# 消融实验：分支数以及卡数的影响

# RepSeg\_D2\_6G

# 0.实验结论

1）基于RepSeg6G进行分支重要性试验，基础3分支（3x3/1x1/shotcut）。 额外加入1x3/3x1 2分支后，涨点0.3%，额外加入conv1x1\_conv3x3/conv1x1\_avg3x3 2分支后，涨点0.23%。额外加入1x3/3x1/conv1x1\_conv3x3/conv1x1\_avg3x3 4分支后涨点0.71% 。加入更多分支可以微弱涨点，但是不明显。猜测原因：虽然使用更多分支，但是没有引入非线性，因此涨点不明显。

2）之前的实验可复现，基于RepVGG6G进行了2个结构的复现实验，实验1复现前后75.93->75.78，实验2复现前后76.77->76.61。

3）可以基于RepSeg的代码激活3分支进行类RepVGG的实验：a. RepVGG-7-4\_ERFHead2-2， 77.04， RepSeg(RepVGG3分支)-7-4\_ERFHead2-2，76.86，性能相当。b. RepVGG-7-4-PSPHead\_160K，74.68，RepSeg(RepVGG3分支)-7-4\_PSPHead\_160K，74.47 ，性能也相当。c. RepVGG-7-4-PSPHead\_320K，74.88, RepSeg(RepVGG3分支)-7-4\_PSPHead\_320K，75.70 ，性能也相当(虽有涨点，认为是波动影响)。

4）基于RepSeg进行多卡实验，batchsize和学习率保证不变的条件下，使用pytorch1.9.0及seed=0，8卡相对2卡性能下降了1.15%，猜测可能的原因是，卡数(线程数)及num\_worker影响了样本的采样顺序，导致收敛的精度不一样。暂容许这种存在。

5）在batchsize和学习率保证不变的条件下，随机初始化，增加1倍的迭代次数（160K->320K），性能提升1个点。且迭代次数降低1倍（160K->80K），性能下降很严重，4%。证明了在随机初始化时增加迭代次数的重要性。

# 1. RepSeg: Baseline

**Torch1.6.0：**

Python: 3.7.10

PyTorch: 1.6.0

TorchVision: 0.7.0

OpenCV: 4.5.2

**MMCV: 1.3.4**

MMCV Compiler: GCC 8.3

MMCV CUDA Compiler: 10.2

**MMSegmentation: 0.13.0+66b0525**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EXP | crop\_size/train\_size=1024\*512, test\_size：1024x2048  seed=0, deterministic=False, **torch1.6.0**+mmseg0.13.0  cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_noaux | | | | |
| lr+iter | batch\_size | miou | 参数 | exp\_name |
| **1** | **syncbn lr=0.01 320k** | **b16\_g2\_s8\_w4** | **76.54** | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | **cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_noaux.py** |
| 2 | syncbn lr=0.01 320k | b16\_g8\_s2\_w2 | 74.70 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_noaux\_b8x2.py |
| 由“repseg复现及试验调参”exp1~5，基于pytorch1.9.0及seed=0，卡数影响了1.15%，还是有影响，  但是比“pytorch1.6.0+不约束seed”影响小，猜测原因有一下2点：  1）随机初始化方式不一样，  2）卡数多少及num\_worker的数量影响了样本的采样顺序，导致收敛的精度不一样 | | | | | |
| 3 | bn lr=0.01 320k | b16\_g8\_s2\_w2 | 70.80 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_bn\_noaux\_b8x2.py |
| 再次验证syncbn相对bn在卡数相对较少时，还是非常有用的 | | | | | |

