目录

[Yaml版本说明 2](#_Toc111444259)

[数据集 2](#_Toc111444260)

[实验结果 6](#_Toc111444261)

[一些错误 7](#_Toc111444262)

# Yaml版本说明

Yolov5s：原版本，anchor=9

Yolov5s\_fusion：加入小目标特征融合层，k=12

是否使用解耦头在yolo.py中启动这句话

self.m = nn.ModuleList(DecoupledHead(x, nc, 1, anchors) for x in ch)

是否使用ASPP，只需要将SPP层改为ASPP，后边的参数改为[6,12,18]

BiFPN\_Concat2与Concat相同

Yolov5s\_BiFPN：使用BiFPN特征融合

yolov5s\_BiFPN\_CBAM：在头部加入CBAM注意力机制

是否可以把上采样改为反卷积操作实现上采样功能呢？

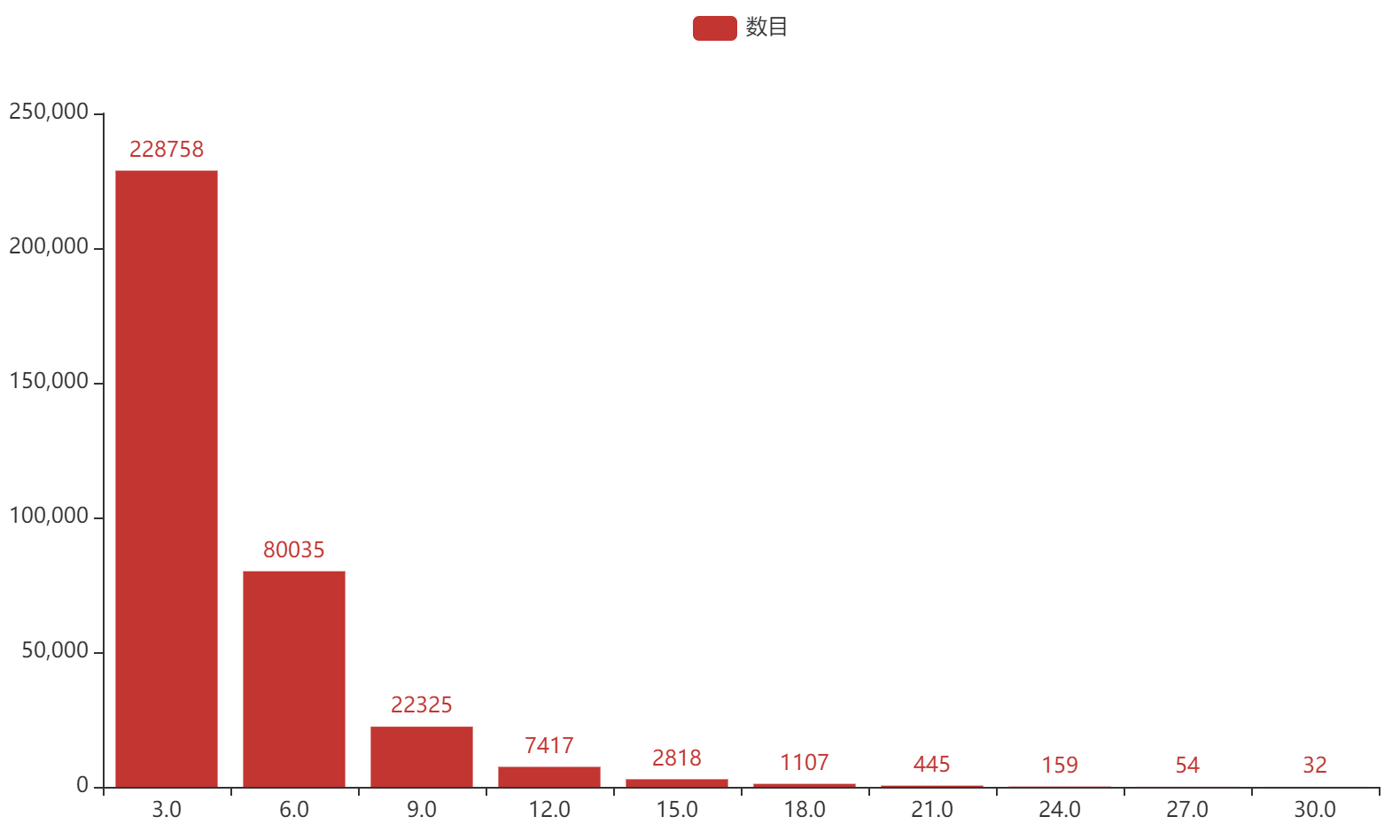
yolov5s\_BiFPN\_CBAM\_ghost：为什么轻量化后GLOPS更大呢？C3函数要更改

