls_loss = F.cross_entropy(
            cls_logits.view(-1, num_classes),
            cls_targets.view(-1),
            weight=classWeight,
            reduction='none'
        ).view(cls_targets.size())

        # Hard Negative Sampling
        foreground_idxs = cls_targets > 0
        num_negative = self.neg_to_pos_ratio * foreground_idxs.sum(1,␣
↪keepdim=True)
        # num_negative[num_negative < self.neg_to_pos_ratio] = self.
↪neg_to_pos_ratio
        negative_loss = cls_loss.clone()
        negative_loss[foreground_idxs] = -float('inf')  # use -inf to detect␣
↪positive values that creeped in the sample
        values, idx = negative_loss.sort(1, descending=True)
        # background_idxs = torch.logical_and(idx.sort(1)[1] < num_negative,␣
↪torch.isfinite(values))
        background_idxs = idx.sort(1)[1] < num_negative

        N = max(1, num_foreground)
        return {
            'bbox_regression': bbox_loss.sum() / N,
            'classification': (cls_loss[foreground_idxs].sum() +␣
↪cls_loss[background_idxs].sum()) / N,
        }

    def forward(self, images: List[Tensor],
                targets: Optional[List[Dict[str, Tensor]]] = None) ->␣
↪Tuple[Dict[str, Tensor], List[Dict[str, Tensor]]]:
        if self.training and targets is None:
            raise ValueError("In training mode, targets should be passed")

        if self.training:
            assert targets is not None
            for target in targets:
                boxes = target["boxes"]
                if isinstance(boxes, torch.Tensor):
```

```
                    if len(boxes.shape) != 2 or boxes.shape[-1] != 4:
                        raise ValueError("Expected target boxes to be a tensor"
                                         "of shape [N, 4], got {:}.".format(
                                             boxes.shape))
                else:
                    raise ValueError("Expected target boxes to be of type "
                                     "Tensor, got {:}.".format(type(boxes)))

        # get the original image sizes
        original_image_sizes: List[Tuple[int, int]] = []
        for img in images:
            val = img.shape[-2:]
            assert len(val) == 2
            original_image_sizes.append((val[0], val[1]))

        # transform the input
        images, targets = self.transform(images, targets)

        # Check for degenerate boxes
        if targets is not None:
            for target_idx, target in enumerate(targets):
                boxes = target["boxes"]
                degenerate_boxes = boxes[:, 2:] <= boxes[:, :2]
                if degenerate_boxes.any():
                    bb_idx = torch.where(degenerate_boxes.any(dim=1))[0][0]
                    degen_bb: List[float] = boxes[bb_idx].tolist()
                    raise ValueError("All bounding boxes should have positive␣
↪height and width."
                                     " Found invalid box {} for target at index␣
↪{}."
                                     .format(degen_bb, target_idx))

        # get the features from the backbone
        features = self.backbone(images.tensors)
        if isinstance(features, torch.Tensor):
            features = OrderedDict([('0', features)])

        features = list(features.values())

        # compute the ssd heads outputs using the features
        head_outputs = self.head(features)

        # create the set of anchors
        anchors = self.anchor_generator(images, features)

        losses = {}
        detections: List[Dict[str, Tensor]] = []
```

```
        if self.training:
            assert targets is not None

            matched_idxs = []
            for anchors_per_image, targets_per_image in zip(anchors, targets):
                if targets_per_image['boxes'].numel() == 0:
                    matched_idxs.append(torch.full((anchors_per_image.
↪size(0),), -1, dtype=torch.int64,
                                                   device=anchors_per_image.
↪device))
                    continue

                match_quality_matrix = box_ops.
↪box_iou(targets_per_image['boxes'], anchors_per_image)
                matched_idxs.append(self.proposal_matcher(match_quality_matrix))

            losses = self.compute_loss(targets, head_outputs, anchors,␣
↪matched_idxs)
        else:
            detections = self.postprocess_detections(head_outputs, anchors,␣
↪images.image_sizes)
            detections = self.transform.postprocess(detections, images.
↪image_sizes, original_image_sizes)

        if torch.jit.is_scripting():
            if not self._has_warned:
                warnings.warn("SSD always returns a (Losses, Detections) tuple␣
↪in scripting")
                self._has_warned = True
            return losses, detections
        return self.eager_outputs(losses, detections)

    def postprocess_detections(self, head_outputs: Dict[str, Tensor],␣
↪image_anchors: List[Tensor],
                               image_shapes: List[Tuple[int, int]]) ->␣
↪List[Dict[str, Tensor]]:
        bbox_regression = head_outputs['bbox_regression']
        pred_scores = F.softmax(head_outputs['cls_logits'], dim=-1)

        num_classes = pred_scores.size(-1)
        device = pred_scores.device

        detections: List[Dict[str, Tensor]] = []

        for boxes, scores, anchors, image_shape in zip(bbox_regression,␣
↪pred_scores, image_anchors, image_shapes):
```

```
            boxes = self.box_coder.decode_single(boxes, anchors)
            boxes = box_ops.clip_boxes_to_image(boxes, image_shape)

            image_boxes = []
            image_scores = []
            image_labels = []
            for label in range(1, num_classes):
                score = scores[:, label]

                keep_idxs = score > self.score_thresh
                score = score[keep_idxs]
                box = boxes[keep_idxs]

                # keep only topk scoring predictions
                num_topk = min(self.topk_candidates, score.size(0))
                score, idxs = score.topk(num_topk)
                box = box[idxs]

                image_boxes.append(box)
                image_scores.append(score)
                image_labels.append(torch.full_like(score, fill_value=label,␣
↪dtype=torch.int64, device=device))

            image_boxes = torch.cat(image_boxes, dim=0)
            image_scores = torch.cat(image_scores, dim=0)
            image_labels = torch.cat(image_labels, dim=0)

            # non-maximum suppression
            keep = box_ops.batched_nms(image_boxes, image_scores, image_labels,␣
↪self.nms_thresh)
            keep = keep[:self.detections_per_img]

            detections.append({
                'boxes': image_boxes[keep],
                'scores': image_scores[keep],
                'labels': image_labels[keep],
            })
        return detections
```

[ ]:
```
def _xavier_normal_init(conv: nn.Module):
    for layer in conv.modules():
        if isinstance(layer, nn.Conv2d):
            torch.nn.init.xavier_normal_(layer.weight)
            if layer.bias is not None:
                torch.nn.init.constant_(layer.bias, 0.0)

def _kaiming_normal_init(conv: nn.Module):
```

```
    for layer in conv.modules():
        if isinstance(layer, nn.Conv2d):
            torch.nn.init.kaiming_normal_(layer.weight)
            if layer.bias is not None:
                torch.nn.init.constant_(layer.bias, 0.0)


# MobileNetV3 Custom build for facemask classification
class CustomMV3(nn.Module):
    def __init__(self, base):
        super().__init__()

        self.features = base.features
        self.avgPool = base.avgpool
        self.flatten = nn.Flatten()
        self.classfier = nn.Sequential(

            nn.Linear(960, 1280, bias=True),
            nn.Hardswish(inplace=False),
            nn.Dropout(p=0.2, inplace=True),
            nn.Linear(in_features=1280, out_features=3, bias=True)

        )


    def forward(self, x):
        out = self.features(x)
        out = self.avgPool(out)
        out = self.flatten(out)
        out = self.classfier(out)
        return out

class SSDFeatureExtractorMobilenetV3(torch.nn.Module):
    """
    create 6: feature extraction layers.
    1. from base model itself from middle layers, [ Take from middle, it;s a␣
 ↪experiment.]
    2. get last layer, add some conv layers to it.
    3. add 4 custom sequential layer(conv, relu, conv, relu)

    Total: 6 number of feature will be @param: returned OrderDict("0":
 ↪feature_map)


    """

    def __init__(self, base):
```

```
        super(SSDFeatureExtractorMobilenetV3, self).__init__()

        self.featuresOneFromBase = nn.Sequential(
            *base[:5]
        )

        self.featuresTwoFromBase = nn.Sequential(
            *base[5:10]  # until InvertedResidual 9
        )

        self.featuresThreeFromBase = nn.Sequential(
            *base[10:]  # until InvertedResidual 10 --> 16
        )

        """
        VGG16 bakbone has put - 4 extra blocks plus 1 for high res image
        MobileNetV3 backbone has put - 3 extra blocks plus 1 for 1024 res image

        [(n+2p-k) / (s)] + 1

        keeping HxW = 16x16, channels: decreasing

        """

        fc1 = nn.Sequential(
            nn.Conv2d(in_channels=960, out_channels=64, kernel_size=1,␣
↪padding=0, stride=1),  # FC6 with atrous
            nn.BatchNorm2d(num_features=64),
            nn.Hardswish(inplace=True),

            nn.Conv2d(in_channels=64, out_channels=128, kernel_size=1,␣
↪padding=0, stride=1),  # FC6 with atrous
            nn.BatchNorm2d(num_features=128),
            nn.Hardswish(inplace=True),

            nn.Conv2d(in_channels=128, out_channels=256, kernel_size=1,␣
↪padding=0, stride=1),  # FC6 with atrous
            nn.BatchNorm2d(num_features=256),

            )

        fc2 = nn.Sequential(
                nn.Conv2d(256, 32, kernel_size=1, padding=0, stride=2),
                nn.BatchNorm2d(num_features=32),
                nn.Hardswish(inplace=True),

                nn.Conv2d(32, 64, kernel_size=1, padding=0, stride=2),
```

```
            nn.BatchNorm2d(num_features=64),
            nn.Hardswish(inplace=True),

            nn.Conv2d(64, 128, kernel_size=1, padding=0, stride=2),
            nn.BatchNorm2d(num_features=128),
        )
        fc3 = nn.Sequential(
            nn.Conv2d(128, 64, kernel_size=1, padding=0, stride=2),
            nn.BatchNorm2d(num_features=64),
            nn.Hardswish(inplace=True),
        )
        fc4 = nn.Sequential(
            nn.Conv2d(64, 32, kernel_size=1, padding=0, stride=1),
            nn.BatchNorm2d(num_features=32),
            nn.Hardswish(inplace=True),
        )
        fc5 = nn.Sequential(
            nn.Conv2d(32, 16, kernel_size=1, padding=0, stride=2),
            nn.BatchNorm2d(num_features=16),
            nn.Hardswish(inplace=True),
        )

        _xavier_normal_init(fc1)
        _xavier_normal_init(fc2)
        _xavier_normal_init(fc3)
        _xavier_normal_init(fc4)
        _xavier_normal_init(fc5)

        self.extra = nn.ModuleList([ fc1, fc2, fc3, fc4, fc5])

        self.scale_weight = nn.Parameter(torch.ones(960) * 200)

    def forward(self, x):
        output = []

        outOne = self.featuresOneFromBase(x)
        output.append(outOne)

        out = self.featuresTwoFromBase(outOne)
        output.append(out)

        out = self.featuresThreeFromBase(out)
        output.append(out)

        for i, block in enumerate(self.extra):
            out = block(out)
            output.append(out)
```

```
        return OrderedDict([(str(i), v) for i, v in enumerate(output)])


def loadMobileNetV3(path, model):

    model.load_state_dict(torch.load(path, map_location=getDevice()))
    model.train()
    return model


def get_ssd_backbone():

    pretrainedMobilenetV3 = torchvision.models.
 ↪mobilenet_v3_large(pretrained=True)

    # pretrainedMobilenetV3 = CustomMV3(pretrainedMobilenetV3)
    # path = "/content/gdrive/MyDrive/facemask_detection/save_model/
 ↪classificationMobileNetV3_512x512.pth"
    # pretrainedMobilenetV3 = loadMobileNetV3(path, pretrainedMobilenetV3)

    pretrainedMobilenetV3 = pretrainedMobilenetV3.features

    for i, b in enumerate(pretrainedMobilenetV3):
        for param in b.parameters():
            param.requires_grad = True

    # TODO: freeze initial layers of MobileNetV3
    return SSDFeatureExtractorMobilenetV3(pretrainedMobilenetV3)


def get_ssd_model(num_classes, size):

    backbone = get_ssd_backbone()
    # total no. of features = 5
    # for highres features = 6, change aspect ration numbers accordingly
    anchor_generator = torchvision.models.detection.anchor_utils.
 ↪DefaultBoxGenerator([[2, 3],
 ↪       [2, 3],
 ↪       [2, 3],
 ↪       [2, 3],
```

```
↪      [2, 3],
↪      [2, 3],
↪      [2, 3], [2]],
                                          scales=[0.07, 0.15, 0.33, 0.51, 0.
↪69, 0.87, 1.05, 1.3, 1.5])

   print(f"number of anchors per grid for each features: {anchor_generator.
↪num_anchors_per_location()}")



   defaults = {
       # Rescale the input in a way compatible to the backbone


       "image_mean": [0.48235, 0.45882, 0.40784],
       "image_std": [1.0 / 255.0, 1.0 / 255.0, 1.0 / 255.0]
       # undo the 0-1 scaling of toTensor
   }

   kwargs = {**defaults}
   SSD_CUSTOM = WeightedLossSSD(backbone, anchor_generator, (size, size),␣
↪num_classes, **kwargs)

   return SSD_CUSTOM
```

[ ]:
```
SSD_MODEL = get_ssd_model(num_classes=4, size=512)
```