# 2.RepVGG: Baseline

Python: 3.7.10

PyTorch: 1.6.0

TorchVision: 0.7.0

OpenCV: 4.5.2

MMCV: 1.3.4

MMCV Compiler: GCC 8.3

MMCV CUDA Compiler: 10.2

MMSegmentation: 0.13.0+66b0525

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EXP | crop\_size/train\_size=1024\*512, test\_size：1024x2048  seed=0, deterministic=False, **torch1.6.0**+mmseg0.13.0  **RepVGG\_6G**\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch | | | | |  |
| lr+iter | batch\_size | miou | 参数 | exp ft. | exp\_name. |
| **1** | **syncbn lr=0.01 320k** | **b16\_g2\_s8\_w4** | **75.93** | RepVGG\_bmseg: 5-12  RepVGG\_PSPHead  no aux  with ppm | **RepVGG\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch** | cityscapes\_RepVGG\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_aux.py |
| 实验1可复现，复现点数，75.78，见“repvgg复现及试验调参”EXP0 | | | | | |  |
| 2 | syncbn lr=0.01 320k | b16\_g2\_s8\_w4 | 75.62 | RepVGG\_bmseg: 5-12  RepVGG\_ERFHead: 2-2  aux(drop0)  no ppm | **RepVGG\_6G**\_5-12-16-64-128\_noppm\_erfhead-2-2\_128\_64\_16\_512x1024\_320k\_scratch\_syncbn\_aux  3个反卷积代替interpolate，backbone的4个repblock移到decoder中，syncbn + aux | cityscapes\_RepVGG\_6G\_5-12-16-64-128\_noppm\_erfhead-2-2\_128\_64\_16\_512x1024\_320k\_scratch\_syncbn\_aux.py |
| 3 | syncbn lr=0.01 320k | b16\_g2\_s8\_w4 | 76.77 | RepVGG\_bmseg: 5-12  RepVGG\_PSPHead  aux(drop0)  with ppm | RepVGG\_6G\_5-12-16-64-128\_ppm128-2346\_erfhead-2-2\_128\_64\_16\_512x1024\_320k\_scratch\_syncbn\_aux  3个反卷积代替interpolate，backbone的4个repblock移到decoder中，syncbn +PPM128-2346+ aux | cityscapes\_RepVGG\_6G\_5-12-16-64-128\_ppm128-2346\_erfhead-2-2\_128\_64\_16\_512x1024\_320k\_scratch\_syncbn\_aux.py |
| 实验3可复现，复现点数，76.61，见“repvgg复现及试验调参”EXP1，  另外， EXP2和EXP3说明在大分辨率下ppm很重要 | | | | | |  |