yolov5s\_BiFPN\_CBAM\_shuffle：GLOPS与s原版大小一致，参数明显降低

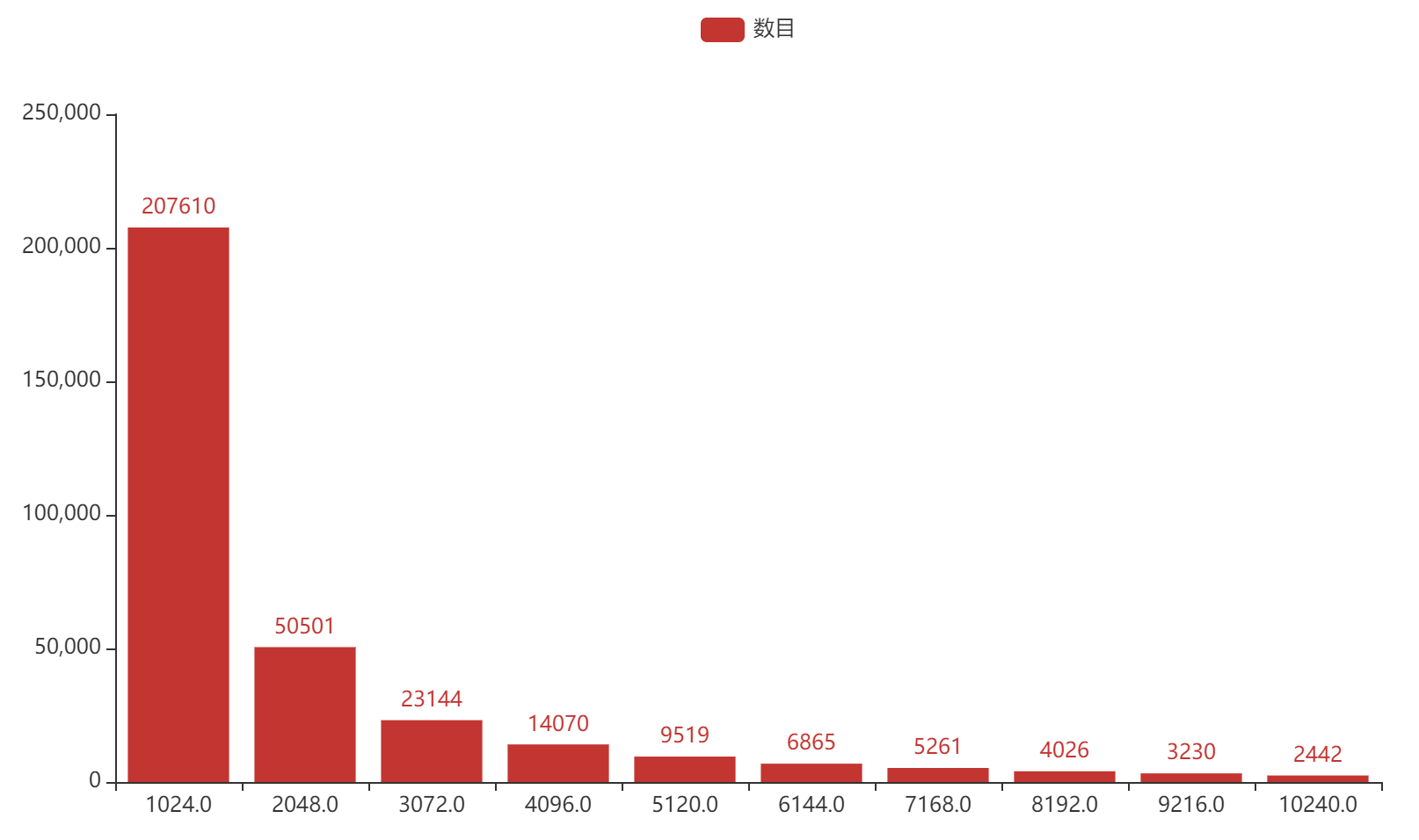
# 数据集

{'3': 144867, '1': 27059, '4': 24956, '5': 12875, '9': 29647, '2': 10480, '6': 4812, '7': 3246, '0': 79337, '8': 5926}

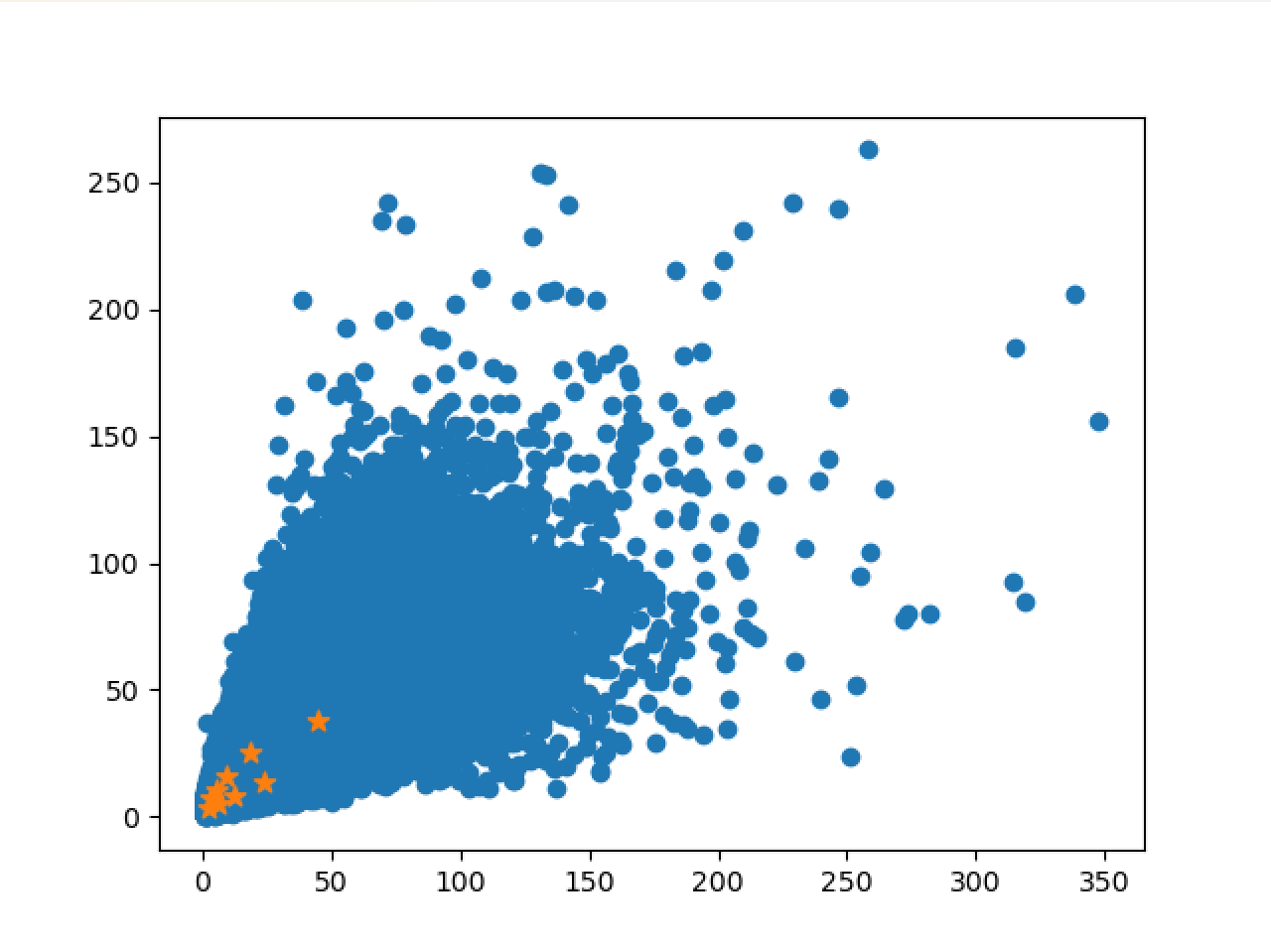
各个类别的标签数量



面积比开根号小于3%的认为是小目标 chart



面积小于32\*32的目标 chart1



数据集visdrone 6471张图片经过kmeans聚类后的图片，k=9

kmeans: [2, 3] [3, 7] [6, 5] [5, 10] [12, 8] [9, 16] [24, 13] [18, 25] [44, 37]

fitness: 0.74925, best possible recall: 0.99955

[[ 2.33820459 3.52 ]

[ 3.2 7.05882353]

[ 6.4 5.02857143]

[ 5.66666667 10.51428587]

[12.8 8.22857143]