```
Downloading:
"https://download.pytorch.org/models/mobilenet_v3_large-8738ca79.pth" to
/root/.cache/torch/hub/checkpoints/mobilenet_v3_large-8738ca79.pth

  0%|          | 0.00/21.1M [00:00<?, ?B/s]

number of anchors per grid for each features: [6, 6, 6, 6, 6, 6, 6, 4]
```

[ ]:
```
SSD_MODEL
```

## 11 SSD MobileNet Inbuilt

[ ]:
```
SSD_MODEL = torchvision.models.detection.
↪ssdlite320_mobilenet_v3_large(pretrained=True)
```

## 12 Train (Inbuilt SSD)

[ ]:
```
device = getDevice()
device
```

[ ]:
```
device(type='cuda')
```

[ ]:
```
from engine import train_one_epoch, evaluate
import utils

if torch.cuda.is_available():
    torch.cuda.empty_cache()

print(f"Training on {device}")
SSD_MODEL.to(device)

# construct an optimizer
params = [p for p in SSD_MODEL.parameters() if p.requires_grad]
optimizer = torch.optim.SGD(params, lr=0.01, momentum=0.9, weight_decay=0.0005)
# optimizer = torch.optim.Adam(params=params, lr=0.01, betas=(0.9, 0.009))

lr_scheduler_step_lr = torch.optim.lr_scheduler.StepLR(optimizer,
                                                       step_size=200,
                                                       gamma=0.7,)

lr_scheduler_cosine_annealing = torch.optim.lr_scheduler.
 ↪CosineAnnealingLR(optimizer, T_max=10)

# https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.
 ↪ReduceLROnPlateau.html
lr_scheduler_platau_reduce = torch.optim.lr_scheduler.
 ↪ReduceLROnPlateau(optimizer, 'min', factor=0.01, patience=2)


num_epochs = 200

##########################################################################################

loss_list = []

for epoch in range(num_epochs):
    # train for one epoch, printing every 10 iterations
    # logger = train_one_epoch(SSD_MODEL, optimizer, train_data_loader, device,␣
 ↪epoch, print_freq=100)
    # TRAINING BATCH OF DATA
    epoch_loss = []
    SSD_MODEL.train()
```

```
    for i, data in enumerate(train_data_loader):

      # data of each batch
      images, targets = data

      # copy data to cuda
      inputs = list(image.to(device) for image in images)
      targets = [{k: v.to(device) for k, v in t.items()} for t in targets]

      # set gradients to 0
      optimizer.zero_grad()

      # pass input to model
      loss_dict = SSD_MODEL(inputs, targets)

      # loss
      losses = sum(loss for loss in loss_dict.values())
      epoch_loss.append(losses.item())

      # backprop
      losses.backward()

      # update weights
      optimizer.step()
      #------==========-----#



    # update the learning rate
    # lr_scheduler_step_lr.step()

    # Epoch end
    train_loss_epoch = np.mean(epoch_loss)
    print("\n")
    print(f"epoch: {epoch}, training loss: {train_loss_epoch}, lr: {optimizer.
↪param_groups[0]['lr']} ") # shows every training loss and parameters
    loss_list.append(train_loss_epoch)

    if epoch > 100:
        if loss_list[epoch] > loss_list[epoch-10]:

            torch.save({
            'epoch': epoch,
            'model_state_dict': SSD_MODEL.state_dict(),
            'optimizer_state_dict': optimizer.state_dict(),
            }, "/content/gdrive/MyDrive/facemask_detection/save_model/
↪facemaskDetectionSSD_320x320_38MAP.pth")
```

```
            break


    # evaluate on the test dataset
    coco_evaluater = evaluate(SSD_MODEL, valid_data_loader, device=device)

    # reduce lr on platau, after validation
    # lr_scheduler_platau_reduce.step(train_loss_epoch)

    print(coco_evaluater.coco_eval)



    torch.save({
            'epoch': epoch,
            'model_state_dict': SSD_MODEL.state_dict(),
            'optimizer_state_dict': optimizer.state_dict(),
            }, "/content/gdrive/MyDrive/facemask_detection/save_model/
↪facemaskDetectionSSD_320x320_38MAP.pth")

    if epoch > 30:
        optimizer.param_groups[0]['lr'] = 0.001 * 0.1
```

```
Training on cuda
```

```
Exception ignored in: <function _MultiProcessingDataLoaderIter.__del__ at
0x7f698179b050>
Traceback (most recent call last):
  File "/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py",
line 1328, in __del__
    self._shutdown_workers()
  File "/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py",
line 1320, in _shutdown_workers
    if w.is_alive():
  File "/usr/lib/python3.7/multiprocessing/process.py", line 151, in is_alive
    assert self._parent_pid == os.getpid(), 'can only test a child process'
AssertionError: can only test a child process
Exception ignored in: <function _MultiProcessingDataLoaderIter.__del__ at
0x7f698179b050>
Traceback (most recent call last):
  File "/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py",
line 1328, in __del__
    self._shutdown_workers()
  File "/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py",
line 1320, in _shutdown_workers
    if w.is_alive():
```

```
  File "/usr/lib/python3.7/multiprocessing/process.py", line 151, in is_alive
    assert self._parent_pid == os.getpid(), 'can only test a child process'
AssertionError: can only test a child process



epoch: 0, training loss: 3.186100948601961, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:49  model_time: 1.1894 (1.1894)  evaluator_time:
0.2984 (0.2984)  time: 2.2656  data: 0.7615  max mem: 5550
Test:  [21/22]  eta: 0:00:01  model_time: 0.7167 (0.7430)  evaluator_time:
0.1709 (0.2303)  time: 0.9663  data: 0.0672  max mem: 5550
Test: Total time: 0:00:24 (1.1024 s / it)
Averaged stats: model_time: 0.7167 (0.7430)  evaluator_time: 0.1709 (0.2303)
Accumulating evaluation results…
DONE (t=0.76s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.246
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.433
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.253
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.025
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.222
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.453
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.270
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.410
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.452
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.137
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.457
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.692
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735b8650>}



epoch: 1, training loss: 2.359612367115915, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:04  model_time: 1.2637 (1.2637)  evaluator_time:
0.3787 (0.3787)  time: 2.9462  data: 1.2812  max mem: 5550
Test:  [21/22]  eta: 0:00:01  model_time: 0.6775 (0.7108)  evaluator_time:
0.1803 (0.2358)  time: 0.9811  data: 0.0604  max mem: 5550
Test: Total time: 0:00:23 (1.0833 s / it)
Averaged stats: model_time: 0.6775 (0.7108)  evaluator_time: 0.1803 (0.2358)
Accumulating evaluation results…
DONE (t=0.81s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.293
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.493
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.321
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.053
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.265
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.521
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.276
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.447
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.482
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.181
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.492
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.700
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962ade050>}


epoch: 2, training loss: 2.0934839230030775, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:11  model_time: 1.4479 (1.4479)  evaluator_time:
0.5906 (0.5906)  time: 3.2723  data: 1.2170  max mem: 5550
Test:  [21/22]  eta: 0:00:01  model_time: 0.6975 (0.7349)  evaluator_time:
0.1850 (0.2524)  time: 0.9968  data: 0.0588  max mem: 5550
Test: Total time: 0:00:24 (1.1180 s / it)
Averaged stats: model_time: 0.6975 (0.7349)  evaluator_time: 0.1850 (0.2524)
Accumulating evaluation results…
DONE (t=0.92s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.292
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.517
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.303
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.044
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.282
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.495
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.310
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.465
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.501
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.204
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.504
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.732
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6972344350>}


epoch: 3, training loss: 1.9617709023877978, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:43  model_time: 0.9125 (0.9125)  evaluator_time:
0.2731 (0.2731)  time: 1.9692  data: 0.7736  max mem: 5550
Test:  [21/22]  eta: 0:00:01  model_time: 0.6891 (0.7058)  evaluator_time:
0.2238 (0.2531)  time: 1.0332  data: 0.0652  max mem: 5550
Test: Total time: 0:00:23 (1.0731 s / it)
Averaged stats: model_time: 0.6891 (0.7058)  evaluator_time: 0.2238 (0.2531)
```

```
Accumulating evaluation results…
DONE (t=0.83s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.302
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.526
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.300
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.047
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.278
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.537
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.293
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.455
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.490
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.221
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.497
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.689
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697234fe10>}


epoch: 4, training loss: 1.8344742273911834, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:56  model_time: 0.9836 (0.9836)  evaluator_time:
0.2860 (0.2860)  time: 2.5598  data: 1.2804  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5733 (0.5983)  evaluator_time:
0.1836 (0.2425)  time: 0.8733  data: 0.0603  max mem: 5550
Test: Total time: 0:00:21 (0.9711 s / it)
Averaged stats: model_time: 0.5733 (0.5983)  evaluator_time: 0.1836 (0.2425)
Accumulating evaluation results…
DONE (t=0.69s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.310
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.531
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.299
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.062
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.306
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.501
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.294
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.435
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.460
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.222
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.461
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.647
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6972395410>}


epoch: 5, training loss: 1.71666428912431, lr: 0.01
creating index…
index created!
```

```
Test:  [ 0/22]  eta: 0:01:07  model_time: 1.0615 (1.0615)  evaluator_time:
0.6913 (0.6913)  time: 3.0752  data: 1.3053  max mem: 5550
Test:  [21/22]  eta: 0:00:01  model_time: 0.6565 (0.6466)  evaluator_time:
0.2036 (0.2584)  time: 0.9278  data: 0.0569  max mem: 5550
Test: Total time: 0:00:22 (1.0420 s / it)
Averaged stats: model_time: 0.6565 (0.6466)  evaluator_time: 0.2036 (0.2584)
Accumulating evaluation results…
DONE (t=0.82s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.328
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.558
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.340
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.056
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.324
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.535
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.313
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.482
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.519
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.264
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.542
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.680
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6972395590>}


epoch: 6, training loss: 1.6804218059405684, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:50  model_time: 0.9336 (0.9336)  evaluator_time:
0.4453 (0.4453)  time: 2.2803  data: 0.8815  max mem: 5550
Test:  [21/22]  eta: 0:00:01  model_time: 0.6100 (0.6563)  evaluator_time:
0.2080 (0.2549)  time: 0.9292  data: 0.0608  max mem: 5550
Test: Total time: 0:00:22 (1.0271 s / it)
Averaged stats: model_time: 0.6100 (0.6563)  evaluator_time: 0.2080 (0.2549)
Accumulating evaluation results…
DONE (t=0.80s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.339
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.576
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.355
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.328
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.334
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.489
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.527
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.239
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.530
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.748
```

```
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697234f5d0>}


epoch: 7, training loss: 1.6146570211276412, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:21  model_time: 0.9658 (0.9658)  evaluator_time:
0.6012 (0.6012)  time: 3.6845  data: 2.0914  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5532 (0.5952)  evaluator_time:
0.2232 (0.2335)  time: 0.8651  data: 0.0610  max mem: 5550
Test: Total time: 0:00:21 (0.9977 s / it)
Averaged stats: model_time: 0.5532 (0.5952)  evaluator_time: 0.2232 (0.2335)
Accumulating evaluation results…
DONE (t=0.74s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.313
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.530
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.334
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.072
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.284
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.539
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.299
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.466
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.498
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.225
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.497
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.719
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962ade0d0>}


epoch: 8, training loss: 1.5138298412784934, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:01  model_time: 0.5983 (0.5983)  evaluator_time:
0.3235 (0.3235)  time: 2.8097  data: 1.8782  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5724 (0.5750)  evaluator_time:
0.2098 (0.2449)  time: 0.8749  data: 0.0617  max mem: 5550
Test: Total time: 0:00:21 (0.9779 s / it)
Averaged stats: model_time: 0.5724 (0.5750)  evaluator_time: 0.2098 (0.2449)
Accumulating evaluation results…
DONE (t=0.77s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.330
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.565
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.358
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.048
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.296
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.592
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.308
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.457
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.502
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.175
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.535
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.700
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6973569ad0>}


epoch: 9, training loss: 1.490667070262134, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:04  model_time: 0.5753 (0.5753)  evaluator_time:
0.3841 (0.3841)  time: 2.9528  data: 1.9804  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5953 (0.5715)  evaluator_time:
0.2564 (0.2591)  time: 0.8890  data: 0.0608  max mem: 5550
Test: Total time: 0:00:21 (0.9929 s / it)
Averaged stats: model_time: 0.5953 (0.5715)  evaluator_time: 0.2564 (0.2591)
Accumulating evaluation results…
DONE (t=0.80s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.338
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.595
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.337
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.062
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.318
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.591
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.306
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.453
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.482
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.231
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.455
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.731
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962784dd0>}


epoch: 10, training loss: 1.4149223840795457, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:53  model_time: 1.0953 (1.0953)  evaluator_time:
0.5397 (0.5397)  time: 2.4366  data: 0.7841  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5084 (0.5634)  evaluator_time:
0.2272 (0.2566)  time: 0.8336  data: 0.0664  max mem: 5550
Test: Total time: 0:00:20 (0.9397 s / it)
Averaged stats: model_time: 0.5084 (0.5634)  evaluator_time: 0.2272 (0.2566)
Accumulating evaluation results…
DONE (t=0.74s).
IoU metric: bbox
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.340
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.567
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.366
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.057
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.342
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.556
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.320
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.481
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.509
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.238
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.492
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.754
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697219fe90>}


epoch: 11, training loss: 1.3522069291211665, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:53  model_time: 1.0701 (1.0701)  evaluator_time:
0.5136 (0.5136)  time: 2.4098  data: 0.8082  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5842 (0.6257)  evaluator_time:
0.1885 (0.2493)  time: 0.8945  data: 0.0642  max mem: 5550
Test: Total time: 0:00:21 (0.9915 s / it)
Averaged stats: model_time: 0.5842 (0.6257)  evaluator_time: 0.1885 (0.2493)
Accumulating evaluation results…
DONE (t=0.81s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.347
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.579
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.363
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.072
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.331
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.580
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.333
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.491
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.538
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.239
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.561
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.731
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69711c2210>}


epoch: 12, training loss: 1.3577878116630018, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:46  model_time: 0.9629 (0.9629)  evaluator_time:
0.2134 (0.2134)  time: 2.1229  data: 0.9285  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5563 (0.6085)  evaluator_time:
```