# 3.repseg复现及试验调参

**Torch1.9.0：**

Python: 3.7.10

**PyTorch: 1.9.0+cu102**

**TorchVision: 0.10.0+cu102**

OpenCV: 4.5.2

**MMCV: 1.3.8**

MMCV Compiler: GCC 8.3

MMCV CUDA Compiler: 10.2

**MMSegmentation: 0.13.0+66b0525**

**1）卡数+迭代次数+学习率的影响**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EXP | crop\_size/train\_size=1024\*512, test\_size：1024x2048  seed=0, deterministic=False, **torch1.9.0**+mmseg0.13.0  cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_noaux | | | | | |
| num | lr+iter | batch\_size | miou | 参数 | time | exp\_name |
| 1 | syncbn lr=0.01 320k | b16\_g2\_s8\_w4 | 75.91 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | 3.25d | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g2\_s8\_w4.py |
| 2 | syncbn lr=0.01 320k | b16\_g2\_s8\_w4 | 待running  做个收尾工作 | RepSeg\_bmseg: 5-12  RepSeg\_ERFHead:2-2  noaux  ppm |  |  |
| 3 | syncbn lr=0.01 320k | b16\_g2\_s8\_w8 | 75.83 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | 3.5d | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g2\_s8\_w8.py |
| 4 | syncbn lr=0.01 320k | b16\_g8\_s2\_w2 | 75.08 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | 1.5d | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g8\_s2\_w2.py |
| 5 | syncbn lr=0.01 320k | b16\_g4\_s4\_w4 | **76.23** | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | 2d | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4.py |
| 由“repseg复现及试验调参”exp1~5，基于pytorch1.9.0及seed=0，卡数影响了1.15%，还是有影响，  猜测主要原因是：1）卡数多少及num\_worker的数量影响了样本的采样顺序，导致收敛的精度不一样 | | | | | | |
| 6 | syncbn lr=0.01 160k | b16\_g4\_s4\_w4 | 75.17 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | 1d | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_160k\_torch1.9.0 |
| EXP6相对EXP5低了近1个点，进一步证明，迭代次数增加的好处。 | | | | | | |
| 7 | syncbn lr=0.02 160k | b16\_g4\_s4\_w4 | 75.23 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | 1d | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_160k\_lr0.02.py |
| EXP7相对EXP6学习率增加，性能基本无影响，说明初始学习率影响并不太大 | | | | | | |
| 8 | syncbn lr=0.01 80k | b16\_g4\_s4\_w4 | 71.58 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  with ppm | 12h | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_80k.py |
| EXP8说明在随机初始化且迭代次数不够的情况下，性能很差 | | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EXP | crop\_size/train\_size=1024\*512, test\_size：1024x2048  seed=0, deterministic=False, **torch1.9.0**+mmseg0.13.0  cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_noaux | | | | | |
| num | lr+iter | batch\_size | miou | 参数 | time | exp\_name |
| 1 | syncbn lr=0.01 320k | b16\_g4\_s4\_w4 | **76.23** | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  ppm | 2d | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4.py |
| 2 | syncbn lr=0.01 320k | b16\_g4\_s4\_w4 | 75.44 | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  aux(drop0)  ppm | 2d | RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_aux0.1\_7-14.py\_torch1.9.0 |
| EXP2相对EXP1说明AUX的dropout基本无影响，另一方面，性能有波动，波动还比较大 | | | | | | |
| 3 | syncbn lr=0.01 320k | b16\_g4\_s4\_w4 | 75.24 | RepSeg\_bmseg: 7-14  RepSeg\_ERFHead:2-2  aux0.1  ppm | 2d | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_aux\_5-12-erf2-2\_torch1.9.0 |
| 4 | syncbn lr=0.01 320k | b16\_g4\_s4\_w4 | 76.54 | RepSeg\_bmseg: 5-12  RepSeg\_ERFHead:2-2  aux0.1  ppm |  | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_aux0.1\_erf2-2-real5-12\_torch1.9.0 |
| EXP3点数完全不应该只有75.24， 波动影响，  EXP4相对EXP1性能有提升，说明了逐步上采样有好处，  相对“repvgg复现及试验调参”的exp5有微弱涨点，进一步说明多分支带来的收益不明显 | | | | | | |

# 4.repseg不同分支影响

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EXP | crop\_size/train\_size=1024\*512, test\_size：1024x2048  seed=0, deterministic=False, **torch1.9.0**+mmseg0.13.0  cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_noaux  **lr+iter：syncbn lr=0.01 160k， batch\_size：b16\_g4\_s4\_w4** | | | | |
| num | 分支数 | miou | time |  | exp\_name |
| 1 | conv3x3 conv1x1 shotcut  conv1x1\_conv3x3  conv1x1\_avg3x3  conv1x3 conv3x1  更多分支有用？ | 75.17 | 1d | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  ppm | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_160k\_torch1.9.0 |
| 2 | conv3x3 conv1x1 shotcut | 74.46 | 1d | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  ppm | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_160k\_repvgg3\_torch1.9.0 |
| 3 | conv3x3 conv1x1 shotcut  +aux(drop=0)， aux基本无用 | 74.47 |  | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  aux(drop0)  ppm | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_160k\_repvgg3\_aux\_torch1.9.0 |
| 4 | conv3x3 conv1x1 shotcut  conv1x3 conv3x1 | 74.76 | 1d | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  ppm | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_160k\_repvgg3\_1x3\_3x1\_torch1.9.0 |
| 5 | conv3x3 conv1x1 shotcut  conv1x1\_conv3x3  conv1x1\_avg3x3 | 74.69 | 1d | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  ppm | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_160k\_dbb5\_torch1.9.0 |
| EXP1~5，基于RepSeg6G进行分支重要性试验，基础3分支（3x3/1x1/shotcut）。 额外加入1x3/3x1 2分支后，涨点0.3%，额外加入conv1x1\_conv3x3/conv1x1\_avg3x3 2分支后，涨点0.23%。额外加入1x3/3x1/conv1x1\_conv3x3/conv1x1\_avg3x3 4分支后涨点0.71% 。加入更多分支可以微弱涨点，但是不明显。理由：虽然使用更多分支，但是没有引入非线性。 | | | | | |
| 6 | 补充实验：conv3x3 conv1x1 shotcut, 320k, 看后面是在repseg分支还是repvgg分支上继续做实验 | 75.70 |  | RepSeg\_bmseg: 7-14  RepSeg\_PSPHead  noaux  ppm | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_320k\_repvgg3\_torch1.9.0 |
| EXP6相对EXP2，增加1倍的迭代次数后涨1个点，进一步证明增加迭代次数的有效性。  EXP6相对“repvgg复现及试验调参”的EXP3的74.88涨点点（波动影响），进一步证明增加迭代次数的有效性。 | | | | | |
| 7 | 补充实验：conv3x3 conv1x1 shotcut, 320k, 看后面是在 | 76.86 |  | RepSeg\_bmseg: 7-14  RepSeg\_ERFHead: 2-2  with aux(drop0.1)  ppm | log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_back0\_aux0.1\_erf2-2-b7-14\_repvgg3\_torch1.9.0 |
| EXP7相对“repvgg复现及试验调参”的EXP4(77.04)，差不多，证明repseg分支继续做类repvgg的实验 | | | | | |

