# Baseline решения

## импорт YOLO

%load\_ext autoreload  
%autoreload 2

The autoreload extension is already loaded. To reload it, use:  
 %reload\_ext autoreload

import ssl  
ssl.\_create\_default\_https\_context = ssl.\_create\_stdlib\_context

import matplotlib.pyplot as plt  
import cv2

# !pip3 install -U ultralytics --user

import ultralytics

from ultralytics import settings  
settings.update({"wandb": False, "tensorboard": False, "clearml":False, "mlflow":False, "neptune":False, 'comet': False, 'dvc': False, 'hub': False })

print(settings)

{'settings\_version': '0.0.4', 'datasets\_dir': 'D:\\Yandex Disk\\Компьютерное зрение\\практики очные\\Короткий CV COurse\\datasets', 'weights\_dir': 'weights', 'runs\_dir': 'runs', 'uuid': 'e6e830fcb721a7c37fbbf7d832aecf0518808b2940252e20b03bd1656987def9', 'sync': True, 'api\_key': '', 'openai\_api\_key': '', 'clearml': False, 'comet': False, 'dvc': False, 'hub': False, 'mlflow': False, 'neptune': False, 'raytune': True, 'tensorboard': False, 'wandb': False}

Также ultralytics предоставляют собственный интерфейс проверки вычислителей.

ultralytics.checks()

Ultralytics YOLOv8.2.77 Python-3.12.3 torch-2.4.0 CPU (Intel Core(TM) i5-7400 3.00GHz)  
Setup complete (4 CPUs, 15.9 GB RAM, 562.7/931.5 GB disk)

from ultralytics import YOLO  
  
model = YOLO("yolov8n.pt")

model.info()

YOLOv8n summary: 225 layers, 3,157,200 parameters, 0 gradients, 8.9 GFLOPs

(225, 3157200, 0, 8.8575488)

## Импорты

import os  
import numpy as np  
import pandas as pd  
from PIL import Image  
from pathlib import Path  
import matplotlib.pyplot as plt  
import matplotlib.patches as patches

from urllib.request import urlretrieve  
from torchvision.datasets.utils import download\_and\_extract\_archive

## Пути к набору данных

### путь к набору данных

root\_path = Path(os.getcwd())   
print(list(root\_path.iterdir()))

[WindowsPath('D:/Yandex Disk/ObjDetV2/.ipynb\_checkpoints'), WindowsPath('D:/Yandex Disk/ObjDetV2/archive (22).zip'), WindowsPath('D:/Yandex Disk/ObjDetV2/LU3M6TGT\_yolo\_format'), WindowsPath('D:/Yandex Disk/ObjDetV2/WS8\_YOLOv8 API.ipynb'), WindowsPath('D:/Yandex Disk/ObjDetV2/yolov8n.pt')]

dataset\_path = root\_path / 'craters' #'LU3M6TGT\_yolo\_format'  
print(list(dataset\_path.iterdir()))

[WindowsPath('D:/Yandex Disk/ObjDetV2/craters/test'), WindowsPath('D:/Yandex Disk/ObjDetV2/craters/train'), WindowsPath('D:/Yandex Disk/ObjDetV2/craters/valid')]

### путь к тренировочной и к валидационной части

ext = 'jpg' #'png'

images\_path = dataset\_path / 'train' / 'images'  
labels\_path = dataset\_path / 'train' / 'labels'  
  
labels\_path = list(labels\_path.glob('\*\*/\*.txt'))  
images\_path = list(images\_path.glob(f'\*\*/\*.{ext}'))