```
0.2072 (0.2530)  time: 0.9142  data: 0.0650  max mem: 5550
Test: Total time: 0:00:21 (0.9825 s / it)
Averaged stats: model_time: 0.5563 (0.6085)  evaluator_time: 0.2072 (0.2530)
Accumulating evaluation results…
DONE (t=0.79s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.349
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.573
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.382
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.061
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.344
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.574
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.319
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.489
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.540
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.254
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.575
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.701
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69626d9110>}


epoch: 13, training loss: 1.309306944720447, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:43  model_time: 0.7828 (0.7828)  evaluator_time:
0.4282 (0.4282)  time: 1.9839  data: 0.7570  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4709 (0.5370)  evaluator_time:
0.2094 (0.2459)  time: 0.8389  data: 0.0681  max mem: 5550
Test: Total time: 0:00:19 (0.9010 s / it)
Averaged stats: model_time: 0.4709 (0.5370)  evaluator_time: 0.2094 (0.2459)
Accumulating evaluation results…
DONE (t=0.65s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.297
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.509
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.309
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.062
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.277
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.510
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.311
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.463
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.492
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.206
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.492
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.720
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69661bc750>}
```

```
epoch: 14, training loss: 1.2380990898236632, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:50  model_time: 0.4606 (0.4606)  evaluator_time:
0.3363 (0.3363)  time: 2.2732  data: 1.4666  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5109 (0.5349)  evaluator_time:
0.2191 (0.2518)  time: 0.8496  data: 0.0627  max mem: 5550
Test: Total time: 0:00:20 (0.9286 s / it)
Averaged stats: model_time: 0.5109 (0.5349)  evaluator_time: 0.2191 (0.2518)
Accumulating evaluation results…
DONE (t=0.73s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.315
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.543
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.323
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.055
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.301
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.541
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.309
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.460
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.495
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.229
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.478
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.736
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f696f1d5bd0>}


epoch: 15, training loss: 1.2387789157219231, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:47  model_time: 0.8071 (0.8071)  evaluator_time:
0.4555 (0.4555)  time: 2.1471  data: 0.8741  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4823 (0.5552)  evaluator_time:
0.1883 (0.2603)  time: 0.8431  data: 0.0641  max mem: 5550
Test: Total time: 0:00:20 (0.9317 s / it)
Averaged stats: model_time: 0.4823 (0.5552)  evaluator_time: 0.1883 (0.2603)
Accumulating evaluation results…
DONE (t=0.76s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.364
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.601
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.413
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.074
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.349
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.612
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.330
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.527
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.257
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.517
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.758
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69628efe10>}


epoch: 16, training loss: 1.2595119569450617, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:50  model_time: 1.0183 (1.0183)  evaluator_time:
0.3202 (0.3202)  time: 2.3113  data: 0.9556  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5156 (0.5521)  evaluator_time:
0.3000 (0.2827)  time: 0.8773  data: 0.0689  max mem: 5550
Test: Total time: 0:00:21 (0.9586 s / it)
Averaged stats: model_time: 0.5156 (0.5521)  evaluator_time: 0.3000 (0.2827)
Accumulating evaluation results…
DONE (t=0.79s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.345
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.585
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.348
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.072
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.350
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.542
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.327
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.502
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.532
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.266
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.521
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.759
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697234f090>}


epoch: 17, training loss: 1.170159401372075, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:05  model_time: 0.5757 (0.5757)  evaluator_time:
0.2992 (0.2992)  time: 2.9855  data: 2.1010  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5201 (0.5638)  evaluator_time:
0.2068 (0.2241)  time: 0.8311  data: 0.0587  max mem: 5550
Test: Total time: 0:00:20 (0.9539 s / it)
Averaged stats: model_time: 0.5201 (0.5638)  evaluator_time: 0.2068 (0.2241)
Accumulating evaluation results…
DONE (t=0.69s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.357
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.602
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.375
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.072
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.353
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.588
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.319
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.504
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.523
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.233
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.528
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.742
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6961fef2d0>}


epoch: 18, training loss: 1.1530232704244554, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:51  model_time: 0.8195 (0.8195)  evaluator_time:
0.5892 (0.5892)  time: 2.3456  data: 0.9192  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5223 (0.5672)  evaluator_time:
0.2176 (0.2682)  time: 0.8886  data: 0.0665  max mem: 5550
Test: Total time: 0:00:21 (0.9608 s / it)
Averaged stats: model_time: 0.5223 (0.5672)  evaluator_time: 0.2176 (0.2682)
Accumulating evaluation results…
DONE (t=0.73s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.371
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.613
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.403
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.071
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.348
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.621
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.318
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.497
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.526
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.267
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.516
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.747
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69628e0e10>}


epoch: 19, training loss: 1.123286777175963, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:09  model_time: 0.5776 (0.5776)  evaluator_time:
0.3394 (0.3394)  time: 3.1705  data: 2.2375  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4908 (0.4963)  evaluator_time:
0.2199 (0.2508)  time: 0.7928  data: 0.0625  max mem: 5550
Test: Total time: 0:00:20 (0.9232 s / it)
Averaged stats: model_time: 0.4908 (0.4963)  evaluator_time: 0.2199 (0.2508)
```

```
Accumulating evaluation results…
DONE (t=0.72s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.321
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.561
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.315
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.086
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.314
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.320
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.489
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.519
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.334
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.498
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.702
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6971255550>}


epoch: 20, training loss: 1.0867871632799506, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:32  model_time: 0.5654 (0.5654)  evaluator_time:
0.4584 (0.4584)  time: 4.1992  data: 3.1653  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5318 (0.5200)  evaluator_time:
0.1789 (0.2166)  time: 0.7907  data: 0.0560  max mem: 5550
Test: Total time: 0:00:20 (0.9469 s / it)
Averaged stats: model_time: 0.5318 (0.5200)  evaluator_time: 0.1789 (0.2166)
Accumulating evaluation results…
DONE (t=0.69s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.349
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.583
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.382
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.090
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.329
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.597
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.329
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.473
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.521
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.288
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.509
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.727
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6971257610>}


epoch: 21, training loss: 1.0863277269527316, lr: 0.01
creating index…
index created!
```

```
Test:  [ 0/22]  eta: 0:00:47  model_time: 0.8751 (0.8751)  evaluator_time:
0.3930 (0.3930)  time: 2.1780  data: 0.8905  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4928 (0.5344)  evaluator_time:
0.1895 (0.2384)  time: 0.8052  data: 0.0619  max mem: 5550
Test: Total time: 0:00:19 (0.8891 s / it)
Averaged stats: model_time: 0.4928 (0.5344)  evaluator_time: 0.1895 (0.2384)
Accumulating evaluation results…
DONE (t=0.63s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.346
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.579
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.361
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.073
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.327
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.583
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.309
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.469
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.492
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.241
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.474
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.725
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6a8ef6d750>}


epoch: 22, training loss: 1.0624138824641705, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:09  model_time: 0.6297 (0.6297)  evaluator_time:
0.5183 (0.5183)  time: 3.1746  data: 2.0096  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4672 (0.5393)  evaluator_time:
0.1980 (0.2375)  time: 0.8187  data: 0.0621  max mem: 5550
Test: Total time: 0:00:20 (0.9420 s / it)
Averaged stats: model_time: 0.4672 (0.5393)  evaluator_time: 0.1980 (0.2375)
Accumulating evaluation results…
DONE (t=0.71s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.366
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.596
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.415
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.080
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.364
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.569
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.323
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.492
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.527
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.291
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.520
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.723
```

```
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962784b90>}


epoch: 23, training loss: 1.0344882323406637, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:44  model_time: 0.7480 (0.7480)  evaluator_time:
0.3626 (0.3626)  time: 2.0202  data: 0.8834  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5180 (0.5205)  evaluator_time:
0.2069 (0.2315)  time: 0.7856  data: 0.0563  max mem: 5550
Test: Total time: 0:00:20 (0.9212 s / it)
Averaged stats: model_time: 0.5180 (0.5205)  evaluator_time: 0.2069 (0.2315)
Accumulating evaluation results…
DONE (t=0.67s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.347
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.578
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.349
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.074
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.355
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.322
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.498
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.541
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.294
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.527
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.757
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f696209d390>}


epoch: 24, training loss: 1.073603137396276, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:51  model_time: 0.7724 (0.7724)  evaluator_time:
0.2871 (0.2871)  time: 2.3367  data: 1.2612  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4641 (0.5188)  evaluator_time:
0.2173 (0.2392)  time: 0.8011  data: 0.0632  max mem: 5550
Test: Total time: 0:00:19 (0.8919 s / it)
Averaged stats: model_time: 0.4641 (0.5188)  evaluator_time: 0.2173 (0.2392)
Accumulating evaluation results…
DONE (t=0.64s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.335
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.560
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.365
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.318
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.579
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.343
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.495
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.531
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.273
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.526
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.743
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962bbe8d0>}


epoch: 25, training loss: 1.0194266475737095, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:51  model_time: 0.8765 (0.8765)  evaluator_time:
0.5664 (0.5664)  time: 2.3333  data: 0.8724  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4668 (0.5326)  evaluator_time:
0.2082 (0.2652)  time: 0.8114  data: 0.0583  max mem: 5550
Test: Total time: 0:00:20 (0.9223 s / it)
Averaged stats: model_time: 0.4668 (0.5326)  evaluator_time: 0.2082 (0.2652)
Accumulating evaluation results…
DONE (t=0.66s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.348
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.577
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.367
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.079
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.346
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.567
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.320
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.462
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.530
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.328
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.542
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.667
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69710c02d0>}


epoch: 26, training loss: 0.9851195416413248, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:47  model_time: 0.7315 (0.7315)  evaluator_time:
0.4646 (0.4646)  time: 2.1713  data: 0.9579  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4635 (0.5369)  evaluator_time:
0.2316 (0.2556)  time: 0.8610  data: 0.0681  max mem: 5550
Test: Total time: 0:00:20 (0.9176 s / it)
Averaged stats: model_time: 0.4635 (0.5369)  evaluator_time: 0.2316 (0.2556)
Accumulating evaluation results…
DONE (t=0.75s).
IoU metric: bbox
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.339
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.561
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.355
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.086
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.331
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.575
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.324
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.478
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.542
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.319
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.528
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69713cdc50>}


epoch: 27, training loss: 0.9502438558265567, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:45  model_time: 0.8496 (0.8496)  evaluator_time:
0.4357 (0.4357)  time: 2.0837  data: 0.7772  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5175 (0.5465)  evaluator_time:
0.1929 (0.2482)  time: 0.8403  data: 0.0668  max mem: 5550
Test: Total time: 0:00:20 (0.9174 s / it)
Averaged stats: model_time: 0.5175 (0.5465)  evaluator_time: 0.1929 (0.2482)
Accumulating evaluation results…
DONE (t=0.63s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.375
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.614
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.427
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.078
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.336
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.645
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.350
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.511
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.559
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.288
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.563
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.764
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962bbefd0>}


epoch: 28, training loss: 0.9401585757732391, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:57  model_time: 0.4816 (0.4816)  evaluator_time:
0.2620 (0.2620)  time: 2.6205  data: 1.8670  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4707 (0.4990)  evaluator_time:
```

```
0.2238 (0.2356)  time: 0.7922  data: 0.0624  max mem: 5550
Test: Total time: 0:00:19 (0.8939 s / it)
Averaged stats: model_time: 0.4707 (0.4990)  evaluator_time: 0.2238 (0.2356)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.360
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.577
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.079
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.360
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.350
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.498
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.533
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.286
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.538
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.715
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697112b3d0>}


epoch: 29, training loss: 0.992657536175102, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:41  model_time: 0.8072 (0.8072)  evaluator_time:
0.2491 (0.2491)  time: 1.8664  data: 0.7921  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5005 (0.5326)  evaluator_time:
0.2079 (0.2344)  time: 0.8206  data: 0.0781  max mem: 5550
Test: Total time: 0:00:19 (0.8960 s / it)
Averaged stats: model_time: 0.5005 (0.5326)  evaluator_time: 0.2079 (0.2344)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.334
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.546
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.355
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.065
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.311
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.575
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.324
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.461
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.503
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.277
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.672
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697109b1d0>}
```