# 5.repvgg复现及试验调参

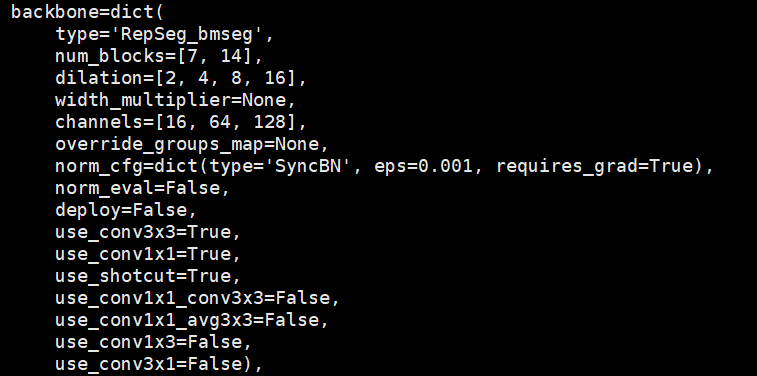
**RepVGG vs RepSeg（试验复现），RepSeg不应该比RepVGG差呀**

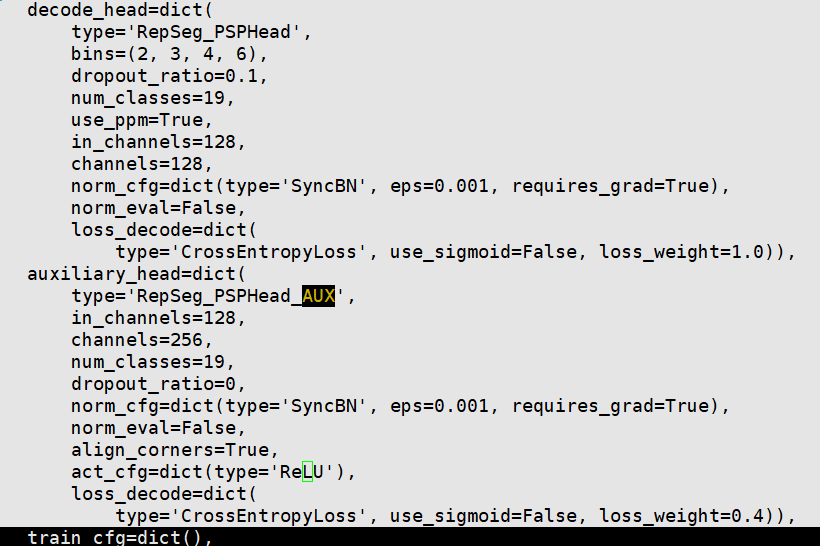
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EXP | crop\_size/train\_size=1024\*512, test\_size：1024x2048  seed=0, deterministic=False, **torch1.9.0**+mmseg0.13.0  RepSeg基础配置：  cityscapes\_RepSeg\_D2-2-4-8-16\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_noaux  lr+iter：syncbn lr=0.01 160k， batch\_size：b16\_g4\_s4\_w4  RepVGG基础配置：  cityscapes\_RepVGG\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_aux.py | | | | |
| num | 分支数 | miou | 参数 | time | exp\_name |
| 0 | lr+iter: syncbn lr=0.01 320k， batch\_size：b16\_g2\_s8\_w4 | 75.78  (75.93) | RepVGG\_bmseg: 5-12  RepVGG\_PSPHead  aux(drop0)  with ppm | 2d | log\_cityscapes\_RepVGG\_6G\_7-14-16-64-128-ppm128\_512x1024\_320k\_scratch\_syncbn\_aux\_b16\_g2\_s8\_w4\_torch1.9.0 |
| 复现了RepVGG baseline1, 75.93 | | | | | |
| 1 | lr+iter: syncbn lr=0.01 320k， batch\_size：b16\_g2\_s8\_w4 | 76.61  (76.77) | RepVGG\_bmseg: 5-12  RepVGG\_ERFHead: 2-2  aux(drop0)  no ppm | 2d | log\_cityscapes\_RepVGG\_6G\_5-12-16-64-128\_ppm128-2346\_erfhead-2-2\_128\_64\_16\_512x1024\_320k\_scratch\_syncbn\_aux\_b16\_g2\_s8\_w4\_torch1.9.0  3个反卷积代替interpolate，backbone的4个repblock移到decoder中，syncbn +PPM128-2346+ aux |
| 复现了RepVGG baseline2, 76.77 | | | | | |
| 2 | lr+iter: syncbn lr=0.01 320k， batch\_size：b16\_g4\_s4\_w4 | 75.29 | RepVGG\_bmseg: 5-12  RepVGG\_PSPHead  aux(drop0)  with ppm | 1d | log\_RepVGG\_6G\_7-14-16-64-128-ppm128\_320k\_lr0.01\_b16\_g4\_s4\_w4\_aux\_torch1.9.0 |
| 3 | lr+iter: syncbn lr=0.01 320k， batch\_size：b16\_g4\_s4\_w4 | 76.38 | RepVGG\_bmseg: 5-12  RepVGG\_ERFHead: 2-2  aux(drop0.1)  with ppm |  | RepVGG\_6G\_7-14-16-64-128-ppm128\_320k\_lr0.01\_b16\_g4\_s4\_w4\_aux0.1\_real6G\_erf2-2\_torch1.9.0 |