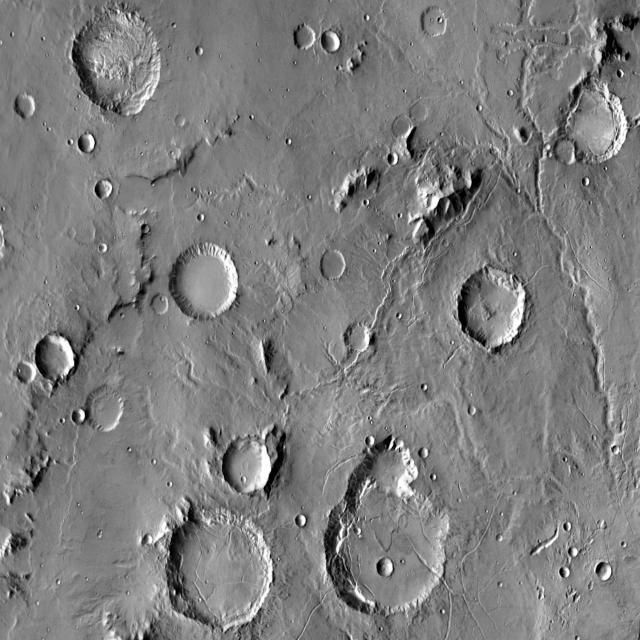
val\_image\_path = dataset\_path / 'valid' / 'images'  
val\_labels\_path = dataset\_path / 'valid' / 'labels'  
  
val\_labels\_path = list(val\_labels\_path.glob('\*\*/\*.txt'))  
val\_image\_path = list(val\_image\_path.glob(f'\*\*/\*.{ext}'))

test\_images\_path = dataset\_path / 'test' / 'images'  
  
test\_images\_path = list(test\_images\_path.glob(f'\*\*/\*.{ext}'))

test\_images\_path[1]

WindowsPath('D:/Yandex Disk/ObjDetV2/craters/test/images/015\_png.rf.7d5b2091b6339c9480a171a59c52c3b9.jpg')

Image.open(images\_path[1])



### описание набора данных

data\_yaml = dataset\_path / 'data.yaml'  
data\_yaml

WindowsPath('D:/Yandex Disk/ObjDetV2/craters/data.yaml')

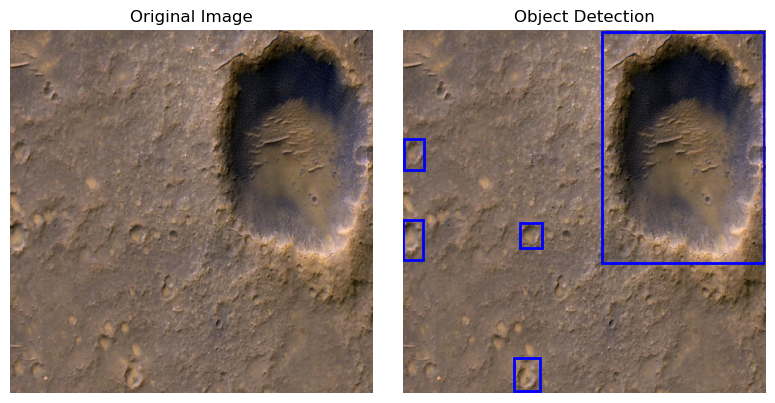
import yaml  
  
data = {  
 f'train': str(dataset\_path / 'train' / 'images'),  
 f'val': str(dataset\_path / 'valid' / 'images'),  
 'nc': 1,  
 'names': ['0']  
}  
  
with open(data\_yaml, 'w') as file:  
 yaml.safe\_dump(data, file)  
  
print("New YAML file created successfully.")  
  
with open(data\_yaml, 'r') as file:  
 data=yaml.safe\_load( file)  
print(data)

New YAML file created successfully.  
{'names': ['0'], 'nc': 1, 'train': 'D:\\Yandex Disk\\ObjDetV2\\craters\\train\\images', 'val': 'D:\\Yandex Disk\\ObjDetV2\\craters\\valid\\images'}