```
epoch: 30, training loss: 0.9635079335421324, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:48  model_time: 0.8621 (0.8621)  evaluator_time:
0.2282 (0.2282)  time: 2.1888  data: 1.0762  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4928 (0.5240)  evaluator_time:
0.2602 (0.2552)  time: 0.8309  data: 0.0658  max mem: 5550
Test: Total time: 0:00:20 (0.9091 s / it)
Averaged stats: model_time: 0.4928 (0.5240)  evaluator_time: 0.2602 (0.2552)
Accumulating evaluation results…
DONE (t=0.68s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.343
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.554
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.357
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.086
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.345
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.536
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.337
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.334
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.567
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.697
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735acf90>}


epoch: 31, training loss: 0.9568158886395395, lr: 0.01
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:52  model_time: 0.7099 (0.7099)  evaluator_time:
0.4941 (0.4941)  time: 2.4026  data: 1.1792  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4442 (0.5151)  evaluator_time:
0.1962 (0.2197)  time: 0.7633  data: 0.0682  max mem: 5550
Test: Total time: 0:00:19 (0.8704 s / it)
Averaged stats: model_time: 0.4442 (0.5151)  evaluator_time: 0.1962 (0.2197)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.324
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.532
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.364
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.059
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.281
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.605
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.314
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.463
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.516
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.242
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.529
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.712
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6967058310>}


epoch: 32, training loss: 0.8973044506274164, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:17  model_time: 0.6748 (0.6748)  evaluator_time:
0.4370 (0.4370)  time: 3.5086  data: 2.3870  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4804 (0.5052)  evaluator_time:
0.1564 (0.2183)  time: 0.7810  data: 0.0658  max mem: 5550
Test: Total time: 0:00:20 (0.9108 s / it)
Averaged stats: model_time: 0.4804 (0.5052)  evaluator_time: 0.1564 (0.2183)
Accumulating evaluation results…
DONE (t=0.61s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.370
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.604
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.421
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.082
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.355
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.618
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.355
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.520
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.322
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.542
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.734
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6973560fd0>}


epoch: 33, training loss: 0.8478483376093209, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:54  model_time: 0.9338 (0.9338)  evaluator_time:
0.6664 (0.6664)  time: 2.4605  data: 0.8425  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4896 (0.5176)  evaluator_time:
0.2115 (0.2608)  time: 0.8210  data: 0.0686  max mem: 5550
Test: Total time: 0:00:19 (0.9035 s / it)
Averaged stats: model_time: 0.4896 (0.5176)  evaluator_time: 0.2115 (0.2608)
Accumulating evaluation results…
DONE (t=0.62s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.616
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.085
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.365
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.620
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.356
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.523
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.319
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.547
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69722ed050>}


epoch: 34, training loss: 0.8454588232561946, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:24  model_time: 0.6664 (0.6664)  evaluator_time:
0.5725 (0.5725)  time: 3.8198  data: 2.5632  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4774 (0.4927)  evaluator_time:
0.1871 (0.2307)  time: 0.7695  data: 0.0696  max mem: 5550
Test: Total time: 0:00:20 (0.9201 s / it)
Averaged stats: model_time: 0.4774 (0.4927)  evaluator_time: 0.1871 (0.2307)
Accumulating evaluation results…
DONE (t=0.64s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.380
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.619
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.088
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.368
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.617
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.357
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.523
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.317
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.736
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69712028d0>}


epoch: 35, training loss: 0.8241126104258001, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:52  model_time: 0.8398 (0.8398)  evaluator_time:
0.7029 (0.7029)  time: 2.4029  data: 0.8405  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4699 (0.5199)  evaluator_time:
0.1719 (0.2464)  time: 0.7927  data: 0.0674  max mem: 5550
Test: Total time: 0:00:19 (0.8886 s / it)
Averaged stats: model_time: 0.4699 (0.5199)  evaluator_time: 0.1719 (0.2464)
```

```
Accumulating evaluation results…
DONE (t=0.62s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.383
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.621
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.436
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.091
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.370
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.617
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.360
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.526
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.313
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.741
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6971207810>}


epoch: 36, training loss: 0.8183679520152509, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:50  model_time: 0.7588 (0.7588)  evaluator_time:
0.4941 (0.4941)  time: 2.2865  data: 1.0235  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4681 (0.5095)  evaluator_time:
0.1829 (0.2376)  time: 0.7852  data: 0.0621  max mem: 5550
Test: Total time: 0:00:20 (0.9145 s / it)
Averaged stats: model_time: 0.4681 (0.5095)  evaluator_time: 0.1829 (0.2376)
Accumulating evaluation results…
DONE (t=0.64s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.385
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.625
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.436
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.092
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.371
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.619
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.358
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.526
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.564
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.313
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.576
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6972041290>}


epoch: 37, training loss: 0.78047746559605, lr: 0.0001
creating index…
index created!
```

```
Test:  [ 0/22]  eta: 0:01:04  model_time: 0.5404 (0.5404)  evaluator_time:
0.4253 (0.4253)  time: 2.9338  data: 1.9575  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4823 (0.5013)  evaluator_time:
0.2182 (0.2481)  time: 0.8109  data: 0.0606  max mem: 5550
Test: Total time: 0:00:20 (0.9140 s / it)
Averaged stats: model_time: 0.4823 (0.5013)  evaluator_time: 0.2182 (0.2481)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.385
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.627
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.437
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.092
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.371
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.618
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.350
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.520
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.559
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.314
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.567
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.736
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735824d0>}


epoch: 38, training loss: 0.775688661262393, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:43  model_time: 0.7959 (0.7959)  evaluator_time:
0.3981 (0.3981)  time: 1.9843  data: 0.7730  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4941 (0.5247)  evaluator_time:
0.2286 (0.2587)  time: 0.8050  data: 0.0673  max mem: 5550
Test: Total time: 0:00:19 (0.9036 s / it)
Averaged stats: model_time: 0.4941 (0.5247)  evaluator_time: 0.2286 (0.2587)
Accumulating evaluation results…
DONE (t=0.61s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.386
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.435
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.092
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.374
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.617
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.352
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.517
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.555
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.316
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.560
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.736
```

```
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69672a5f50>}


epoch: 39, training loss: 0.7727686730213463, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:55  model_time: 1.0758 (1.0758)  evaluator_time:
0.4522 (0.4522)  time: 2.5221  data: 0.9771  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4517 (0.5428)  evaluator_time:
0.1946 (0.2573)  time: 0.7979  data: 0.0598  max mem: 5550
Test: Total time: 0:00:20 (0.9175 s / it)
Averaged stats: model_time: 0.4517 (0.5428)  evaluator_time: 0.1946 (0.2573)
Accumulating evaluation results…
DONE (t=0.67s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.385
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.375
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.618
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.510
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.557
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.343
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.741
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697359ea50>}


epoch: 40, training loss: 0.7794302888214588, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:37  model_time: 0.6913 (0.6913)  evaluator_time:
0.1953 (0.1953)  time: 1.7182  data: 0.8138  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4646 (0.5191)  evaluator_time:
0.2434 (0.2426)  time: 0.8056  data: 0.0656  max mem: 5550
Test: Total time: 0:00:19 (0.8798 s / it)
Averaged stats: model_time: 0.4646 (0.5191)  evaluator_time: 0.2434 (0.2426)
Accumulating evaluation results…
DONE (t=0.61s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.386
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.628
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.433
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.371
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.620
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.350
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.324
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.540
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.749
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6973592dd0>}


epoch: 41, training loss: 0.7754894639365375, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:47  model_time: 0.8508 (0.8508)  evaluator_time:
0.4560 (0.4560)  time: 2.1442  data: 0.8190  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4983 (0.5272)  evaluator_time:
0.2120 (0.2440)  time: 0.8302  data: 0.0679  max mem: 5550
Test: Total time: 0:00:19 (0.8930 s / it)
Averaged stats: model_time: 0.4983 (0.5272)  evaluator_time: 0.2120 (0.2440)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.386
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.431
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.376
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.621
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.344
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.509
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.316
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.542
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.755
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69611031d0>}


epoch: 42, training loss: 0.7545679155737162, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:44  model_time: 0.4828 (0.4828)  evaluator_time:
0.2548 (0.2548)  time: 2.0312  data: 1.2832  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4634 (0.4868)  evaluator_time:
0.2057 (0.2473)  time: 0.7932  data: 0.0719  max mem: 5550
Test: Total time: 0:00:19 (0.8746 s / it)
Averaged stats: model_time: 0.4634 (0.4868)  evaluator_time: 0.2057 (0.2473)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.388
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.432
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.377
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.510
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.556
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.321
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.751
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6972344210>}


epoch: 43, training loss: 0.7503184089437127, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:12  model_time: 0.5184 (0.5184)  evaluator_time:
0.2466 (0.2466)  time: 3.2964  data: 2.5213  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4728 (0.4772)  evaluator_time:
0.1963 (0.2314)  time: 0.7634  data: 0.0630  max mem: 5550
Test: Total time: 0:00:19 (0.8964 s / it)
Averaged stats: model_time: 0.4728 (0.4772)  evaluator_time: 0.1963 (0.2314)
Accumulating evaluation results…
DONE (t=0.57s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.433
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.094
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.377
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.344
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.323
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.546
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.750
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6a7518d050>}


epoch: 44, training loss: 0.7427291590720415, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:01  model_time: 0.7865 (0.7865)  evaluator_time:
0.3900 (0.3900)  time: 2.8007  data: 1.6014  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4726 (0.4939)  evaluator_time:
```

```
0.1969 (0.2383)  time: 0.7545  data: 0.0638  max mem: 5550
Test: Total time: 0:00:19 (0.8811 s / it)
Averaged stats: model_time: 0.4726 (0.4939)  evaluator_time: 0.1969 (0.2383)
Accumulating evaluation results…
DONE (t=0.62s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.630
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.435
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.621
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.345
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.502
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.547
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.326
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.542
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.729
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69628d9c10>}


epoch: 45, training loss: 0.7431679600849748, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:00  model_time: 0.8131 (0.8131)  evaluator_time:
0.4544 (0.4544)  time: 2.7395  data: 1.4530  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4847 (0.5148)  evaluator_time:
0.1785 (0.2276)  time: 0.7535  data: 0.0636  max mem: 5550
Test: Total time: 0:00:19 (0.8941 s / it)
Averaged stats: model_time: 0.4847 (0.5148)  evaluator_time: 0.1785 (0.2276)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.632
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.378
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.621
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.344
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.508
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.312
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.545
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.755
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735f9210>}
```

```
epoch: 46, training loss: 0.7368227518163621, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:47  model_time: 0.7986 (0.7986)  evaluator_time:
0.4565 (0.4565)  time: 2.1642  data: 0.8934  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4657 (0.4950)  evaluator_time:
0.1718 (0.2409)  time: 0.7615  data: 0.0548  max mem: 5550
Test: Total time: 0:00:19 (0.8926 s / it)
Averaged stats: model_time: 0.4657 (0.4950)  evaluator_time: 0.1718 (0.2409)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.387
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.429
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.378
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.621
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.344
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.514
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.561
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.344
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.755
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735fd250>}


epoch: 47, training loss: 0.7472575001884252, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:52  model_time: 0.7414 (0.7414)  evaluator_time:
0.2899 (0.2899)  time: 2.3824  data: 1.3301  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5011 (0.5166)  evaluator_time:
0.1709 (0.2262)  time: 0.7798  data: 0.0673  max mem: 5550
Test: Total time: 0:00:19 (0.8862 s / it)
Averaged stats: model_time: 0.5011 (0.5166)  evaluator_time: 0.1709 (0.2262)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.630
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.431
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.380
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.621
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.345
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.509
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.556
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.316
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.755
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6972231710>}


epoch: 48, training loss: 0.7292201286181808, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:11  model_time: 0.7323 (0.7323)  evaluator_time:
0.2957 (0.2957)  time: 3.2667  data: 2.2211  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4699 (0.4887)  evaluator_time:
0.1795 (0.2230)  time: 0.7410  data: 0.0615  max mem: 5550
Test: Total time: 0:00:19 (0.8871 s / it)
Averaged stats: model_time: 0.4699 (0.4887)  evaluator_time: 0.1795 (0.2230)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.628
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.376
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.503
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.318
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.734
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697206d110>}


epoch: 49, training loss: 0.7317307190969586, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:49  model_time: 0.9977 (0.9977)  evaluator_time:
0.3256 (0.3256)  time: 2.2720  data: 0.9299  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4871 (0.5228)  evaluator_time:
0.1829 (0.2217)  time: 0.7857  data: 0.0571  max mem: 5550
Test: Total time: 0:00:19 (0.9001 s / it)
Averaged stats: model_time: 0.4871 (0.5228)  evaluator_time: 0.1829 (0.2217)
Accumulating evaluation results…
DONE (t=0.63s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.628
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.432
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.380
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.321
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6961100ad0>}


epoch: 50, training loss: 0.7309305733069777, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:10  model_time: 0.7068 (0.7068)  evaluator_time:
0.1920 (0.1920)  time: 3.2100  data: 2.2938  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4760 (0.5002)  evaluator_time:
0.1670 (0.2077)  time: 0.7337  data: 0.0629  max mem: 5550
Test: Total time: 0:00:19 (0.8875 s / it)
Averaged stats: model_time: 0.4760 (0.5002)  evaluator_time: 0.1670 (0.2077)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.388
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.627
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.377
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.347
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.509
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.556
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.345
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962ac0050>}


epoch: 51, training loss: 0.723961332347244, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:43  model_time: 0.7600 (0.7600)  evaluator_time:
0.3551 (0.3551)  time: 1.9644  data: 0.8257  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4563 (0.5271)  evaluator_time:
0.1912 (0.2378)  time: 0.7945  data: 0.0682  max mem: 5550
Test: Total time: 0:00:19 (0.8886 s / it)
Averaged stats: model_time: 0.4563 (0.5271)  evaluator_time: 0.1912 (0.2378)
```