| EXP2，再次说明卡数的多少有一定影响，  EXP3相对EXP2，说明erfhead确实有涨点 | | | | | |
| 4 | lr+iter: syncbn lr=0.01 320k， batch\_size：b16\_g4\_s4\_w4 | 74.88 | RepVGG\_bmseg: 7-14  RepVGG\_PSPHead  aux(drop0.1)  with ppm | 1d | log\_RepVGG\_6G\_7-14-16-64-128-ppm128\_320k\_lr0.01\_b16\_g4\_s4\_w4\_aux0.1\_real7-14\_torch1.9.0 |
| 5 | lr+iter: syncbn lr=0.01 320k， batch\_size：b16\_g4\_s4\_w4 | 77.04 | RepVGG\_bmseg: 7-14  RepVGG\_ERFHead: 2-2  aux(drop0.1)  with ppm |  | RepVGG\_6G\_7-14-16-64-128-ppm128\_320k\_lr0.01\_b16\_g4\_s4\_w4\_aux0.1\_falk6G\_erf2-2\_torch1.9.0 |
| EXP4/5，网络波动影响，且erfhead确实有涨点 | | | | | |
| 6 | lr+iter: syncbn lr=0.01 160k， batch\_size：b16\_g4\_s4\_w4 | 74.58 | RepVGG\_bmseg: 5-12  RepVGG\_PSPHead  aux(drop0)  with ppm | 12h | log\_RepVGG\_6G\_7-14-16-64-128-ppm128\_160k\_lr0.01\_b16\_g4\_s4\_w4\_aux\_torch1.9.0 |
| 7 | lr+iter: syncbn lr=0.01 160k， batch\_size：b16\_g4\_s4\_w4 | 74.68  多加layer收益甚微呀 | RepVGG\_bmseg: 7-14  RepVGG\_PSPHead  aux(drop0)  with ppm |  | log\_RepVGG\_6G\_7-14-16-64-128-ppm128\_160k\_lr0.01\_b16\_g4\_s4\_w4\_aux0\_7-14 |
| 8 | lr+iter: syncbn lr=0.01 160k， batch\_size：b16\_g4\_s4\_w4 | 73.73 | RepVGG\_bmseg: 5-12  RepVGG\_PSPHead  no aux  with ppm | 12h | log\_RepVGG\_6G\_7-14-16-64-128-ppm128\_160k\_lr0.01\_b16\_g4\_s4\_w4\_noaux\_torch1.9.0 |
| 9 | lr+iter: syncbn lr=0.01 160k， batch\_size：b16\_g4\_s4\_w4 | 74.17 | RepVGG\_bmseg: 7-14  RepVGG\_PSPHead  no aux  with ppm |  | log\_RepVGG\_6G\_7-14-16-64-128-ppm128\_160k\_lr0.01\_b16\_g4\_s4\_w4\_noaux\_7-14\_torch1.9.0 |
| EXP6~9，证明多加4个block影响很小  EXP8相对EXP6掉点较多，不能说明什么，认为是网络波动的影响。 | | | | | |