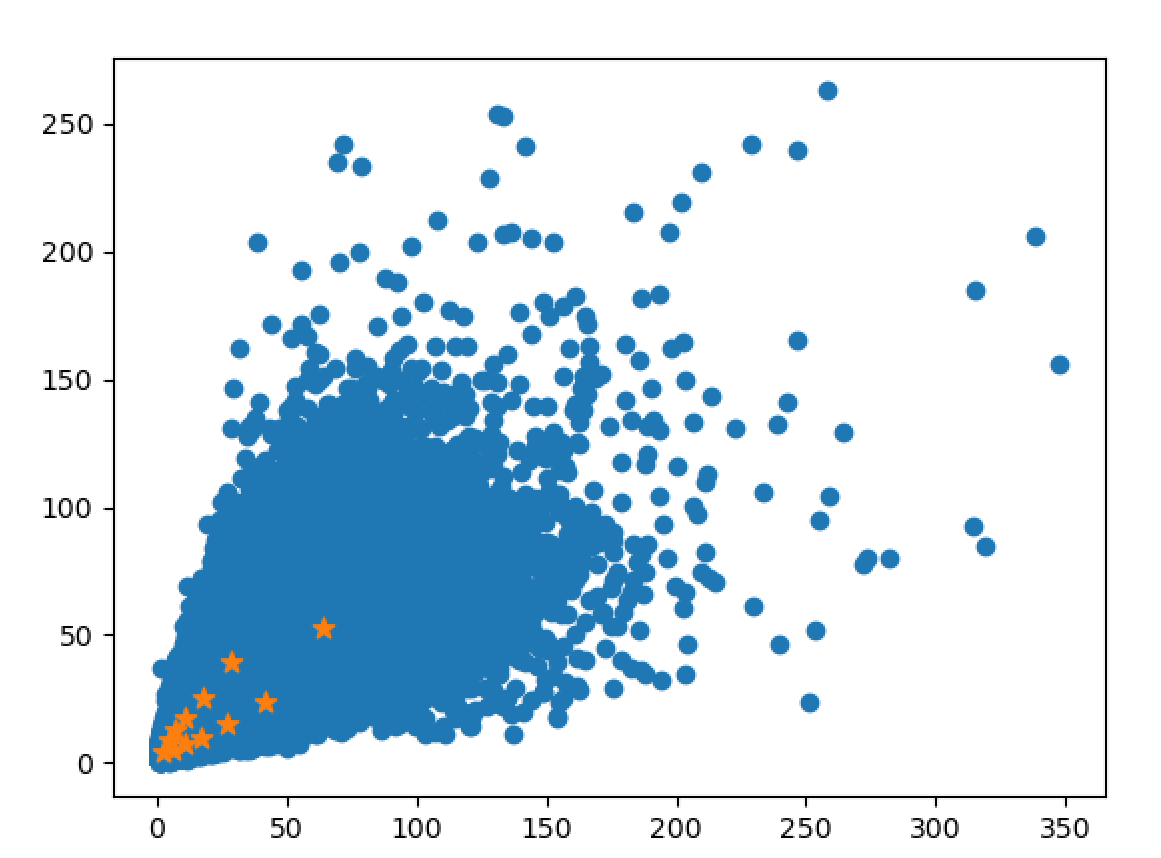
[ 9.6 16. ]

[24. 13.71428591]

[18.37160752 25.14285714]

[44.8 37.48571483]]

计算速度非常快



Kmeans,k=12

kmeans: [2, 4] [6, 5] [4, 8] [10, 7] [7, 12] [17, 10] [11, 17] [26, 15] [17, 25] [42, 23] [28, 39] [64, 53]

fitness: 0.77375, best possible recall: 0.99941

[[ 2.66666667 4.11428571]

[ 6. 5.02857143]

[ 4.57142857 8.94117647]

[10.97142857 7.31428571]

[ 7.31428571 12.80000019]

[17.37142857 10.057143 ]

[11.29411765 17.82857143]

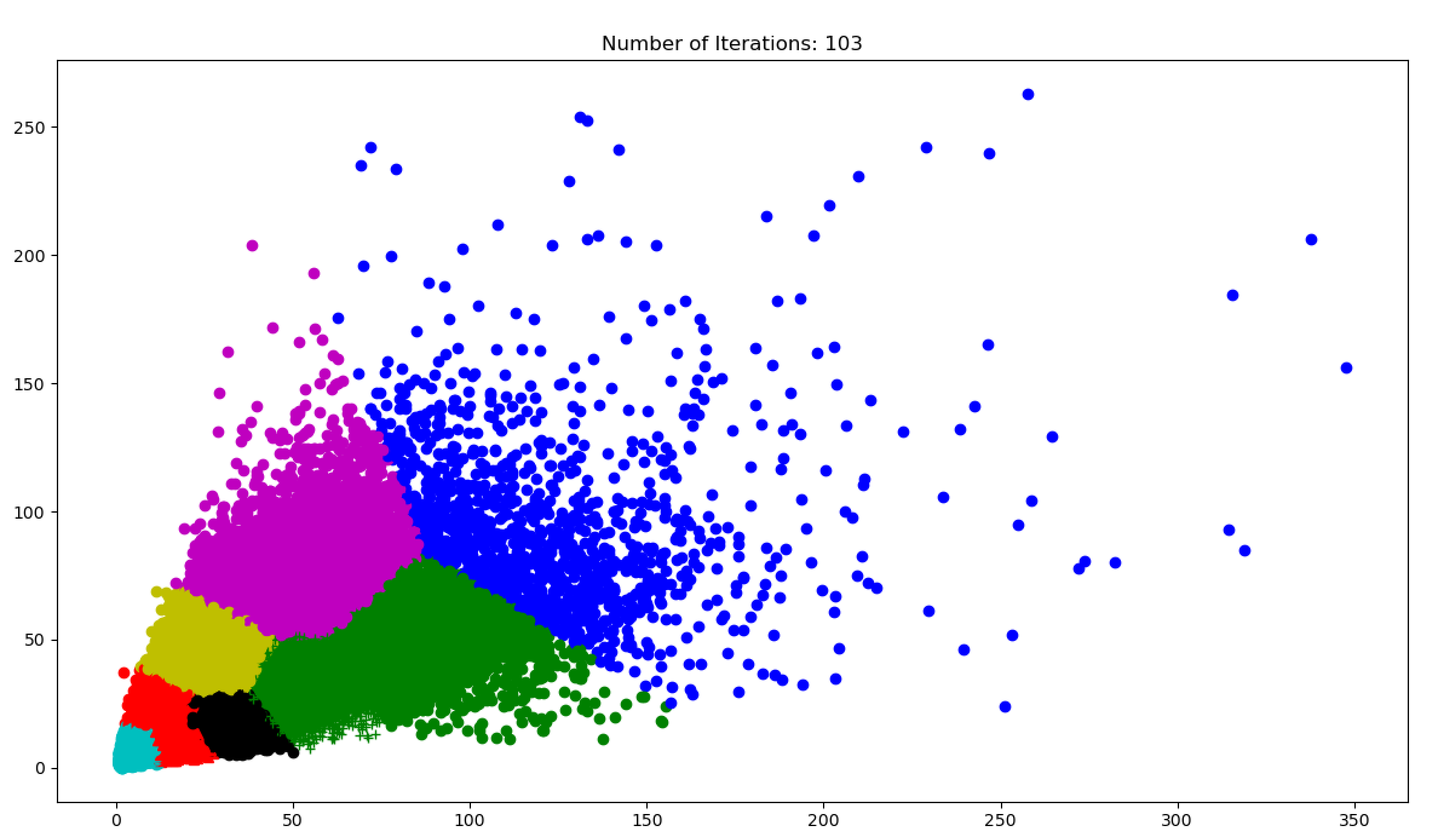
[26.97142857 15.08571429]

[17.82857143 25.14285714]

[42.05714286 23.71607442]

[28.8 39.31428571]