```
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.628
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.377
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.626
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.349
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.323
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.734
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697359ea50>}


epoch: 52, training loss: 0.7230595569126308, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:01  model_time: 0.8615 (0.8615)  evaluator_time:
0.2962 (0.2962)  time: 2.8005  data: 1.6268  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4872 (0.5255)  evaluator_time:
0.1812 (0.2262)  time: 0.7697  data: 0.0618  max mem: 5550
Test: Total time: 0:00:19 (0.8993 s / it)
Averaged stats: model_time: 0.4872 (0.5255)  evaluator_time: 0.1812 (0.2262)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.431
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.376
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.324
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697233bb10>}


epoch: 53, training loss: 0.735398753080517, lr: 0.0001
creating index…
index created!
```

```
Test:  [ 0/22]  eta: 0:01:14  model_time: 0.6500 (0.6500)  evaluator_time:
0.3308 (0.3308)  time: 3.3874  data: 2.3971  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4674 (0.5087)  evaluator_time:
0.1892 (0.2116)  time: 0.7721  data: 0.0592  max mem: 5550
Test: Total time: 0:00:19 (0.8999 s / it)
Averaged stats: model_time: 0.4674 (0.5087)  evaluator_time: 0.1892 (0.2116)
Accumulating evaluation results…
DONE (t=0.63s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.392
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.630
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.427
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.381
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.630
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.513
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.558
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.347
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.741
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69720478d0>}


epoch: 54, training loss: 0.7226636332925409, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:42  model_time: 0.7082 (0.7082)  evaluator_time:
0.3100 (0.3100)  time: 1.9358  data: 0.8988  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4527 (0.5163)  evaluator_time:
0.1756 (0.2397)  time: 0.8054  data: 0.0755  max mem: 5550
Test: Total time: 0:00:19 (0.8866 s / it)
Averaged stats: model_time: 0.4527 (0.5163)  evaluator_time: 0.1756 (0.2397)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.388
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.412
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.373
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.627
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.503
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.321
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.544
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
```

```
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735d9250>}


epoch: 55, training loss: 0.6978816972114146, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:51  model_time: 0.8910 (0.8910)  evaluator_time:
0.4527 (0.4527)  time: 2.3183  data: 0.9550  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4664 (0.4974)  evaluator_time:
0.2010 (0.2564)  time: 0.7899  data: 0.0600  max mem: 5550
Test: Total time: 0:00:19 (0.8877 s / it)
Averaged stats: model_time: 0.4664 (0.4974)  evaluator_time: 0.2010 (0.2564)
Accumulating evaluation results…
DONE (t=0.61s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.626
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.094
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.375
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.627
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.349
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.319
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69721ecc90>}


epoch: 56, training loss: 0.7311661839485168, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:40  model_time: 0.6678 (0.6678)  evaluator_time:
0.2169 (0.2169)  time: 1.8515  data: 0.9523  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5091 (0.5281)  evaluator_time:
0.1983 (0.2362)  time: 0.8135  data: 0.0729  max mem: 5550
Test: Total time: 0:00:19 (0.8940 s / it)
Averaged stats: model_time: 0.5091 (0.5281)  evaluator_time: 0.1983 (0.2362)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.428
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.097
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.376
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.629
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.349
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.512
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.556
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.345
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69720c4dd0>}


epoch: 57, training loss: 0.7259415695443749, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:02  model_time: 0.9089 (0.9089)  evaluator_time:
0.3746 (0.3746)  time: 2.8269  data: 1.5265  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4837 (0.5387)  evaluator_time:
0.1900 (0.2207)  time: 0.8041  data: 0.0667  max mem: 5550
Test: Total time: 0:00:19 (0.9068 s / it)
Averaged stats: model_time: 0.4837 (0.5387)  evaluator_time: 0.1900 (0.2207)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.627
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.421
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.098
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.373
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.625
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.345
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.323
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.547
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.736
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6961100190>}


epoch: 58, training loss: 0.7125398677308112, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:46  model_time: 0.8457 (0.8457)  evaluator_time:
0.3994 (0.3994)  time: 2.1183  data: 0.8554  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4844 (0.5245)  evaluator_time:
0.1657 (0.2258)  time: 0.7868  data: 0.0702  max mem: 5550
Test: Total time: 0:00:19 (0.9017 s / it)
Averaged stats: model_time: 0.4844 (0.5245)  evaluator_time: 0.1657 (0.2258)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.419
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.376
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.629
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.319
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6973583410>}


epoch: 59, training loss: 0.7150231753475964, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:52  model_time: 0.7241 (0.7241)  evaluator_time:
0.3517 (0.3517)  time: 2.3727  data: 1.2858  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4642 (0.5254)  evaluator_time:
0.1773 (0.2215)  time: 0.7839  data: 0.0635  max mem: 5550
Test: Total time: 0:00:19 (0.8813 s / it)
Averaged stats: model_time: 0.4642 (0.5254)  evaluator_time: 0.1773 (0.2215)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.628
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.432
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.374
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.512
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.557
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.349
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.736
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697118add0>}


epoch: 60, training loss: 0.7208215785212815, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:43  model_time: 0.8371 (0.8371)  evaluator_time:
0.3782 (0.3782)  time: 1.9853  data: 0.7482  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4806 (0.5240)  evaluator_time:
```

```
0.1942 (0.2373)  time: 0.7971  data: 0.0664  max mem: 5550
Test: Total time: 0:00:19 (0.8816 s / it)
Averaged stats: model_time: 0.4806 (0.5240)  evaluator_time: 0.1942 (0.2373)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.631
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.420
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.094
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.374
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.629
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.326
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.545
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.740
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6973648150>}


epoch: 61, training loss: 0.7082522052805871, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:42  model_time: 0.7316 (0.7316)  evaluator_time:
0.2982 (0.2982)  time: 1.9507  data: 0.9044  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4529 (0.5255)  evaluator_time:
0.1648 (0.2227)  time: 0.7510  data: 0.0611  max mem: 5550
Test: Total time: 0:00:19 (0.8788 s / it)
Averaged stats: model_time: 0.4529 (0.5255)  evaluator_time: 0.1648 (0.2227)
Accumulating evaluation results…
DONE (t=0.57s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.416
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.094
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.371
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.629
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.329
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.545
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.741
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69710c0d90>}
```

```
epoch: 62, training loss: 0.7075089241843671, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:45  model_time: 0.7093 (0.7093)  evaluator_time:
0.2765 (0.2765)  time: 2.0874  data: 1.0901  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5229 (0.5179)  evaluator_time:
0.2105 (0.2444)  time: 0.8327  data: 0.0769  max mem: 5550
Test: Total time: 0:00:19 (0.8994 s / it)
Averaged stats: model_time: 0.5229 (0.5179)  evaluator_time: 0.2105 (0.2444)
Accumulating evaluation results…
DONE (t=0.61s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.387
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.630
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.417
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.094
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.373
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.331
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.545
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.734
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735f9210>}


epoch: 63, training loss: 0.7019422589801252, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:52  model_time: 0.9637 (0.9637)  evaluator_time:
0.5118 (0.5118)  time: 2.3815  data: 0.8849  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4672 (0.5189)  evaluator_time:
0.1902 (0.2318)  time: 0.7766  data: 0.0689  max mem: 5550
Test: Total time: 0:00:19 (0.8880 s / it)
Averaged stats: model_time: 0.4672 (0.5189)  evaluator_time: 0.1902 (0.2318)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.631
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.417
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.097
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.373
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.628
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.513
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.558
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.357
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.544
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.740
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6966144f10>}


epoch: 64, training loss: 0.7096419697627425, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:14  model_time: 0.5039 (0.5039)  evaluator_time:
0.1837 (0.1837)  time: 3.3782  data: 2.6810  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4651 (0.4851)  evaluator_time:
0.1855 (0.2183)  time: 0.7650  data: 0.0652  max mem: 5550
Test: Total time: 0:00:19 (0.9017 s / it)
Averaged stats: model_time: 0.4651 (0.4851)  evaluator_time: 0.1855 (0.2183)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.630
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.429
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.376
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.323
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.742
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735ba390>}


epoch: 65, training loss: 0.7018594392575324, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:49  model_time: 0.9476 (0.9476)  evaluator_time:
0.3778 (0.3778)  time: 2.2355  data: 0.8929  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4575 (0.5308)  evaluator_time:
0.1792 (0.2383)  time: 0.7679  data: 0.0675  max mem: 5550
Test: Total time: 0:00:19 (0.8941 s / it)
Averaged stats: model_time: 0.4575 (0.5308)  evaluator_time: 0.1792 (0.2383)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.418
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.094
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.374
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.322
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.743
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6973543350>}


epoch: 66, training loss: 0.6957095521502197, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:44  model_time: 0.6620 (0.6620)  evaluator_time:
0.3595 (0.3595)  time: 2.0183  data: 0.9801  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4726 (0.5161)  evaluator_time:
0.2148 (0.2528)  time: 0.8022  data: 0.0651  max mem: 5550
Test: Total time: 0:00:19 (0.8931 s / it)
Averaged stats: model_time: 0.4726 (0.5161)  evaluator_time: 0.2148 (0.2528)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.416
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.376
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.345
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.326
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697107af50>}


epoch: 67, training loss: 0.6778616174124181, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:48  model_time: 0.8188 (0.8188)  evaluator_time:
0.4623 (0.4623)  time: 2.1825  data: 0.8911  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4785 (0.5332)  evaluator_time:
0.1751 (0.2459)  time: 0.7942  data: 0.0744  max mem: 5550
Test: Total time: 0:00:20 (0.9097 s / it)
Averaged stats: model_time: 0.4785 (0.5332)  evaluator_time: 0.1751 (0.2459)
```

```
Accumulating evaluation results…
DONE (t=0.61s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.631
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.418
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.094
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.378
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.346
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.322
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.741
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f692c88e5d0>}


epoch: 68, training loss: 0.7089858367107809, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:44  model_time: 0.8045 (0.8045)  evaluator_time:
0.3349 (0.3349)  time: 2.0202  data: 0.8699  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4783 (0.5177)  evaluator_time:
0.2135 (0.2305)  time: 0.8133  data: 0.0825  max mem: 5550
Test: Total time: 0:00:19 (0.8870 s / it)
Averaged stats: model_time: 0.4783 (0.5177)  evaluator_time: 0.2135 (0.2305)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.388
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.631
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.416
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.374
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.624
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.345
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.321
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.547
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.741
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735ae310>}


epoch: 69, training loss: 0.7045988992322236, lr: 0.0001
creating index…
index created!
```

```
Test:  [ 0/22]  eta: 0:00:40  model_time: 0.7351 (0.7351)  evaluator_time:
0.3034 (0.3034)  time: 1.8514  data: 0.7959  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4836 (0.5165)  evaluator_time:
0.1926 (0.2428)  time: 0.7967  data: 0.0644  max mem: 5550
Test: Total time: 0:00:19 (0.8762 s / it)
Averaged stats: model_time: 0.4836 (0.5165)  evaluator_time: 0.1926 (0.2428)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.631
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.419
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.097
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.377
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.621
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.341
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.511
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.558
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.353
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.734
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6971115f10>}


epoch: 70, training loss: 0.6852119578979909, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:48  model_time: 0.6545 (0.6545)  evaluator_time:
0.2560 (0.2560)  time: 2.2247  data: 1.3005  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4883 (0.5144)  evaluator_time:
0.1839 (0.2334)  time: 0.7780  data: 0.0647  max mem: 5550
Test: Total time: 0:00:19 (0.8851 s / it)
Averaged stats: model_time: 0.4883 (0.5144)  evaluator_time: 0.1839 (0.2334)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.631
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.418
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.341
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.322
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
```