## Построение примеров данных

def yolo2coco(xc, yc, w, h, image\_width, image\_height):  
 xc, w = xc\*image\_width, w\*image\_width  
 yc, h = yc\*image\_height, h\*image\_height  
 xmin = xc - w//2  
 ymin = yc - h//2  
 return xmin,ymin,w, h  
  
def plot\_bbox(image\_pth, ann\_pth = None, predicted\_boxes = None, pred\_classes = None):  
 image = np.asarray(Image.open(image\_pth))  
 height, width = image.shape[:2]  
  
 if ann\_pth == None:  
 ann\_name = image\_pth.parts[-1][:-4]+'.txt'  
 ann\_pth = Path(\*image\_pth.parts[:-2]) / 'labels'/ann\_name  
 data = pd.read\_csv(ann\_pth, sep=" ", header=None).values   
   
 bbox = np.asarray(data)[:,1:5]  
   
 fig, axs = plt.subplots(1, 2, figsize = (8,4))  
 image\_ = np.copy(image).astype(float)/image.max()  
   
 # Ground Truth   
 axs[0].imshow(np.clip(image\_,0,1), 'gray')  
 axs[0].axis('off'); axs[0].set\_title('Original Image')  
 #----------------------------  
   
 # Object Detection BBoxes  
 axs[1].imshow(np.clip(image\_,0,1), 'gray')  
   
 for box in bbox:  
 xmin,ymin,w,h = yolo2coco(\*box, width,height)  
 bb = patches.Rectangle(xy = (xmin,ymin),   
 width = w,height = h,   
 linewidth=2, edgecolor="blue", facecolor="none")  
 axs[1].add\_patch(bb)   
  
 if predicted\_boxes != None:  
 if pred\_classes == None:  
 pred\_classes = ['0']\*len(predicted\_boxes)  
 for cls,box in zip(np.array(pred\_classes),np.array(predicted\_boxes)):  
 xmin,ymin,w,h = yolo2coco(\*box, width,height)  
 bb = patches.Rectangle(xy = (xmin,ymin),   
 width = w,height = h,   
 linewidth=2, edgecolor="red", facecolor="none")  
 axs[1].add\_patch(bb)  
  
  
 axs[1].text(xmin,ymin,str(cls),fontsize=10, color="green", weight="bold")  
   
 axs[1].axis('off'); axs[1].set\_title('Object Detection')  
 plt.tight\_layout()  
 plt.show()

plot\_bbox(images\_path[47] )



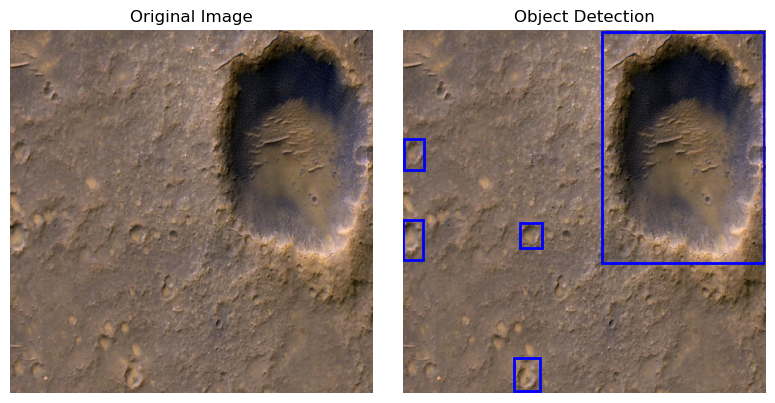
## Проверка модели и построение результата

model = YOLO("yolov8n.pt")  
  
results = model(images\_path[47])

image 1/1 D:\Yandex Disk\ObjDetV2\craters\train\images\mars\_crater--35-\_jpg.rf.826bc4b2a3a69c48144834f40d92439f.jpg: 640x640 (no detections), 188.0ms  
Speed: 6.0ms preprocess, 188.0ms inference, 1.0ms postprocess per image at shape (1, 3, 640, 640)

def plot\_result(result):  
 image\_pth = Path(result.path)  
 image = np.asarray(Image.open(image\_pth))  
 height, width = image.shape[:2]  
  
 ann\_name = image\_pth.parts[-1][:-4]+'.txt'  
 ann\_pth = Path(\*image\_pth.parts[:-2]) / 'labels'/ann\_name  
 data = pd.read\_csv(ann\_pth, sep=" ", header=None).values   
   
 bbox = np.asarray(data)[:,1:5]  
   
 fig, axs = plt.subplots(1, 2, figsize = (8,4))  
 image\_ = np.copy(image).astype(float)/image.max()  
   
 # Ground Truth   
 axs[0].imshow(np.clip(image\_,0,1), 'gray')  
 axs[0].axis('off'); axs[0].set\_title('Original Image')  
 #----------------------------  
   
 # Object Detection BBoxes  
 axs[1].imshow(np.clip(image\_,0,1), 'gray')  
   
 for box in bbox:  
 xmin,ymin,w,h = yolo2coco(\*box, width,height)  
 bb = patches.Rectangle(xy = (xmin,ymin),   
 width = w,height = h,   
 linewidth=2, edgecolor="blue", facecolor="none")  
 axs[1].add\_patch(bb)   
  
 predicted\_boxes = result.boxes.xywhn  
 pred\_classes = result.boxes.cls.long()  
 if pred\_classes == []:  
 pred\_classes = len(predicted\_boxes)\*['0']  
   
 if predicted\_boxes != None:  
 if pred\_classes == None:  
 pred\_classes = ['0']\*len(predicted\_boxes)  
   
 for cls,box in zip(np.array(pred\_classes),np.array(predicted\_boxes)):  
 xmin,ymin,w,h = yolo2coco(\*box, width,height)  
 bb = patches.Rectangle(xy = (xmin,ymin),   
 width = w,height = h,   
 linewidth=2, edgecolor="red", facecolor="none")  
 axs[1].add\_patch(bb)  
  
  
 axs[1].text(xmin,ymin,str(cls),fontsize=10, color="green", weight="bold")  
   
 axs[1].axis('off'); axs[1].set\_title('Object Detection')  
 plt.tight\_layout()  
 plt.show()

plot\_result(results[0] )



## Тренировка модели

model = YOLO("yolov8n.pt")  
results = model.train(data=data\_yaml,imgsz=640, epochs = 12)

New https://pypi.org/project/ultralytics/8.2.82 available Update with 'pip install -U ultralytics'  
Ultralytics YOLOv8.2.77 Python-3.12.3 torch-2.4.0 CPU (Intel Core(TM) i5-7400 3.00GHz)  
engine\trainer: task=detect, mode=train, model=yolov8n.pt, data=D:\Yandex Disk\ObjDetV2\craters\data.yaml, epochs=12, time=None, patience=100, batch=16, imgsz=640, save=True, save\_period=-1, cache=False, device=None, workers=8, project=None, name=train9, exist\_ok=False, pretrained=True, optimizer=auto, verbose=True, seed=0, deterministic=True, single\_cls=False, rect=False, cos\_lr=False, close\_mosaic=10, resume=False, amp=True, fraction=1.0, profile=False, freeze=None, multi\_scale=False, overlap\_mask=True, mask\_ratio=4, dropout=0.0, val=True, split=val, save\_json=False, save\_hybrid=False, conf=None, iou=0.7, max\_det=300, half=False, dnn=False, plots=True, source=None, vid\_stride=1, stream\_buffer=False, visualize=False, augment=False, agnostic\_nms=False, classes=None, retina\_masks=False, embed=None, show=False, save\_frames=False, save\_txt=False, save\_conf=False, save\_crop=False, show\_labels=True, show\_conf=True, show\_boxes=True, line\_width=None, format=torchscript, keras=False, optimize=False, int8=False, dynamic=False, simplify=False, opset=None, workspace=4, nms=False, lr0=0.01, lrf=0.01, momentum=0.937, weight\_decay=0.0005, warmup\_epochs=3.0, warmup\_momentum=0.8, warmup\_bias\_lr=0.1, box=7.5, cls=0.5, dfl=1.5, pose=12.0, kobj=1.0, label\_smoothing=0.0, nbs=64, hsv\_h=0.015, hsv\_s=0.7, hsv\_v=0.4, degrees=0.0, translate=0.1, scale=0.5, shear=0.0, perspective=0.0, flipud=0.0, fliplr=0.5, bgr=0.0, mosaic=1.0, mixup=0.0, copy\_paste=0.0, auto\_augment=randaugment, erasing=0.4, crop\_fraction=1.0, cfg=None, tracker=botsort.yaml, save\_dir=runs\detect\train9  
Overriding model.yaml nc=80 with nc=1  
  