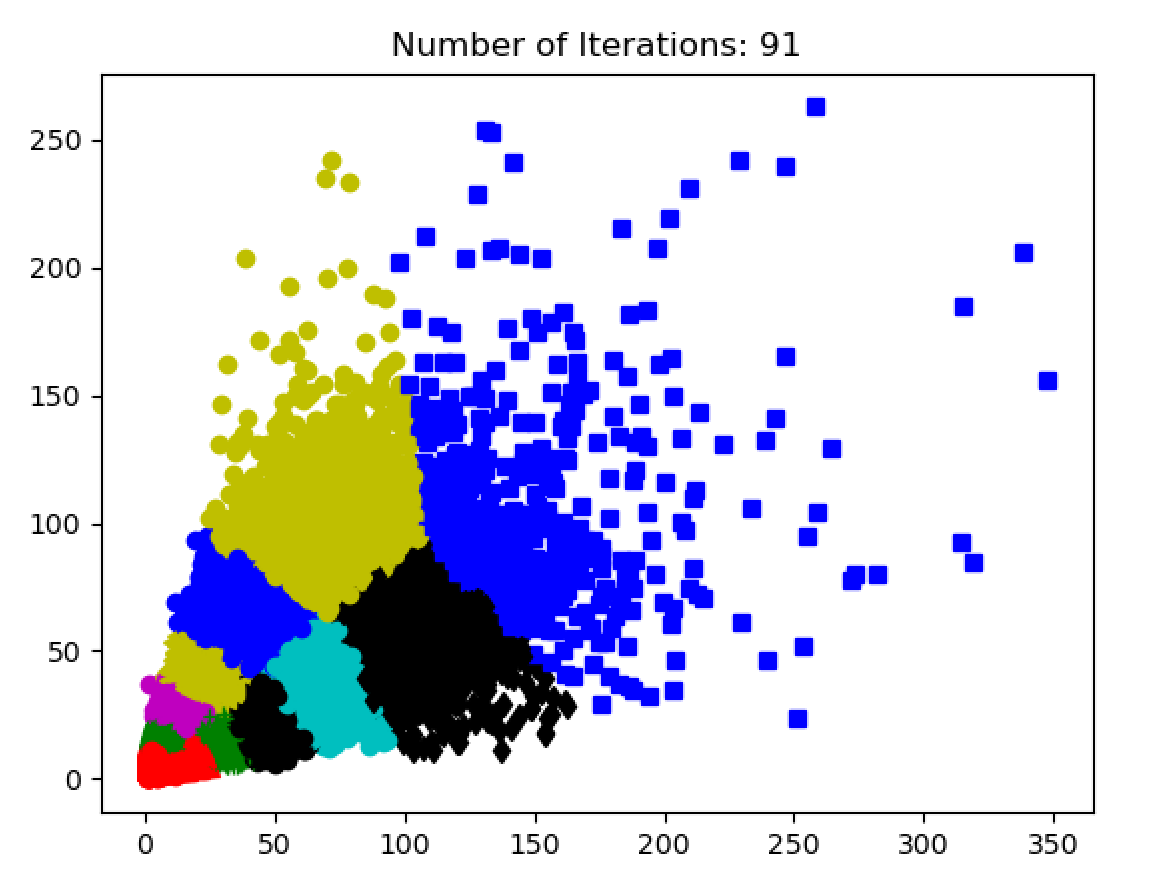
[64. 53.02857143]]



Kmeans++,k=9

kmeans: [5.747678193039241, 6.561947406522153] [9.785927086234055, 15.51328168729093] [20.586664684940956, 12.0573669932274] [19.330193328985686, 28.84815492310076] [39.637536914427706, 22.52381848766016] [34.88598096174333, 49.51245439074425] [69.60210621973907, 39.429339959780506] [60.5335680205991, 87.15022137285801] [124.13237134052589, 75.43681304042669]

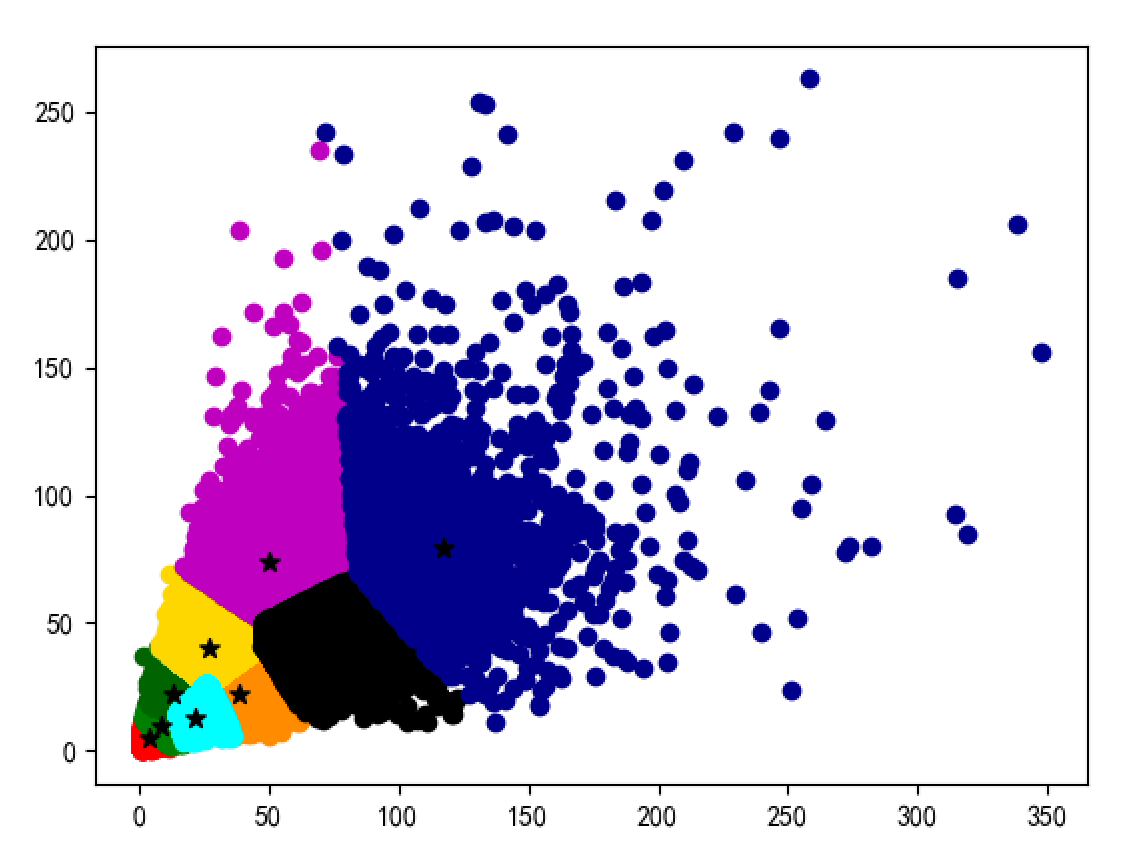
fitness: 0.68557, best possible recall: 0.98459

kmeans

kmeans++ ,k=12

kmeans: [4.9289808045778996, 5.874979838216507] [7.833113044369644, 13.319885597837734] [15.721028742136953, 9.771295561771039] [14.20321924917401, 22.866218423275875] [27.59240824729452, 16.174624379186596] [24.75652407084264, 36.35554020131952] [43.23672128052264, 24.6058493878137] [39.39262841497565, 57.34444676657576] [64.38857428998219, 37.14811444794271] [97.68134230631351, 52.87702134845113] [65.10379332466701, 93.36534035046476] [148.0633808940638, 101.25742161029011]