```
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6961100a90>}


epoch: 71, training loss: 0.688182502053678, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:03  model_time: 0.8013 (0.8013)  evaluator_time:
0.3912 (0.3912)  time: 2.8872  data: 1.6837  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4626 (0.5040)  evaluator_time:
0.2217 (0.2205)  time: 0.8029  data: 0.0879  max mem: 5550
Test: Total time: 0:00:19 (0.8982 s / it)
Averaged stats: model_time: 0.4626 (0.5040)  evaluator_time: 0.2217 (0.2205)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.631
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.420
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.097
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.341
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.326
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6971231810>}


epoch: 72, training loss: 0.7010308583267033, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:42  model_time: 0.8364 (0.8364)  evaluator_time:
0.3282 (0.3282)  time: 1.9485  data: 0.7693  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.5006 (0.5277)  evaluator_time:
0.2113 (0.2264)  time: 0.8079  data: 0.0538  max mem: 5550
Test: Total time: 0:00:19 (0.8914 s / it)
Averaged stats: model_time: 0.5006 (0.5277)  evaluator_time: 0.2113 (0.2264)
Accumulating evaluation results…
DONE (t=0.61s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.389
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.416
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.098
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.378
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.341
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.505
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.325
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.550
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.734
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69735ca750>}


epoch: 73, training loss: 0.6900442542973906, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:19  model_time: 0.9341 (0.9341)  evaluator_time:
0.3925 (0.3925)  time: 3.6047  data: 2.2603  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4562 (0.5127)  evaluator_time:
0.1703 (0.2035)  time: 0.7503  data: 0.0612  max mem: 5550
Test: Total time: 0:00:19 (0.8927 s / it)
Averaged stats: model_time: 0.4562 (0.5127)  evaluator_time: 0.1703 (0.2035)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.418
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.098
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.624
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.341
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.506
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.324
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.555
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962784890>}


epoch: 74, training loss: 0.6928675111848861, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:01  model_time: 0.5542 (0.5542)  evaluator_time:
0.2285 (0.2285)  time: 2.7762  data: 1.9835  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4284 (0.4980)  evaluator_time:
0.2220 (0.2262)  time: 0.7910  data: 0.0629  max mem: 5550
Test: Total time: 0:00:19 (0.8904 s / it)
Averaged stats: model_time: 0.4284 (0.4980)  evaluator_time: 0.2220 (0.2262)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.630
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.418
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.624
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.342
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.328
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.552
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69744ba450>}


epoch: 75, training loss: 0.6810111734084785, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:46  model_time: 0.7736 (0.7736)  evaluator_time:
0.4158 (0.4158)  time: 2.1112  data: 0.9043  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4695 (0.5266)  evaluator_time:
0.2104 (0.2357)  time: 0.8142  data: 0.0697  max mem: 5550
Test: Total time: 0:00:19 (0.8912 s / it)
Averaged stats: model_time: 0.4695 (0.5266)  evaluator_time: 0.2104 (0.2357)
Accumulating evaluation results…
DONE (t=0.57s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.420
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.096
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.377
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.342
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.555
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.325
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.556
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.735
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6960f82b50>}


epoch: 76, training loss: 0.6898893732577562, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:01:05  model_time: 0.4910 (0.4910)  evaluator_time:
0.2134 (0.2134)  time: 2.9908  data: 2.2763  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4802 (0.5101)  evaluator_time:
```

```
0.1751 (0.2150)  time: 0.7792  data: 0.0651  max mem: 5550
Test: Total time: 0:00:19 (0.9064 s / it)
Averaged stats: model_time: 0.4802 (0.5101)  evaluator_time: 0.1751 (0.2150)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.392
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.628
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.091
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.383
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.349
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.510
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.329
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697360a7d0>}


epoch: 77, training loss: 0.6910979349631816, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:47  model_time: 0.5197 (0.5197)  evaluator_time:
0.2465 (0.2465)  time: 2.1583  data: 1.3822  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4600 (0.5116)  evaluator_time:
0.1766 (0.2445)  time: 0.7957  data: 0.0609  max mem: 5550
Test: Total time: 0:00:19 (0.8928 s / it)
Averaged stats: model_time: 0.4600 (0.5116)  evaluator_time: 0.1766 (0.2445)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.629
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.419
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.380
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.342
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.501
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.548
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.296
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962b02750>}
```

```
epoch: 78, training loss: 0.6775415851734579, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:52  model_time: 0.8828 (0.8828)  evaluator_time:
0.5119 (0.5119)  time: 2.3862  data: 0.9743  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4948 (0.5277)  evaluator_time:
0.1613 (0.2263)  time: 0.7608  data: 0.0611  max mem: 5550
Test: Total time: 0:00:19 (0.8977 s / it)
Averaged stats: model_time: 0.4948 (0.5277)  evaluator_time: 0.1613 (0.2263)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.627
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.430
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.092
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.623
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.349
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.507
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.322
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.741
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f697360a7d0>}


epoch: 79, training loss: 0.6817199701908976, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:45  model_time: 0.6886 (0.6886)  evaluator_time:
0.2486 (0.2486)  time: 2.0613  data: 1.1068  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4933 (0.5205)  evaluator_time:
0.1960 (0.2533)  time: 0.8325  data: 0.0814  max mem: 5550
Test: Total time: 0:00:20 (0.9176 s / it)
Averaged stats: model_time: 0.4933 (0.5205)  evaluator_time: 0.1960 (0.2533)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.626
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.417
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.095
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.378
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.348
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.500
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.547
```

```
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.299
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.551
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.734
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6971086fd0>}


epoch: 80, training loss: 0.6823933951091021, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:50  model_time: 0.7129 (0.7129)  evaluator_time:
0.3560 (0.3560)  time: 2.3032  data: 1.2165  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4845 (0.5181)  evaluator_time:
0.1952 (0.2282)  time: 0.7616  data: 0.0671  max mem: 5550
Test: Total time: 0:00:19 (0.8850 s / it)
Averaged stats: model_time: 0.4845 (0.5181)  evaluator_time: 0.1952 (0.2282)
Accumulating evaluation results…
DONE (t=0.58s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.391
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.627
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.429
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.093
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.381
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.624
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.349
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.503
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.549
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.297
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.554
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.740
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f6962ae0e10>}


epoch: 81, training loss: 0.6852330153342336, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:46  model_time: 0.8172 (0.8172)  evaluator_time:
0.3514 (0.3514)  time: 2.1078  data: 0.9229  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4814 (0.5234)  evaluator_time:
0.1560 (0.2317)  time: 0.7860  data: 0.0617  max mem: 5550
Test: Total time: 0:00:19 (0.8888 s / it)
Averaged stats: model_time: 0.4814 (0.5234)  evaluator_time: 0.1560 (0.2317)
Accumulating evaluation results…
DONE (t=0.59s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.390
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.626
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.429
```

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.091
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.624
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.356
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.499
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.547
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.295
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.553
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.736
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69722e1f50>}


epoch: 82, training loss: 0.6953576051164418, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:45  model_time: 0.8563 (0.8563)  evaluator_time:
0.2746 (0.2746)  time: 2.0554  data: 0.9117  max mem: 5550
Test:  [21/22]  eta: 0:00:00  model_time: 0.4290 (0.5511)  evaluator_time:
0.1900 (0.2053)  time: 0.8160  data: 0.0783  max mem: 5550
Test: Total time: 0:00:19 (0.8857 s / it)
Averaged stats: model_time: 0.4290 (0.5511)  evaluator_time: 0.1900 (0.2053)
Accumulating evaluation results…
DONE (t=0.60s).
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.388
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.625
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.427
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.091
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.380
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.622
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.347
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.508
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.557
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.327
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.556
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.739
{'bbox': <pycocotools.cocoeval.COCOeval object at 0x7f69610ada90>}


epoch: 83, training loss: 0.6889228262007236, lr: 0.0001
creating index…
index created!
Test:  [ 0/22]  eta: 0:00:44  model_time: 0.7438 (0.7438)  evaluator_time:
0.4011 (0.4011)  time: 2.0074  data: 0.8449  max mem: 5550
```

# 13 Inference

### 13.0.1 Load Model From Checkpoint

[ ]:
```
SSD_MODEL = torchvision.models.detection.
 ↪ssdlite320_mobilenet_v3_large(pretrained=False)

checkPoint = torch.load("/content/gdrive/MyDrive/facemask_detection/save_model/
 ↪facemaskDetectionSSD_320x320_38MAP.pth",
                         map_location=getDevice())

SSD_MODEL.load_state_dict(checkPoint['model_state_dict'])
SSD_MODEL.eval()
SSD_MODEL.to(getDevice())
```