 from n params module arguments   
 0 -1 1 464 ultralytics.nn.modules.conv.Conv [3, 16, 3, 2]   
 1 -1 1 4672 ultralytics.nn.modules.conv.Conv [16, 32, 3, 2]   
 2 -1 1 7360 ultralytics.nn.modules.block.C2f [32, 32, 1, True]   
 3 -1 1 18560 ultralytics.nn.modules.conv.Conv [32, 64, 3, 2]   
 4 -1 2 49664 ultralytics.nn.modules.block.C2f [64, 64, 2, True]   
 5 -1 1 73984 ultralytics.nn.modules.conv.Conv [64, 128, 3, 2]   
 6 -1 2 197632 ultralytics.nn.modules.block.C2f [128, 128, 2, True]   
 7 -1 1 295424 ultralytics.nn.modules.conv.Conv [128, 256, 3, 2]   
 8 -1 1 460288 ultralytics.nn.modules.block.C2f [256, 256, 1, True]   
 9 -1 1 164608 ultralytics.nn.modules.block.SPPF [256, 256, 5]   
 10 -1 1 0 torch.nn.modules.upsampling.Upsample [None, 2, 'nearest']   
 11 [-1, 6] 1 0 ultralytics.nn.modules.conv.Concat [1]   
 12 -1 1 148224 ultralytics.nn.modules.block.C2f [384, 128, 1]   
 13 -1 1 0 torch.nn.modules.upsampling.Upsample [None, 2, 'nearest']   
 14 [-1, 4] 1 0 ultralytics.nn.modules.conv.Concat [1]   
 15 -1 1 37248 ultralytics.nn.modules.block.C2f [192, 64, 1]   
 16 -1 1 36992 ultralytics.nn.modules.conv.Conv [64, 64, 3, 2]   
 17 [-1, 12] 1 0 ultralytics.nn.modules.conv.Concat [1]   
 18 -1 1 123648 ultralytics.nn.modules.block.C2f [192, 128, 1]   
 19 -1 1 147712 ultralytics.nn.modules.conv.Conv [128, 128, 3, 2]   
 20 [-1, 9] 1 0 ultralytics.nn.modules.conv.Concat [1]   
 21 -1 1 493056 ultralytics.nn.modules.block.C2f [384, 256, 1]   
 22 [15, 18, 21] 1 751507 ultralytics.nn.modules.head.Detect [1, [64, 128, 256]]   
Model summary: 225 layers, 3,011,043 parameters, 3,011,027 gradients, 8.2 GFLOPs  
  
Transferred 319/355 items from pretrained weights  
Freezing layer 'model.22.dfl.conv.weight'

train: Scanning D:\Yandex Disk\ObjDetV2\craters\train\labels.cache... 98 images, 9 backgrounds, 0 corrupt: 100%|███████

albumentations: Blur(p=0.01, blur\_limit=(3, 7)), MedianBlur(p=0.01, blur\_limit=(3, 7)), ToGray(p=0.01), CLAHE(p=0.01, clip\_limit=(1, 4.0), tile\_grid\_size=(8, 8))

val: Scanning D:\Yandex Disk\ObjDetV2\craters\test\labels.cache... 19 images, 0 backgrounds, 0 corrupt: 100%|██████████

Plotting labels to runs\detect\train9\labels.jpg...   
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum' automatically...   
optimizer: AdamW(lr=0.002, momentum=0.9) with parameter groups 57 weight(decay=0.0), 64 weight(decay=0.0005), 63 bias(decay=0.0)  
Image sizes 640 train, 640 val  
Using 0 dataloader workers  
Logging results to runs\detect\train9  
Starting training for 12 epochs...  
  
 Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

1/12 0G 1.752 3.367 1.413 30 640: 100%|██████████| 7/7 [00:57<00:00, 8.26  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:05<0

all 19 151 0.0144 0.543 0.0679 0.0358

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

2/12 0G 1.677 2.457 1.299 54 640: 100%|██████████| 7/7 [00:51<00:00, 7.42  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:03<0

all 19 151 0.765 0.119 0.205 0.104

Closing dataloader mosaic  
albumentations: Blur(p=0.01, blur\_limit=(3, 7)), MedianBlur(p=0.01, blur\_limit=(3, 7)), ToGray(p=0.01), CLAHE(p=0.01, clip\_limit=(1, 4.0), tile\_grid\_size=(8, 8))  
  
 Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

3/12 0G 1.614 2.167 1.249 50 640: 100%|██████████| 7/7 [00:38<00:00, 5.53  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:02<0

all 19 151 0.0327 0.682 0.175 0.107

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

4/12 0G 1.625 1.936 1.246 15 640: 100%|██████████| 7/7 [00:37<00:00, 5.34  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:02<0

all 19 151 0.0207 0.781 0.219 0.133

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

5/12 0G 1.291 5.522 1.055 0 640: 100%|██████████| 7/7 [00:37<00:00, 5.33  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:02<0

all 19 151 0.0958 0.358 0.197 0.119

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

6/12 0G 1.55 1.654 1.238 28 640: 100%|██████████| 7/7 [00:39<00:00, 5.59  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:03<0

all 19 151 0.0726 0.377 0.226 0.138

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

7/12 0G 1.441 1.897 1.289 2 640: 100%|██████████| 7/7 [00:39<00:00, 5.68  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:02<0

all 19 151 0.951 0.129 0.333 0.192

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

8/12 0G 1.477 1.636 1.244 5 640: 100%|██████████| 7/7 [00:42<00:00, 6.05  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:03<0

all 19 151 0.863 0.126 0.312 0.183

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

9/12 0G 1.407 1.56 1.151 32 640: 100%|██████████| 7/7 [00:41<00:00, 5.89  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:02<0

all 19 151 0.859 0.146 0.401 0.226

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

10/12 0G 1.5 1.522 1.157 10 640: 100%|██████████| 7/7 [00:39<00:00, 5.60  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:03<0

all 19 151 0.88 0.232 0.5 0.279

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

11/12 0G 1.499 1.576 1.181 20 640: 100%|██████████| 7/7 [00:39<00:00, 5.58  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:02<0

all 19 151 0.872 0.272 0.513 0.284

Epoch GPU\_mem box\_loss cls\_loss dfl\_loss Instances Size

12/12 0G 1.429 1.57 1.173 19 640: 100%|██████████| 7/7 [00:41<00:00, 5.95  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:03<0

all 19 151 0.783 0.286 0.512 0.282

12 epochs completed in 0.160 hours.  
Optimizer stripped from runs\detect\train9\weights\last.pt, 6.2MB  
Optimizer stripped from runs\detect\train9\weights\best.pt, 6.2MB  
  
Validating runs\detect\train9\weights\best.pt...  
Ultralytics YOLOv8.2.77 Python-3.12.3 torch-2.4.0 CPU (Intel Core(TM) i5-7400 3.00GHz)  
Model summary (fused): 168 layers, 3,005,843 parameters, 0 gradients, 8.1 GFLOPs

Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 1/1 [00:02<0

all 19 151 0.874 0.272 0.513 0.284  
Speed: 3.9ms preprocess, 118.5ms inference, 0.0ms loss, 13.2ms postprocess per image  
Results saved to runs\detect\train9

## Проверка результата на валидации

### все результаты

results = model.val(data=data\_yaml,imgsz=640)