# 6.CMD

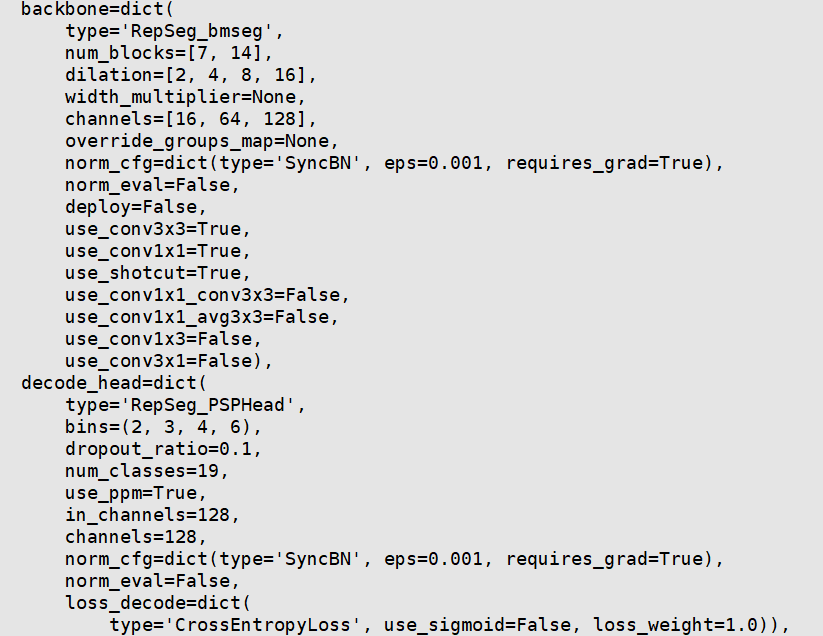
RepSeg(VGG): **AUX (mIoU未知)**

backbone=dict( type='RepSeg\_bmseg', num\_blocks=[7, 14],)