[ ]:
```
SSD(
  (backbone): SSDLiteFeatureExtractorMobileNet(
    (features): Sequential(
      (0): Sequential(
        (0): ConvBNActivation(
          (0): Conv2d(3, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1),
bias=False)
          (1): BatchNorm2d(16, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): Hardswish()
        )
        (1): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=16, bias=False)
              (1): BatchNorm2d(16, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (1): ConvBNActivation(
              (0): Conv2d(16, 16, kernel_size=(1, 1), stride=(1, 1), bias=False)
              (1): BatchNorm2d(16, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (2): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(16, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)
```

```
              (1): BatchNorm2d(64, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (1): ConvBNActivation(
              (0): Conv2d(64, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1,
1), groups=64, bias=False)
              (1): BatchNorm2d(64, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (2): ConvBNActivation(
              (0): Conv2d(64, 24, kernel_size=(1, 1), stride=(1, 1), bias=False)
              (1): BatchNorm2d(24, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (3): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(24, 72, kernel_size=(1, 1), stride=(1, 1), bias=False)
              (1): BatchNorm2d(72, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (1): ConvBNActivation(
              (0): Conv2d(72, 72, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=72, bias=False)
              (1): BatchNorm2d(72, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (2): ConvBNActivation(
              (0): Conv2d(72, 24, kernel_size=(1, 1), stride=(1, 1), bias=False)
              (1): BatchNorm2d(24, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (4): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(24, 72, kernel_size=(1, 1), stride=(1, 1), bias=False)
              (1): BatchNorm2d(72, eps=0.001, momentum=0.03, affine=True,
```

```
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (1): ConvBNActivation(
              (0): Conv2d(72, 72, kernel_size=(5, 5), stride=(2, 2), padding=(2,
2), groups=72, bias=False)
              (1): BatchNorm2d(72, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (2): SqueezeExcitation(
              (fc1): Conv2d(72, 24, kernel_size=(1, 1), stride=(1, 1))
              (relu): ReLU(inplace=True)
              (fc2): Conv2d(24, 72, kernel_size=(1, 1), stride=(1, 1))
            )
            (3): ConvBNActivation(
              (0): Conv2d(72, 40, kernel_size=(1, 1), stride=(1, 1), bias=False)
              (1): BatchNorm2d(40, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (5): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(40, 120, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(120, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (1): ConvBNActivation(
              (0): Conv2d(120, 120, kernel_size=(5, 5), stride=(1, 1),
padding=(2, 2), groups=120, bias=False)
              (1): BatchNorm2d(120, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (2): SqueezeExcitation(
              (fc1): Conv2d(120, 32, kernel_size=(1, 1), stride=(1, 1))
              (relu): ReLU(inplace=True)
              (fc2): Conv2d(32, 120, kernel_size=(1, 1), stride=(1, 1))
            )
            (3): ConvBNActivation(
              (0): Conv2d(120, 40, kernel_size=(1, 1), stride=(1, 1),
bias=False)
```

```
              (1): BatchNorm2d(40, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (6): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(40, 120, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(120, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (1): ConvBNActivation(
              (0): Conv2d(120, 120, kernel_size=(5, 5), stride=(1, 1),
padding=(2, 2), groups=120, bias=False)
              (1): BatchNorm2d(120, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): ReLU(inplace=True)
            )
            (2): SqueezeExcitation(
              (fc1): Conv2d(120, 32, kernel_size=(1, 1), stride=(1, 1))
              (relu): ReLU(inplace=True)
              (fc2): Conv2d(32, 120, kernel_size=(1, 1), stride=(1, 1))
            )
            (3): ConvBNActivation(
              (0): Conv2d(120, 40, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(40, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (7): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(40, 240, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(240, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (1): ConvBNActivation(
              (0): Conv2d(240, 240, kernel_size=(3, 3), stride=(2, 2),
```

```
padding=(1, 1), groups=240, bias=False)
              (1): BatchNorm2d(240, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): ConvBNActivation(
              (0): Conv2d(240, 80, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(80, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (8): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(80, 200, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(200, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (1): ConvBNActivation(
              (0): Conv2d(200, 200, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), groups=200, bias=False)
              (1): BatchNorm2d(200, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): ConvBNActivation(
              (0): Conv2d(200, 80, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(80, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (9): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(80, 184, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(184, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
```

```
            )
            (1): ConvBNActivation(
              (0): Conv2d(184, 184, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), groups=184, bias=False)
              (1): BatchNorm2d(184, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): ConvBNActivation(
              (0): Conv2d(184, 80, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(80, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (10): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(80, 184, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(184, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (1): ConvBNActivation(
              (0): Conv2d(184, 184, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), groups=184, bias=False)
              (1): BatchNorm2d(184, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): ConvBNActivation(
              (0): Conv2d(184, 80, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(80, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (11): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(80, 480, kernel_size=(1, 1), stride=(1, 1),
bias=False)
```

```
              (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (1): ConvBNActivation(
              (0): Conv2d(480, 480, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), groups=480, bias=False)
              (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): SqueezeExcitation(
              (fc1): Conv2d(480, 120, kernel_size=(1, 1), stride=(1, 1))
              (relu): ReLU(inplace=True)
              (fc2): Conv2d(120, 480, kernel_size=(1, 1), stride=(1, 1))
            )
            (3): ConvBNActivation(
              (0): Conv2d(480, 112, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(112, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (12): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(112, 672, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(672, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (1): ConvBNActivation(
              (0): Conv2d(672, 672, kernel_size=(3, 3), stride=(1, 1),
padding=(1, 1), groups=672, bias=False)
              (1): BatchNorm2d(672, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): SqueezeExcitation(
              (fc1): Conv2d(672, 168, kernel_size=(1, 1), stride=(1, 1))
              (relu): ReLU(inplace=True)
              (fc2): Conv2d(168, 672, kernel_size=(1, 1), stride=(1, 1))
            )
            (3): ConvBNActivation(
```

```
              (0): Conv2d(672, 112, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(112, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (13): ConvBNActivation(
          (0): Conv2d(112, 672, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(672, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): Hardswish()
        )
      )
      (1): Sequential(
        (0): Sequential(
          (1): ConvBNActivation(
            (0): Conv2d(672, 672, kernel_size=(5, 5), stride=(2, 2), padding=(2,
2), groups=672, bias=False)
            (1): BatchNorm2d(672, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): Hardswish()
          )
          (2): SqueezeExcitation(
            (fc1): Conv2d(672, 168, kernel_size=(1, 1), stride=(1, 1))
            (relu): ReLU(inplace=True)
            (fc2): Conv2d(168, 672, kernel_size=(1, 1), stride=(1, 1))
          )
          (3): ConvBNActivation(
            (0): Conv2d(672, 80, kernel_size=(1, 1), stride=(1, 1), bias=False)
            (1): BatchNorm2d(80, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): Identity()
          )
        )
        (1): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(80, 480, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (1): ConvBNActivation(
              (0): Conv2d(480, 480, kernel_size=(5, 5), stride=(1, 1),
```

```
padding=(2, 2), groups=480, bias=False)
              (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): SqueezeExcitation(
              (fc1): Conv2d(480, 120, kernel_size=(1, 1), stride=(1, 1))
              (relu): ReLU(inplace=True)
              (fc2): Conv2d(120, 480, kernel_size=(1, 1), stride=(1, 1))
            )
            (3): ConvBNActivation(
              (0): Conv2d(480, 80, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(80, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
          )
        )
        (2): InvertedResidual(
          (block): Sequential(
            (0): ConvBNActivation(
              (0): Conv2d(80, 480, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (1): ConvBNActivation(
              (0): Conv2d(480, 480, kernel_size=(5, 5), stride=(1, 1),
padding=(2, 2), groups=480, bias=False)
              (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Hardswish()
            )
            (2): SqueezeExcitation(
              (fc1): Conv2d(480, 120, kernel_size=(1, 1), stride=(1, 1))
              (relu): ReLU(inplace=True)
              (fc2): Conv2d(120, 480, kernel_size=(1, 1), stride=(1, 1))
            )
            (3): ConvBNActivation(
              (0): Conv2d(480, 80, kernel_size=(1, 1), stride=(1, 1),
bias=False)
              (1): BatchNorm2d(80, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
              (2): Identity()
            )
```

```
          )
        )
        (3): ConvBNActivation(
          (0): Conv2d(80, 480, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): Hardswish()
        )
      )
    )
    (extra): ModuleList(
      (0): Sequential(
        (0): ConvBNActivation(
          (0): Conv2d(480, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (1): ConvBNActivation(
          (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1,
1), groups=256, bias=False)
          (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (2): ConvBNActivation(
          (0): Conv2d(256, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(512, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
      )
      (1): Sequential(
        (0): ConvBNActivation(
          (0): Conv2d(512, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(128, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (1): ConvBNActivation(
          (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1,
1), groups=128, bias=False)
          (1): BatchNorm2d(128, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (2): ConvBNActivation(
```

```
          (0): Conv2d(128, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
      )
      (2): Sequential(
        (0): ConvBNActivation(
          (0): Conv2d(256, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(128, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (1): ConvBNActivation(
          (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1,
1), groups=128, bias=False)
          (1): BatchNorm2d(128, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (2): ConvBNActivation(
          (0): Conv2d(128, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
      )
      (3): Sequential(
        (0): ConvBNActivation(
          (0): Conv2d(256, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(64, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (1): ConvBNActivation(
          (0): Conv2d(64, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1),
groups=64, bias=False)
          (1): BatchNorm2d(64, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
        (2): ConvBNActivation(
          (0): Conv2d(64, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
          (1): BatchNorm2d(128, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
          (2): ReLU6(inplace=True)
        )
```

```
      )
    )
  )
  (anchor_generator): DefaultBoxGenerator(aspect_ratios=[[2, 3], [2, 3], [2, 3],
[2, 3], [2, 3], [2, 3]], clip=True, scales=[0.2, 0.35, 0.5, 0.65, 0.8, 0.95,
1.0], steps=None)
  (head): SSDLiteHead(
    (classification_head): SSDLiteClassificationHead(
      (module_list): ModuleList(
        (0): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(672, 672, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=672, bias=False)
            (1): BatchNorm2d(672, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(672, 546, kernel_size=(1, 1), stride=(1, 1))
        )
        (1): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(480, 480, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=480, bias=False)
            (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(480, 546, kernel_size=(1, 1), stride=(1, 1))
        )
        (2): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=512, bias=False)
            (1): BatchNorm2d(512, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(512, 546, kernel_size=(1, 1), stride=(1, 1))
        )
        (3): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=256, bias=False)
            (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
```

```
          (1): Conv2d(256, 546, kernel_size=(1, 1), stride=(1, 1))
        )
        (4): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=256, bias=False)
            (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(256, 546, kernel_size=(1, 1), stride=(1, 1))
        )
        (5): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=128, bias=False)
            (1): BatchNorm2d(128, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(128, 546, kernel_size=(1, 1), stride=(1, 1))
        )
      )
    )
    (regression_head): SSDLiteRegressionHead(
      (module_list): ModuleList(
        (0): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(672, 672, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=672, bias=False)
            (1): BatchNorm2d(672, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(672, 24, kernel_size=(1, 1), stride=(1, 1))
        )
        (1): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(480, 480, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=480, bias=False)
            (1): BatchNorm2d(480, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(480, 24, kernel_size=(1, 1), stride=(1, 1))
        )
        (2): Sequential(
```

```
          (0): ConvBNActivation(
            (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=512, bias=False)
            (1): BatchNorm2d(512, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(512, 24, kernel_size=(1, 1), stride=(1, 1))
        )
        (3): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=256, bias=False)
            (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(256, 24, kernel_size=(1, 1), stride=(1, 1))
        )
        (4): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=256, bias=False)
            (1): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(256, 24, kernel_size=(1, 1), stride=(1, 1))
        )
        (5): Sequential(
          (0): ConvBNActivation(
            (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=128, bias=False)
            (1): BatchNorm2d(128, eps=0.001, momentum=0.03, affine=True,
track_running_stats=True)
            (2): ReLU6(inplace=True)
          )
          (1): Conv2d(128, 24, kernel_size=(1, 1), stride=(1, 1))
        )
      )
    )
  )
  (transform): GeneralizedRCNNTransform(
      Normalize(mean=[0.5, 0.5, 0.5], std=[0.5, 0.5, 0.5])
      Resize(min_size=(320,), max_size=320, mode='bilinear')
  )
)
```

### 13.0.2 Show Output

[ ]:
```
device = getDevice()
device
```

[ ]:
```
device(type='cuda')
```

[ ]:
```
from matplotlib import pyplot as plt
import matplotlib.patches as patches


def show_output_with_bbox(img, bboxes, labels, scores):

    print(img.shape)

    img_height = img.shape[0]
    img_width = img.shape[1]

    fig, ax = plt.subplots(figsize=(8,8))

    ax.imshow(img)

    for box, class_name, score in zip(bboxes, labels, scores):

        if score > 0.1:

            class_name = class_dict[class_name]
            xmin, ymin, xmax, ymax = box
            xmin, ymin, width, height = [x for x in [xmin, ymin, (xmax-xmin),␣
↪(ymax-ymin)]]
            rect = patches.Rectangle((xmin, ymin), width, height, linewidth=1,␣
↪edgecolor='r', facecolor='none')
            ax.add_patch(rect)
            ax.text(xmin, ymin, f"{class_name, np.round(score, 2)}", color='r',␣
↪fontsize='large')

            # {'xx-small', 'x-small', 'small', 'medium', 'large', 'x-large',␣
↪'xx-large'}

    plt.show()



################################################################################
```

[ ]:
```
test_images = VALID_DF['filename'].unique().tolist()
train_images = TRAIN_DF['filename'].unique().tolist()

for index, image in enumerate(test_images):

    if index == 10:
        break



    class_dict = {
        0:"background",
        1:"with_mask",
        2:"without_mask",
        3:"mask_weared_incorrect"
    }


    img = cv2.imread(test_images[index])
    img = cv2.resize(img, (320, 320))
    img = cv2.cvtColor(img, cv2.COLOR_BGRA2RGB).astype(np.float32)
    img = img/255.0


    SSD_MODEL.eval()
    img_tensor_float = torch.tensor(img, dtype=torch.float32).permute(2,0,1).
↪unsqueeze(dim=0).to(device)
    out = SSD_MODEL(img_tensor_float)

    keepIndex = torchvision.ops.nms(out[0]['boxes'], out[0]['scores'],␣
↪iou_threshold=0.2)

    bboxTensors = out[0]['boxes'][keepIndex]
    labelsTensors = out[0]['labels'][keepIndex]
    scoresTensors = out[0]['scores'][keepIndex]


    boxes = bboxTensors.detach().cpu().numpy().astype(np.int)
    labels = labelsTensors.detach().cpu().numpy()
    scores = scoresTensors.detach().cpu().numpy()

    show_output_with_bbox(img, boxes, labels, scores)
```

### 13.0.3 From Single Image

[ ]:
```
from matplotlib import pyplot as plt
import matplotlib.patches as patches


def show_output_with_bbox(img, bboxes, labels, scores):

    print(img.shape)

    img_height = img.shape[0]
    img_width = img.shape[1]

    fig, ax = plt.subplots(figsize=(8,8))

    ax.imshow(img)

    for box, class_name, score in zip(bboxes, labels, scores):

        if score > 0.2:

            class_name = class_dict[class_name]
            xmin, ymin, xmax, ymax = box
            xmin, ymin, width, height = [x for x in [xmin, ymin, (xmax-xmin),␣
↪(ymax-ymin)]]
            rect = patches.Rectangle((xmin, ymin), width, height, linewidth=1,␣
↪edgecolor='r', facecolor='none')
            ax.add_patch(rect)
            ax.text(xmin, ymin, f"{class_name, np.round(score, 2)}", color='r',␣
↪fontsize='large')

            # {'xx-small', 'x-small', 'small', 'medium', 'large', 'x-large',␣
↪'xx-large'}

    plt.show()




################################################################################
```

[ ]:
```
imagePath = "/content/Screenshot from 2021-10-14 21-25-39.png"

img = cv2.imread(imagePath)
img = cv2.resize(img, (320, 320))
img = cv2.cvtColor(img, cv2.COLOR_BGRA2RGB).astype(np.float32)
img = img/255.0
```

```
SSD_MODEL.eval()
img_tensor_float = torch.tensor(img, dtype=torch.float32).permute(2,0,1).
 ↪unsqueeze(dim=0).to(device)
out = SSD_MODEL(img_tensor_float)

keepIndex = torchvision.ops.nms(out[0]['boxes'], out[0]['scores'],␣
 ↪iou_threshold=0.1)

bboxTensors = out[0]['boxes'][keepIndex]
labelsTensors = out[0]['labels'][keepIndex]
scoresTensors = out[0]['scores'][keepIndex]


boxes = bboxTensors.detach().cpu().numpy().astype(np.int)
labels = labelsTensors.detach().cpu().numpy()
scores = scoresTensors.detach().cpu().numpy()

show_output_with_bbox(img, boxes, labels, scores)
```

```
(320, 320, 3)
```


## 14 Deployed Heroku API

[ ]:
```
import requests

file = "/content/IMG_20211011_022805_1.jpg"
resp = requests.post("https://facemask-detection-api.herokuapp.com/predict",␣
↪files={'filename': open(file, 'rb')})
```

[ ]:
```
resp.content
```

[ ]:
```
b'{"result":{"boxes":[[100,151,234,306],[116,195,219,304],[68,86,255,291],[133,1
58,226,273],[114,183,157,231],[203,141,227,166],[218,142,238,164],[113,160,163,2
```