Ultralytics YOLOv8.2.77 Python-3.12.3 torch-2.4.0 CPU (Intel Core(TM) i5-7400 3.00GHz)  
Model summary (fused): 168 layers, 3,005,843 parameters, 0 gradients, 8.1 GFLOPs

val: Scanning D:\Yandex Disk\ObjDetV2\craters\test\labels.cache... 19 images, 0 backgrounds, 0 corrupt: 100%|██████████  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 2/2 [00:02<0

all 19 151 0.874 0.272 0.513 0.284  
Speed: 2.8ms preprocess, 96.8ms inference, 0.0ms loss, 12.0ms postprocess per image  
Results saved to runs\detect\train92

### Целевой результат

print(f'map 50:95 = {results.box.map:.3f}') # map50-95

map 50:95 = 0.284

### Проверим что модель что то находит на тестовых данных

results = model(test\_images\_path[:], )

0: 640x640 2 0s, 91.2ms  
1: 640x640 1 0, 91.2ms  
2: 640x640 (no detections), 91.2ms  
3: 640x640 (no detections), 91.2ms  
4: 640x640 1 0, 91.2ms  
5: 640x640 (no detections), 91.2ms  
6: 640x640 (no detections), 91.2ms  
7: 640x640 (no detections), 91.2ms  
8: 640x640 (no detections), 91.2ms  
9: 640x640 1 0, 91.2ms  
10: 640x640 1 0, 91.2ms  
11: 640x640 (no detections), 91.2ms  
12: 640x640 (no detections), 91.2ms  
13: 640x640 (no detections), 91.2ms  
14: 640x640 (no detections), 91.2ms  
15: 640x640 1 0, 91.2ms  
16: 640x640 (no detections), 91.2ms  
17: 640x640 (no detections), 91.2ms  
18: 640x640 (no detections), 91.2ms  
Speed: 3.8ms preprocess, 91.2ms inference, 0.4ms postprocess per image at shape (1, 3, 640, 640)

Результаты модели можно проверить "на глаз" при желании.

## Как найти обученную модель для подачи на проверку.

* Модель можно найти в описании тренировки - параметр save\_dir например в виде =runs\detect\train9
* Также можно поменять директорию для сохранения модели как settings.update({'runs\_dir': '/path/to/runs'})
* В директории как правило будут два файла last.pt и best.pt. Нужен best.pt.
* Файл best.pt нужно сохранить под своим именем и сдать.
* Проверить модель можно так, как это показано ниже.
* Отметим, что загруженную модель также можно дообучить при желании.

Также отметим что путь к обученной модели можно получить следующим образом:

save\_dir = model.trainer.save\_dir   
best\_trained\_weights = save\_dir / 'weights' / 'best.pt'  
best\_trained\_weights

WindowsPath('runs/detect/train9/weights/best.pt')

model = YOLO(best\_trained\_weights) # load a partially trained model  
# results = model.train(data=data\_yaml,imgsz=640, epochs = 7)  
results = model.val(data=data\_yaml,imgsz=640)

Ultralytics YOLOv8.2.77 Python-3.12.3 torch-2.4.0 CPU (Intel Core(TM) i5-7400 3.00GHz)  
Model summary (fused): 168 layers, 3,005,843 parameters, 0 gradients, 8.1 GFLOPs

val: Scanning D:\Yandex Disk\ObjDetV2\craters\test\labels.cache... 19 images, 0 backgrounds, 0 corrupt: 100%|██████████  
 Class Images Instances Box(P R mAP50 mAP50-95): 100%|██████████| 2/2 [00:02<0

all 19 151 0.874 0.272 0.513 0.284  
Speed: 3.6ms preprocess, 113.3ms inference, 0.0ms loss, 12.9ms postprocess per image  
Results saved to runs\detect\val10

# Задание на итоговый проект

**Цель** - обучить нейронную сеть типа YOLO для решения проблемы поиска и определения краторов на изобажениях лунохода.

Действия:

1. Прочитать пример работы с набором данных.
2. Обучить модель детекции краторов, используюя целевую метрику map50:95
3. Выгрузить обученную модель (файл best.pt).
4. Переименовать модель в соответствии с фио на Английском языке и сдать ее.
5. Модель должна дать на тестовой части набора данных точность не ниже 40%. В противном случае баллы будут снижены в двое.
6. На решение задачи дается 2 попытки