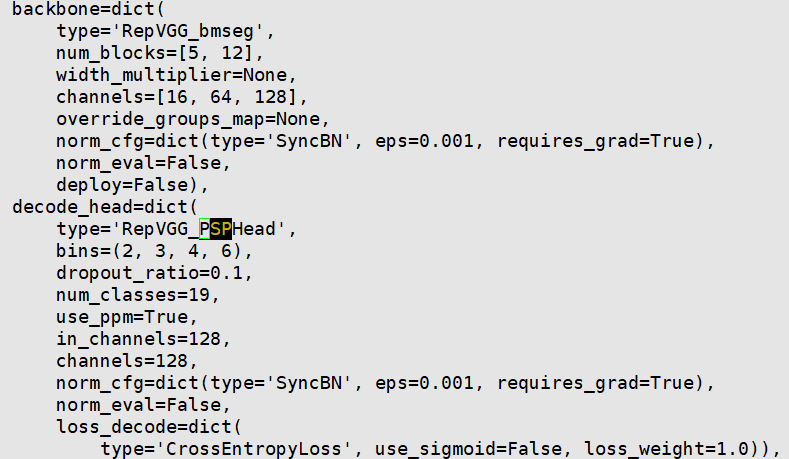




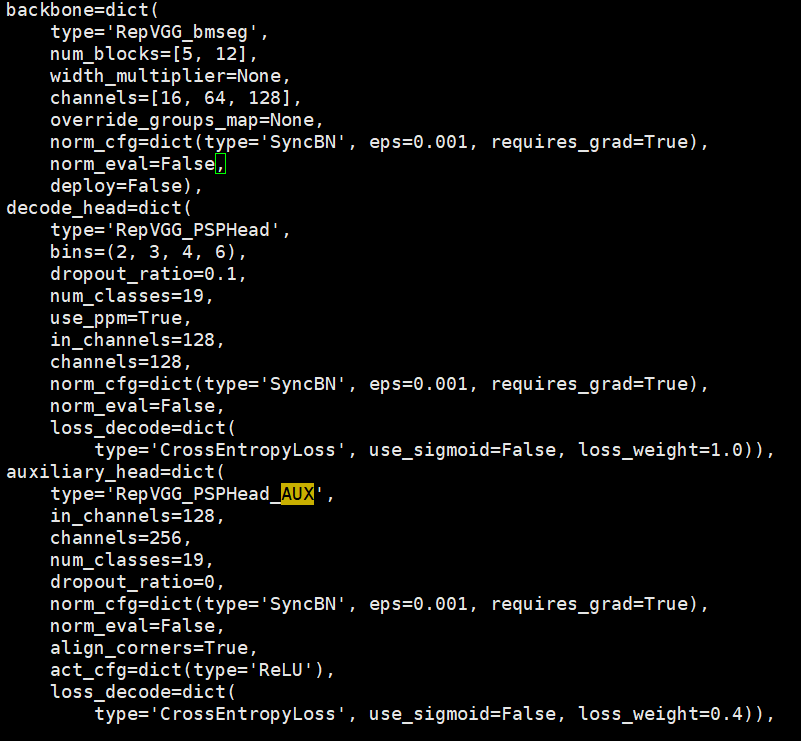
RepSeg(VGG): **noAUX(74.76)**



RepVGG (**73.73) NoAUX**



RepVGG (**74.58) AUX**



命令行：

conda activate ll-torch19

cp -rf /home/llei/project/new\_semseg/mmsegmentation\_8155\_double/configs/repseg\_gpus /home/llei/project/city\_semseg/mmsegmentation/configs/

RepVGG\_6G\_7-14-erf2-2, repvgg增加运算量：

CUDA\_VISIBLE\_DEVICES=0,1,2,3 PORT=6077 ./tools/dist\_train.sh ./configs/repvgg\_gpus/RepVGG\_6G\_7-14-16-64-128-ppm128\_320k\_lr0.01\_b16\_g4\_s4\_w4\_aux0.1\_falk6G\_erf2-2.py 4 --seed=0 >log\_RepVGG\_6G\_7-14-16-64-128-ppm128\_320k\_lr0.01\_b16\_g4\_s4\_w4\_aux0.1\_falk6G\_erf2-2\_torch1.9.0 2>&1 &

CUDA\_VISIBLE\_DEVICES=4,5,6,7 PORT=8889 ./tools/dist\_train.sh ./configs/repseg\_gpus/RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_320k\_repvgg3.py 4 --seed=0 >log\_RepSeg\_D2\_6G\_320k\_syncbn\_b16\_g4\_s4\_w4\_320k\_repvgg3\_torch1.9.0 2>&1 &