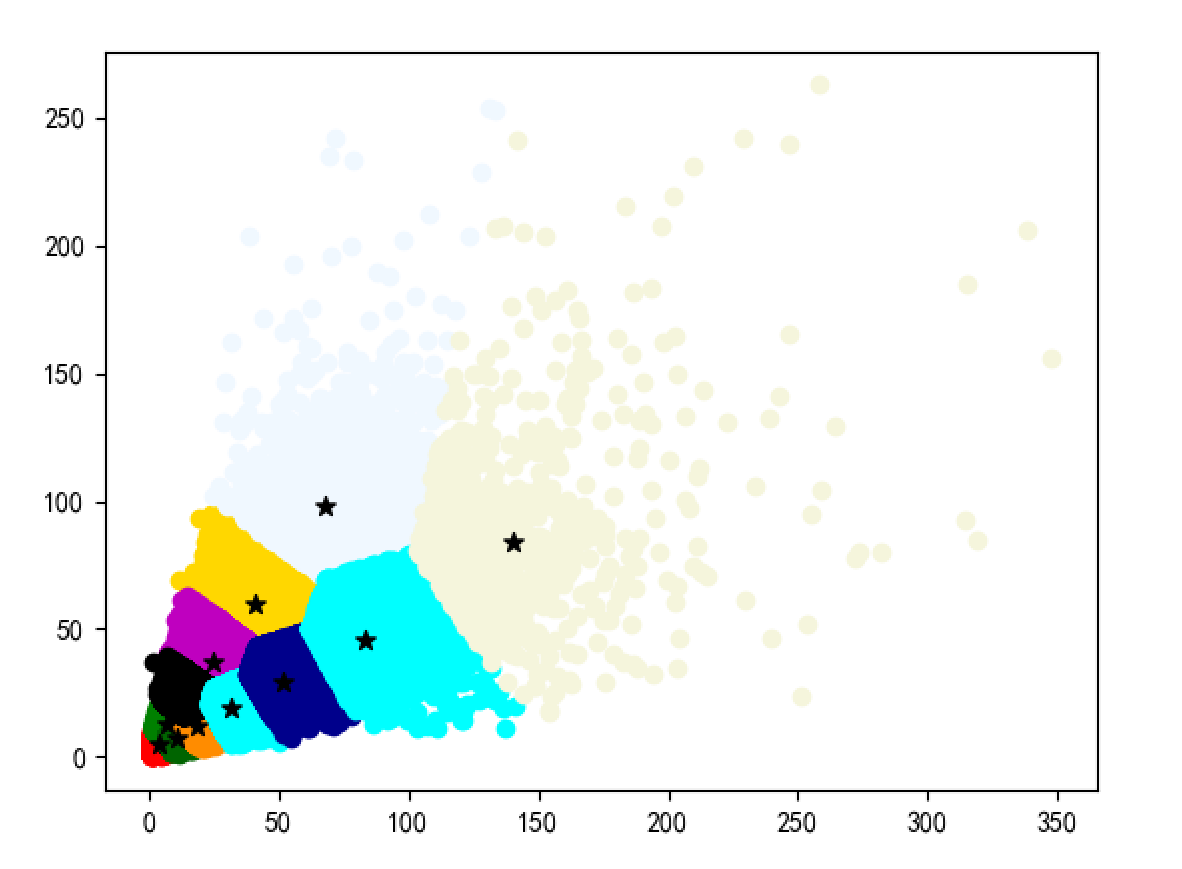
fitness: 0.71848, best possible recall: 0.98955



canopy = Canopy(points, t1=160, t2=90)

kmeans: [4, 5] [9, 10] [13, 22] [22, 13] [27, 40] [39, 22] [50, 74] [67, 38] [117, 79]

fitness: 0.70277, best possible recall: 0.99843



Canopy

canopy = Canopy(points, t1=140, t2=65)

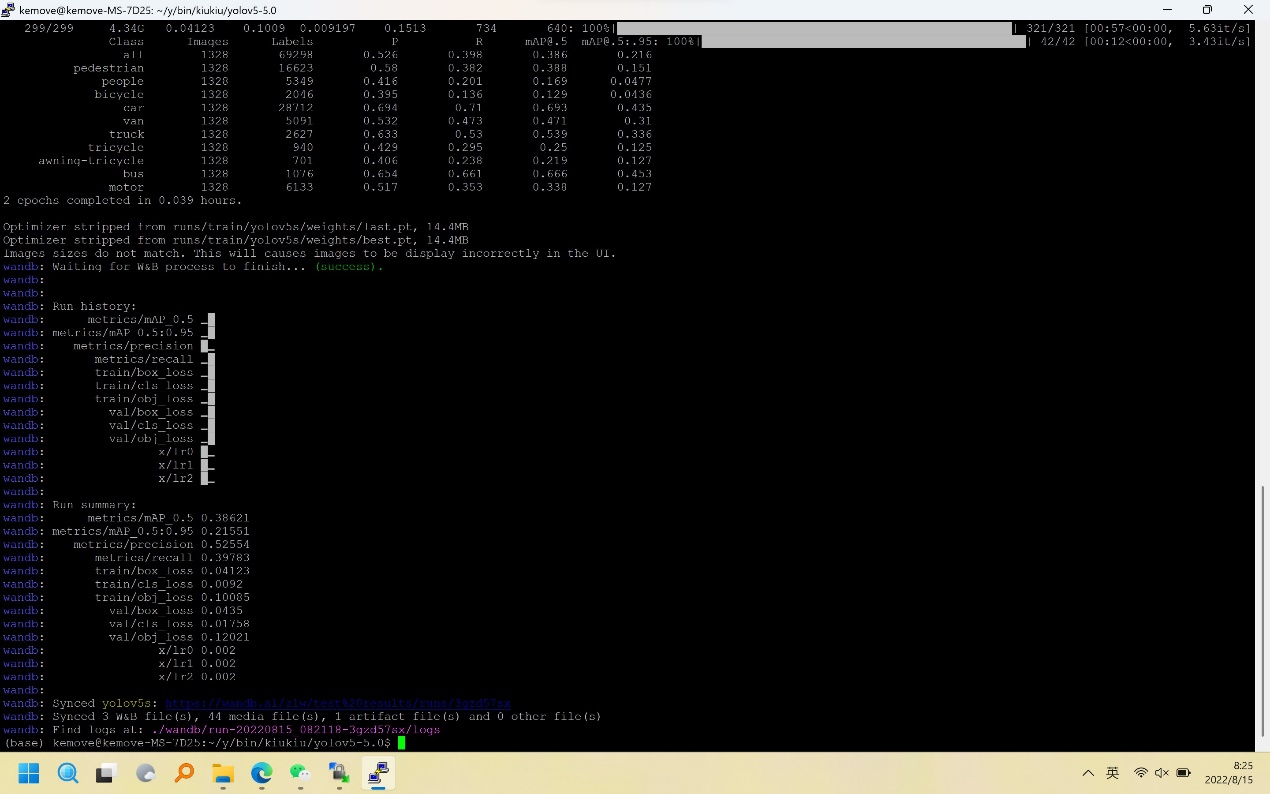
kmeans: [4, 5] [7, 13] [11, 7] [19, 12] [41, 60] [25, 37] [14, 23] [32, 19] [52, 29] [68, 98] [83, 46] [140, 84]

fitness: 0.73556, best possible recall: 0.99843

# 实验结果

1. anchor为coco128默认的anchor值

python train.py --data data/visdrone.yaml --cfg models/yolov5s.yaml --weights weights/yolov5s.pt --batch-size 16 --epochs 300 --workers 4 --name yolov5s



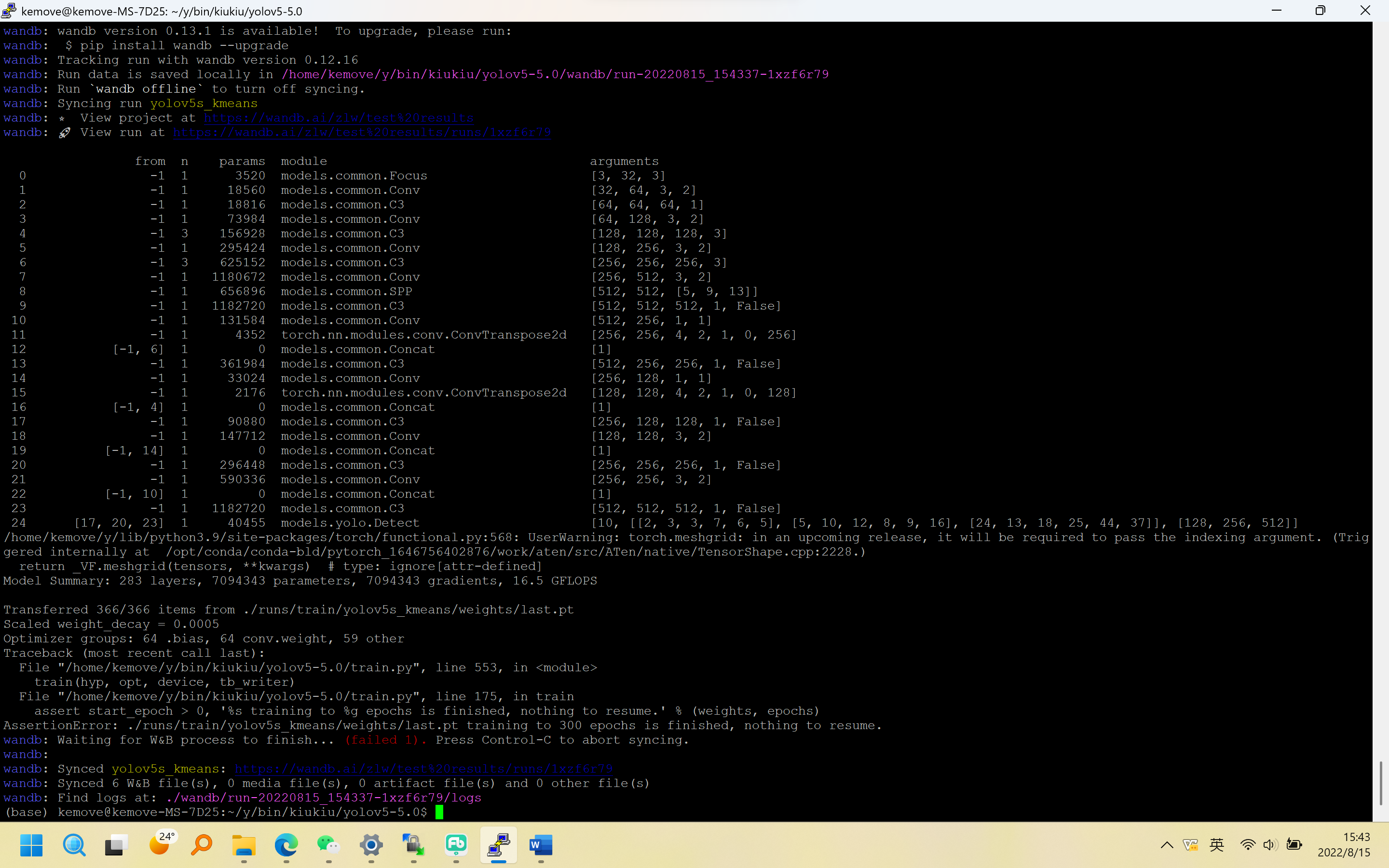
1. anchors:

- [2,3 ,3,7, 6,5] # P3/8

- [5,10, 12,8, 9,16] # P4/16

- [24,13, 18,25, 44,37] # P5/32

python train.py --data data/visdrone.yaml --cfg models/yolov5s.yaml --weights weights/yolov5s.pt --batch-size 16 --epochs 300 --workers 4 --name yolov5s\_kmeans

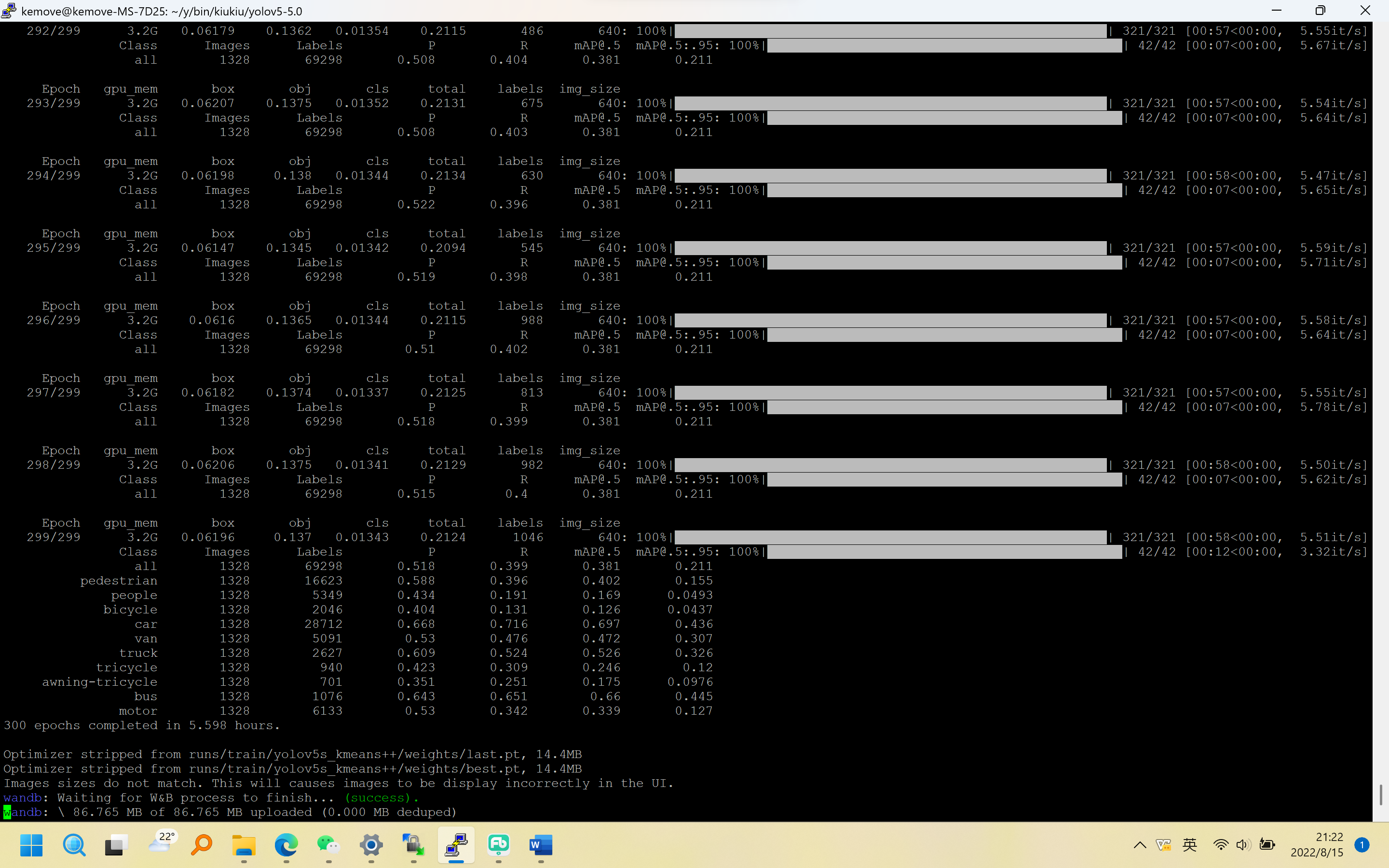


1. - [5.747678193039241, 6.561947406522153,9.785927086234055, 15.51328168729093,20.586664684940956, 12.0573669932274] # P3/8

- [19.330193328985686, 28.84815492310076,39.637536914427706, 22.52381848766016,34.88598096174333, 49.51245439074425] # P4/16

- [69.60210621973907, 39.429339959780506,60.5335680205991, 87.15022137285801,124.13237134052589, 75.43681304042669] # P5/32

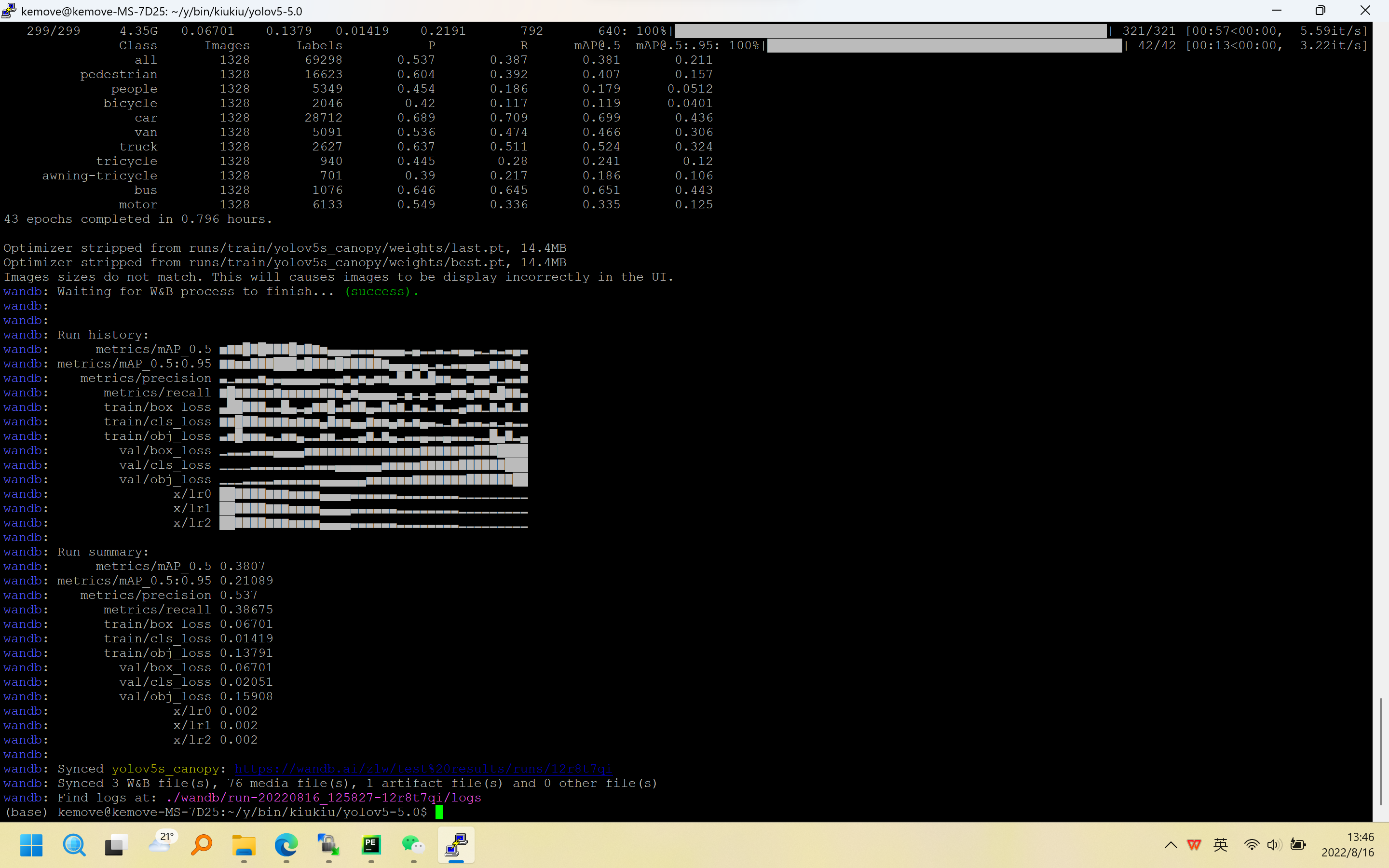
python train.py --data data/visdrone.yaml --cfg models/yolov5s.yaml --weights weights/yolov5s.pt --batch-size 16 --epochs 300 --workers 4 --name yolov5s\_kmeans++



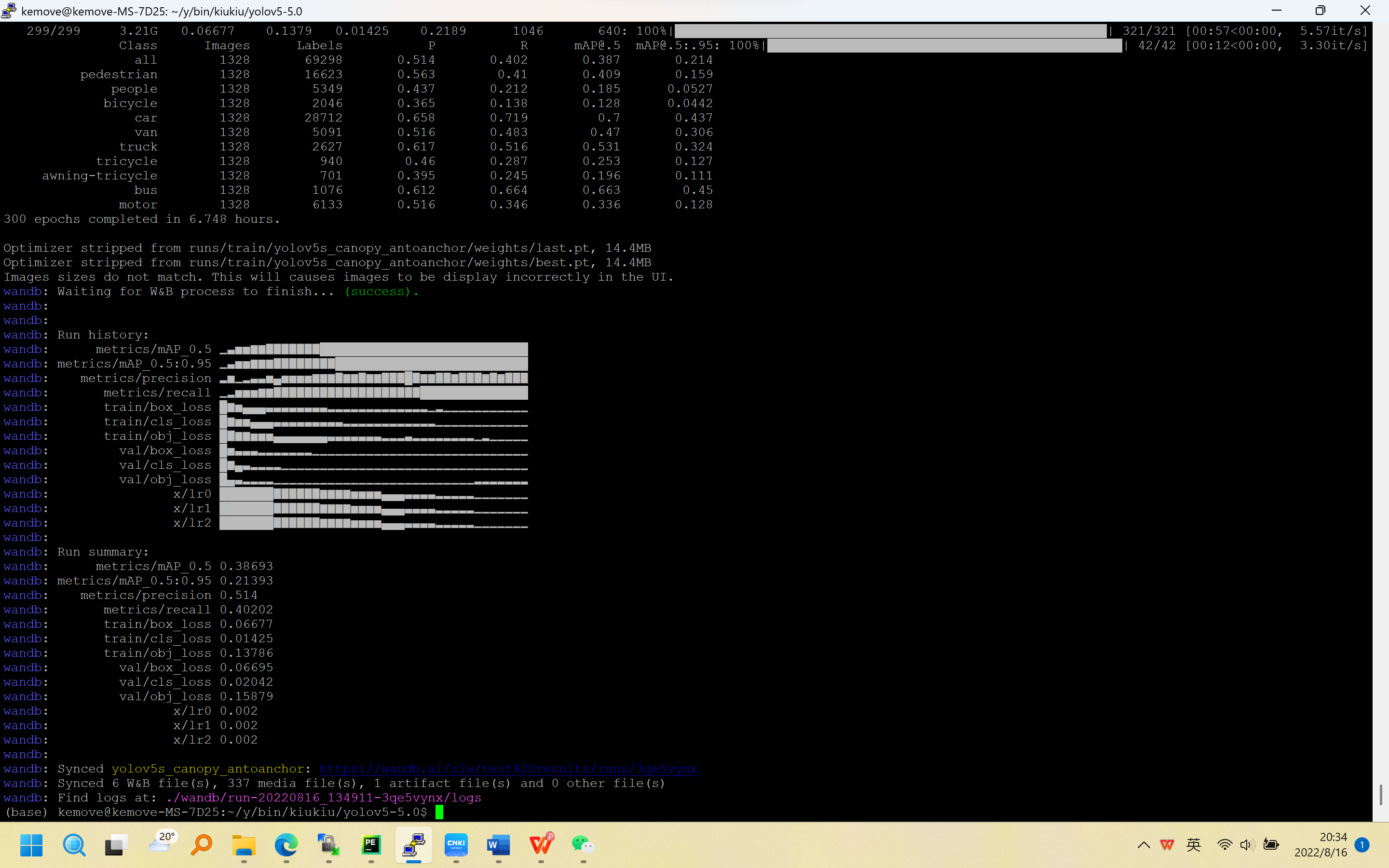
1. canopy = Canopy(points, t1=160, t2=90)

kmeans: [4, 5] [9, 10] [13, 22] [22, 13] [27, 40] [39, 22] [50, 74] [67, 38] [117, 79]

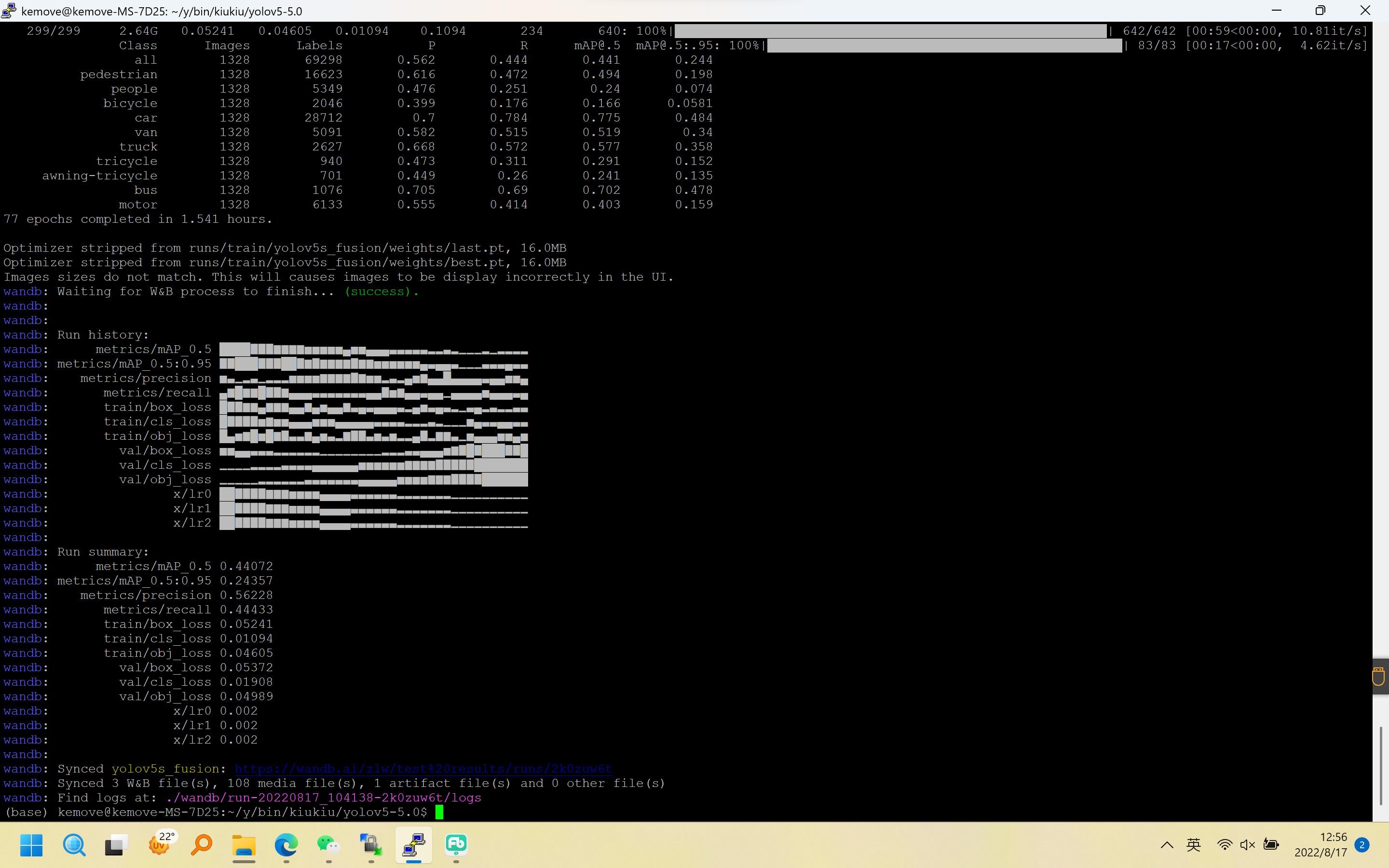
python train.py --data data/visdrone.yaml --cfg models/yolov5s.yaml --weights weights/yolov5s.pt --batch-size 16 --epochs 300 --workers 4 --name yolov5s\_canopy



1. python train.py --data data/visdrone.yaml --cfg models/yolov5s.yaml --weights weights/yolov5s.pt --batch-size 16 --epochs 300 --workers 4 --name yolov5s\_canopy\_antoanchor



1. python train.py --data data/visdrone.yaml --cfg models/yolov5s\_fusion.yaml --weights weights/yolov5s.pt --batch-size 8 --epochs 300 --workers 4 --name yolov5s\_fusion



# 一些错误

如果训练中断，将resume中default=True,重新运行usage中的代码即可恢复，记得使用完后将default还原为False。

图片切割代码链接

[(2条消息) 解决not well-formed (invalid token)BUG，xml标签转到txt标签，txt标签转到xml标签，滑动窗口切割图像并且同步标签\_言初-xys的博客-CSDN博客](https://blog.csdn.net/weixin_42433234/article/details/123555977?spm=1001.2101.3001.6661.1&utm_medium=distribute.pc_relevant_t0.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-1-123555977-blog-86379723.pc_relevant_multi_platform_featuressortv2removedup&depth_1-utm_source=distribute.pc_relevant_t0.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-1-123555977-blog-86379723.pc_relevant_multi_platform_featuressortv2removedup&utm_relevant_index=1)

\_\_init\_\_() missing 1 required positional argument: 'c2'

[(3条消息) Python成功解决TypeError: \_\_init\_\_() missing 1 required positional argument: ‘comment‘\_肥鼠路易的博客-CSDN博客](https://blog.csdn.net/weixin_44991673/article/details/110099428)

在使用ghostnet进行轻量化时需要将c3模块进行修改

为啥用了ghostnet后的GLOPS会变得更大呢?

在colab运行代码时，出现images中没有图片，是因为挂载云盘超时，可以多次挂载将数据全部读出来以后再进行训练，很不智能啊！！！

Colab读取云盘信息特别慢，因此在运行代码前将所有代码放到colab本地

import shutil

# 把olddir拷贝一份newdir

shutil.copytree('drive/MyDrive/qxf/yolov5-5.0', 'simple\_data')