21],[127,182,189,219],[129,157,185,217],[135,162,204,201],[205,130,226,151],[111
,290,205,320],[131,188,163,214],[149,159,198,214],[139,196,193,235],[192,170,234
,259],[89,286,117,320],[137,177,224,225],[239,123,258,146],[110,149,203,246],[15
5,207,183,242],[75,108,153,248],[239,140,257,158],[135,197,173,230],[198,159,232
,221],[196,125,213,147],[221,120,239,141],[112,250,211,301],[139,208,166,237],[2
38,112,263,140],[197,139,216,164],[237,130,260,153],[168,165,227,248],[78,153,17
1,279],[200,142,221,162],[215,129,238,152],[144,177,198,237],[189,197,233,231],[
105,6,241,146],[141,293,227,320],[210,140,245,166],[146,152,207,185],[157,255,23
8,302],[239,152,256,172],[218,99,243,131],[21,122,130,235],[201,147,223,166],[16
3,165,221,204],[84,296,104,320],[121,206,213,261],[45,70,169,213],[45,284,101,32
0],[192,184,235,214],[205,113,226,137],[196,142,224,163],[62,292,82,320],[138,19
0,222,240],[213,135,242,155],[116,196,152,227],[204,141,235,165],[131,168,175,19
7],[221,128,258,151],[212,146,240,169],[10,140,132,284],[126,149,196,196],[170,1
98,200,233],[220,152,238,172],[177,295,242,320],[36,178,283,314],[155,187,185,21
8],[23,132,193,309],[237,89,264,130],[194,112,213,137],[146,213,194,261],[194,13
9,228,163],[72,36,204,174],[209,147,243,190],[256,120,281,156],[169,106,193,140]
,[188,133,219,153],[102,174,144,233],[157,283,233,309],[35,0,284,181],[238,102,2
59,133],[84,18,192,140],[191,286,242,309],[143,7,279,150],[117,279,210,311],[202
,170,231,199],[23,85,132,202],[179,191,229,240],[256,134,281,172],[114,188,174,2
42],[93,292,160,320],[175,140,205,168],[77,77,202,224],[229,119,276,163],[103,21
0,154,250],[0,50,22,107],[254,103,281,143],[212,83,256,133],[99,183,143,275],[24
0,168,256,193],[0,161,21,215],[3,82,155,320],[211,164,237,207],[26,284,83,320],[
121,223,213,279],[168,120,238,265],[0,49,105,171],[161,113,180,140],[57,298,103,
320],[161,162,210,222],[59,273,84,320],[160,98,180,128],[194,137,224,156],[193,1
02,211,127],[44,290,66,320],[0,129,21,183],[0,145,21,199],[178,209,242,252],[125
,153,177,184],[219,19,249,66],[0,191,22,248],[118,114,320,320],[217,297,271,320]
,[198,147,229,180],[0,112,21,167],[234,18,266,66],[0,179,21,230],[45,8,174,153],
[84,294,132,320],[0,96,21,152],[124,185,189,251],[209,10,258,73],[172,95,193,126
],[240,156,260,185],[227,99,277,149],[14,35,147,182],[137,209,169,253],[240,1,26
2,32],[118,50,232,167],[207,102,224,125],[0,0,168,252],[219,156,238,181],[107,20
,136,66],[29,298,92,320],[123,20,153,66],[54,48,161,169],[0,33,22,91],[72,296,11
6,320],[0,10,29,76],[257,148,280,186],[214,162,247,212],[0,215,24,284],[139,20,1
68,66],[203,20,232,65],[0,0,34,55],[255,0,278,33],[155,20,184,66],[224,228,270,3
13],[222,92,242,122],[224,1,245,32],[187,20,216,65],[0,70,28,141],[200,144,229,1
89],[171,20,200,66],[221,64,244,104],[183,87,224,132],[119,185,149,222],[219,111
,250,133],[193,10,242,72],[217,220,308,316],[97,10,145,73],[199,246,235,305],[8,
7,90,150],[113,10,161,73],[27,289,53,320],[129,10,177,72],[249,54,320,168],[95,1
94,156,295],[11,285,36,320],[145,10,193,72],[208,1,229,32],[151,0,320,256],[177,
10,225,72],[161,10,209,72],[176,1,197,32],[160,1,181,32],[144,1,165,32],[192,1,2
13,32],[197,86,239,131],[128,1,149,32],[205,67,228,104],[112,1,133,32],[238,0,32
0,91],[210,19,317,137],[227,78,276,136],[203,285,231,320],[208,149,235,171],[236
,297,297,320],[208,91,236,116],[102,278,170,311],[223,8,275,74],[157,63,181,106]
,[235,59,261,104],[44,34,286,312],[189,67,212,105],[76,0,213,89],[8,156,40,220],
[164,82,206,132],[0,0,29,34],[172,65,196,105],[90,19,121,66],[297,36,320,116],[6
,183,41,255],[0,6,20,43],[124,60,149,105],[140,60,165,105],[0,0,84,91],[137,291,
169,320],[149,290,182,320],[12,3,35,39],[228,131,273,178],[61,62,85,105],[217,16
7,234,196],[108,61,133,104],[224,91,253,116],[298,157,320,239],[247,12,283,70],[

269,0,295,35],[217,0,256,35],[218,73,247,118],[77,279,114,320],[80,10,130,73],[1
70,124,193,151],[3,208,44,294],[189,81,215,119],[232,0,272,35],[208,279,259,320]
,[95,1,117,32],[7,45,41,113],[240,179,258,209],[172,0,309,89],[257,161,280,198],
[251,81,281,132],[144,102,164,136],[210,109,255,149],[142,86,165,124],[0,265,22,
320],[48,1,70,33],[203,80,229,117],[201,0,240,35],[170,292,195,320],[169,0,208,3
5],[153,0,192,35],[137,0,176,35],[185,0,224,35],[121,0,160,35],[297,0,320,67],[1
05,0,144,35],[8,124,40,188],[41,271,68,320],[77,0,103,33],[155,76,184,121],[241,
108,292,167],[92,61,117,105],[4,0,45,37],[1,299,65,320]],"labels":[2,2,2,2,2,1,1
,2,2,2,2,1,1,2,2,2,2,1,2,1,2,2,2,1,2,2,1,1,2,2,1,1,1,2,2,2,1,2,2,2,1,1,2,2,1,1,2
,3,2,1,2,2,1,2,1,3,1,2,1,2,3,2,1,3,2,2,2,1,2,2,2,2,1,1,2,2,2,3,1,1,1,2,2,2,1,2,2
,2,2,2,2,2,1,2,1,2,2,1,2,1,1,1,2,1,1,2,3,1,2,2,2,1,1,2,1,1,1,1,1,1,1,2,2,1,1,2,1
,3,1,1,1,2,1,1,2,1,1,1,1,2,2,1,2,1,1,1,1,1,1,2,1,1,1,1,3,1,1,1,1,1,1,1,1,1,1,1,2
,1,1,1,2,1,1,1,1,2,2,1,1,1,2,2,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,1,1,1,1,1,2,1,1,1
,3,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,3,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1],"scores":[0.99818354845047,0.164
71607983112335,0.01440907921642065,0.007808068301528692,0.007485716138035059,0.0
05793731193989515,0.0051054866053164005,0.005057410802692175,0.00503470562398433
7,0.0049173240549862385,0.004815001972019672,0.004743639845401049,0.004616092424
839735,0.004407483618706465,0.004347012843936682,0.004329334478825331,0.00423215
189948678,0.004178541246801615,0.004153837449848652,0.004149815998971462,0.00406
3377156853676,0.003933472093194723,0.003872279310598969 5,0.003706706454977393,0.
0035758190788328648,0.0034870589151978493,0.003429180243983865,0.003396775340661
4065,0.0033336381893604994,0.00332901510410010 8,0.0032840301282703876,0.00327407
00989961624,0.003200782462954521,0.003189387964084744 5,0.003150433301925659,0.00
3148614894598722 5,0.0030055548761039972 3,0.002984946826472878 5,0.0029739174060523
51,0.002918317448347807,0.002804357325658202,0.002784165088087320 3,0.00266054482
19925165,0.002642608014866709 7,0.002630608156323433,0.002597722923383116 7,0.0025
95288446173072,0.002592343371361494,0.002554713748395443,0.002535599982365966,0.
002502132672816515,0.002488792641088366 5,0.002419411670416593 6,0.002413894981145
8588,0.002413755981251597 4,0.002396832685917616,0.002390017034485936,0.002377409
0223014355,0.002375279786065221,0.002350834663957357 4,0.002347283530980348 6,0.00
232070655176520348,0.002311875810846686 4,0.002297634957358241,0.00229656975716352
46,0.002293499885126948 4,0.002282439498230815,0.002258974127471447,0.00220302469
09707785,0.002193320542573929,0.002183700911700725 6,0.002159257885068655,0.00214
5852195098996,0.002099222969263792,0.002096094656735658 6,0.002078442135825753,0.
00206529232673347,0.002062481362372637,0.002024046843871474 3,0.00202184799127280
7,0.002019363222643733,0.001984503120183944 7,0.001965213799849152 6,0.00191594567
14987755,0.001875510090030729 8,0.001830424065701663 5,0.001796728582121431 8,0.001
7821724759414792,0.001775349141098559,0.001770142116583883 8,0.001751522184349596
5,0.001748403417877853,0.001746531808748841 3,0.001745151472277939 3,0.00174089765
64183831,0.001739571802318096 2,0.001712842029519379 1,0.001706850365735590 5,0.001
68997619766742,0.001686563598923385 1,0.001673737308010459,0.001670124824158847 3,
0.001660584472119808 2,0.001657890155911445 6,0.001650237594731152,0.0016429062234
23779,0.001636307220906019 2,0.001634732470847666 3,0.001625317498110234 7,0.001600
1722542569041,0.001599229755811393 3,0.001597438938915729 5,0.001593581284396350 4,
0.001591965672560036 2,0.001586665282957255 8,0.001575852627865970 1,0.001574232708
6627483,0.001548997592180967 3,0.001547137275338173,0.001535300131205469 37,0.00152

32519945129752,0.001514149596914649,0.001507014269092325,0.0015022751176729798,
0.0015014008386060596,0.0014999363338574767,0.0014962218701839447,0.001496217329
9863935,0.0014923999551683664,0.0014787133550271392,0.0014750001719221473,0.0014
697472797706723,0.0014632644597440958,0.001461434061639011,0.0014608936617150903
,0.0014431339222937822,0.0014232895337045193,0.0014157508267089725,0.00141174893
36058497,0.0014091441407799 72,0.0014082142151892185,0.0014045001007616 52,0.00140
11674793437123,0.0014011637540534139,0.0013938621850684285,0.0013840317260473967
,0.0013821367174386978,0.0013742017326876521,0.0013721405994147062,0.00136986514
5534277,0.0013648574240505695,0.0013627833686769009,0.0013619204983115196,0.0013
617892982438207,0.0013569580623880029,0.0013567578280344605,0.0013562017120420933
,0.0013521746732294 56,0.0013458242174237967,0.001334638916887343,0.001332846237
34653,0.0013309161877259612,0.0013284091837704182,0.0013235552469268441,0.001313
957036472857,0.0013130586594343185,0.0013127595884725451,0.0013122461969032884,0
.0013020304031670094,0.0012994048884138465,0.0012987260706722736,0.0012928050709
88834,0.0012861113063991 07,0.0012846011668443 68,0.0012826028978452086,0.00127838
10961991549,0.0012761630350723863,0.0012731442693620 92,0.0012644161470234394,0.0
012639377964660525,0.0012577680172398686,0.0012569597456604242,0.001249959226697
6833,0.0012498770374804735,0.0012492816895246506,0.0012453026138246 06,0.00124503
46257537603,0.0012399195693433285,0.0012386472662910 82,0.0012385691516101 36,0.00
12384398141875863,0.0012375334044918418,0.0012374531943351 03,0.0012360832188278 4
37,0.0012332525802776217,0.0012315895874053 24,0.0012287893332540 99,0.0012276927 0
09224892,0.0012267718557268381,0.0012253267923370004,0.0012192405993118882,0.001
218406017869711,0.0012091726530343294,0.0012067040661349893,0.0012010800419375 3,
0.0011982260039076 21,0.0011921288678422 57,0.0011858935467898846,0.0011845453409 4
78068,0.0011827201815322042,0.0011815419420599937,0.0011806224938482046,0.001177
8734624385834,0.0011756250169128 18,0.0011726247612386942,0.0011710436083376408,0
.0011654995614662766,0.0011623450554907322,0.0011597849661484 36,0.0011573779629 5
43424,0.0011538844555616379,0.0011533374199643731,0.0011476489016786218,0.001147
0596073195338,0.0011392548913136125,0.0011305513326078653,0.0011298366589471698,
0.0011274936841800809,0.0011259699240326881,0.0011199618456885219,0.001119256718
0842161,0.0011175676481798291,0.0011106460588052869,0.0011071154149249196,0.0011
065710568800569,0.0011043440317735076,0.0011026815045624971,0.0010999870656616985
8,0.0010975177865475416,0.0010936460457742214,0.0010910271666944027,0.0010889532
277360559,0.0010823864722624421,0.0010737490374594927,0.0010713861556723714,0.00
10675836820155382,0.0010652048513293266,0.0010643661953508854,0.0010634816717356
443,0.0010626617586240172,0.0010548051213845 61,0.0010538556380197406,0.001048661
65202111,0.0010480402270331979,0.0010408080415800214,0.0010378758888691664,0.001
0377306025475264,0.0010375411948189 14,0.0010374597047093635,0.0010355329141020775
,0.0010343752801418304,0.0010325741022825241,0.0010296597611159086,0.00102814624
55168366,0.0010207975283265114,0.0010206286096945405,0.0010200011311098933,0.001
0193875059485435,0.0010099031496793032,0.0010091960430145264]},"success":true}\n
'

[ ]: