UNet多类分割试验汇总

储黄瑞

2020.6.28

使用的数据集是multi-person

方案一: UNet

Resize=（256,256）

Batch\_size=32,

data\_augmentation=False，

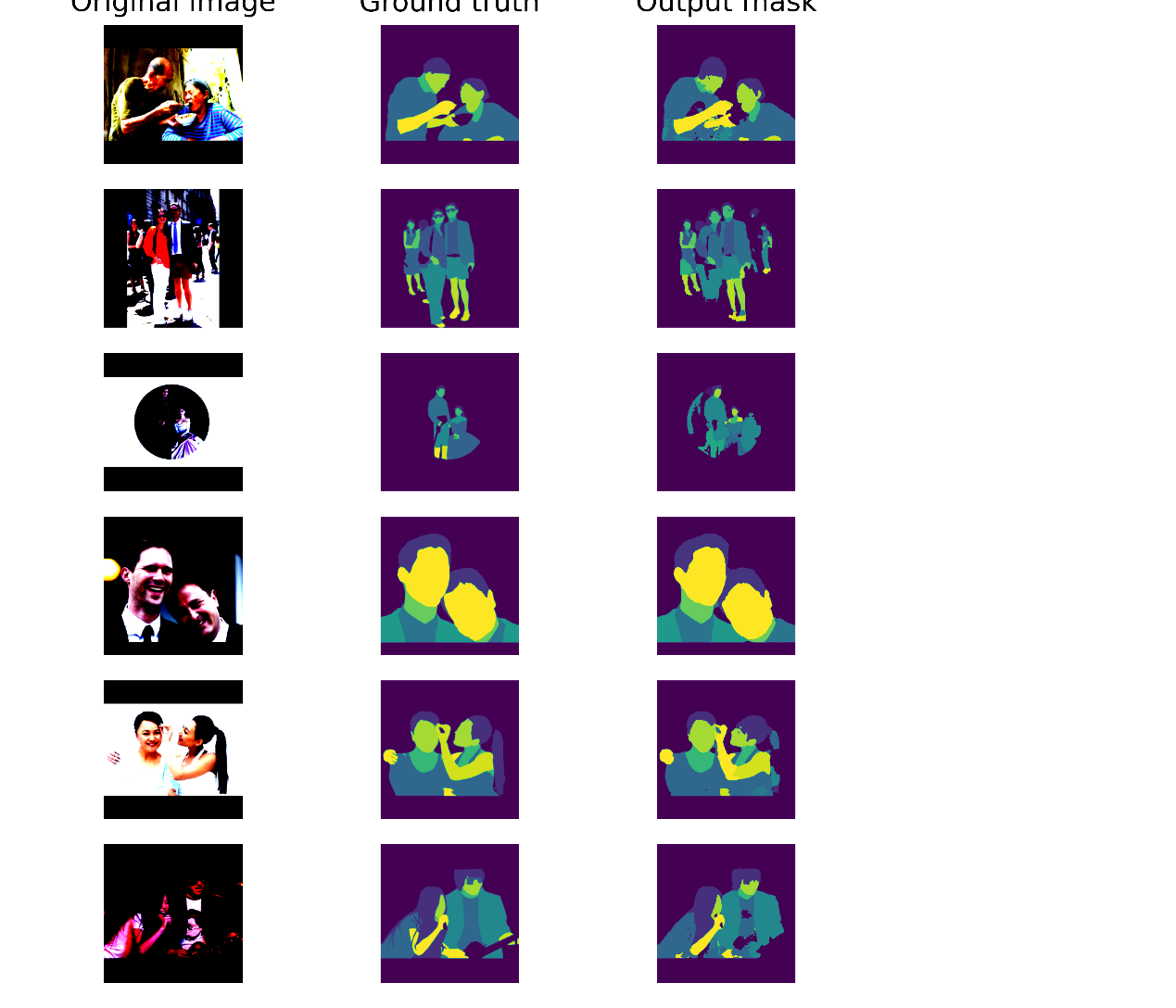
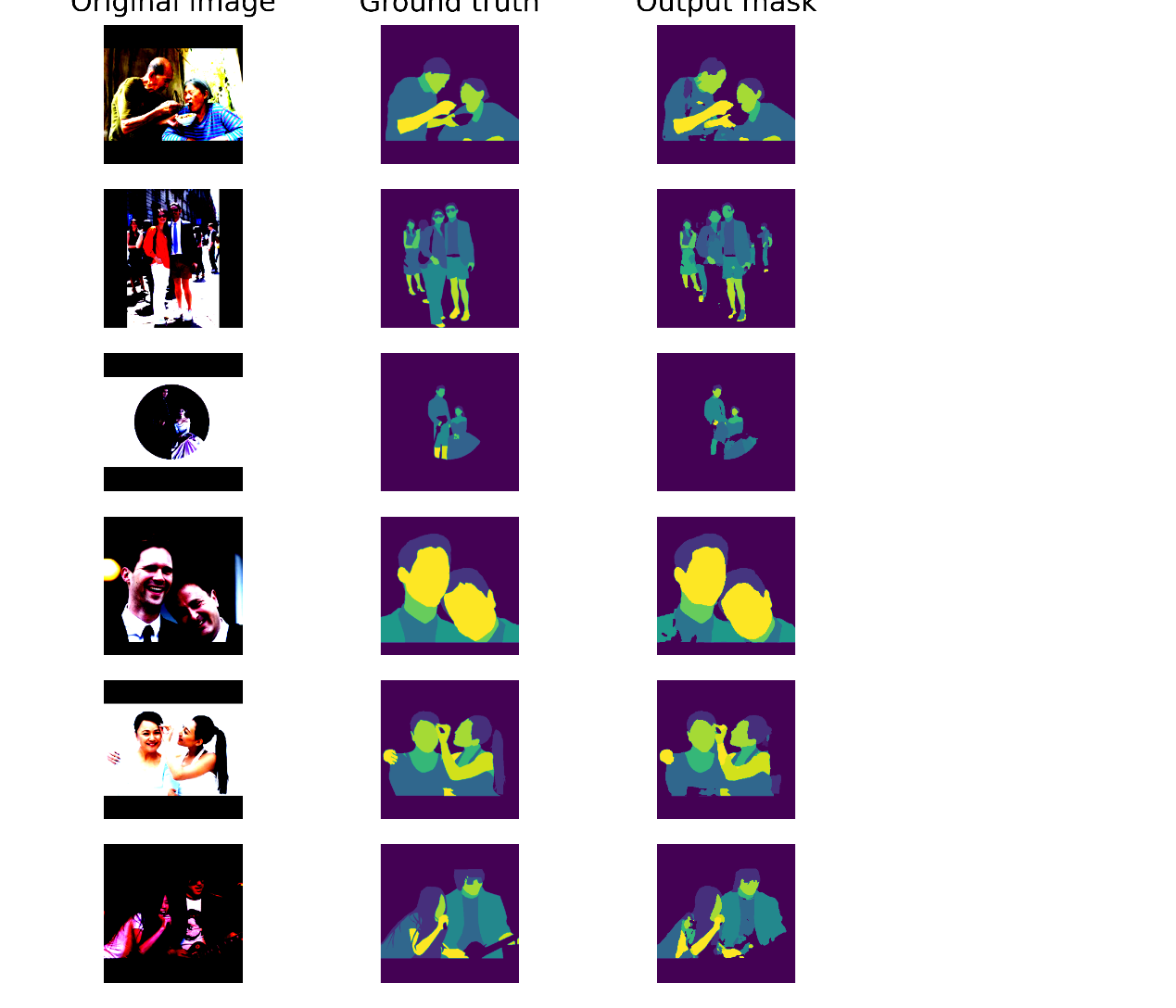
loss= CrossEntry

最后一层就是conv2d，其他添加，如sigmoid，tanh会让训练结果变差。

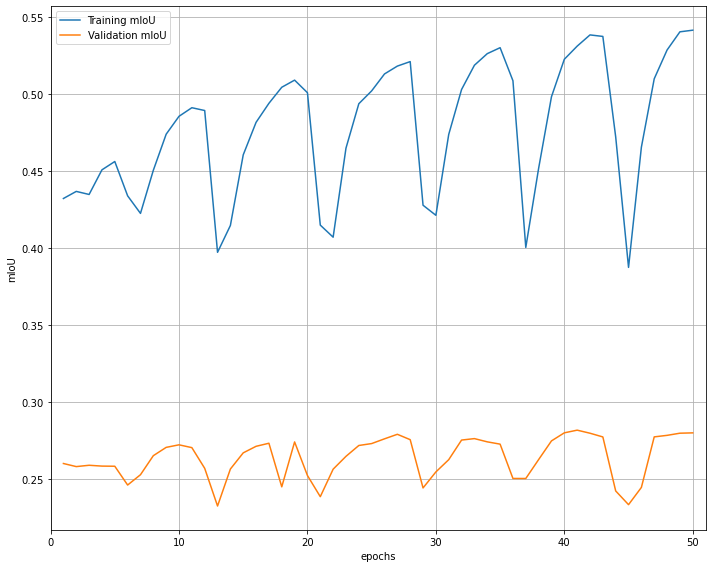
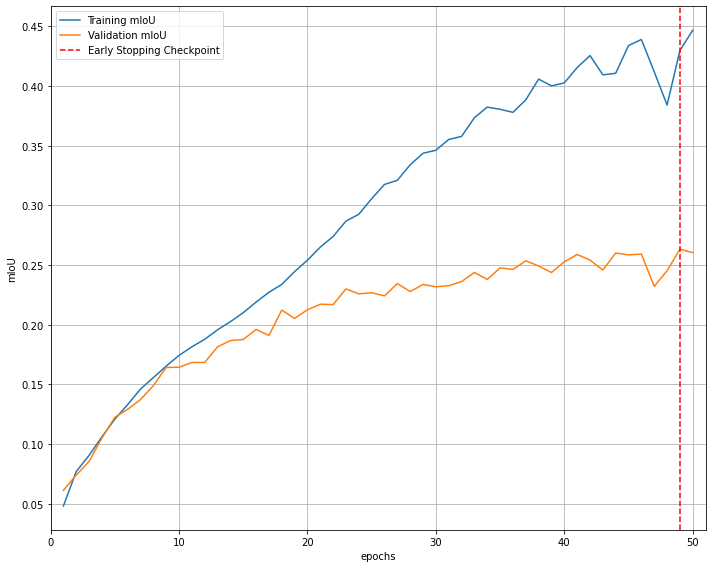
其前50epoch的训练效果：Valid\_mIoU= 25%,test\_mIoU=25%

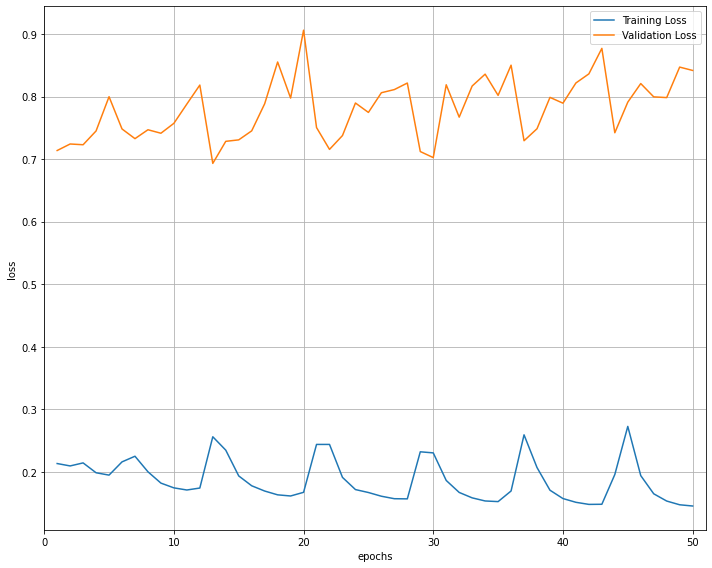
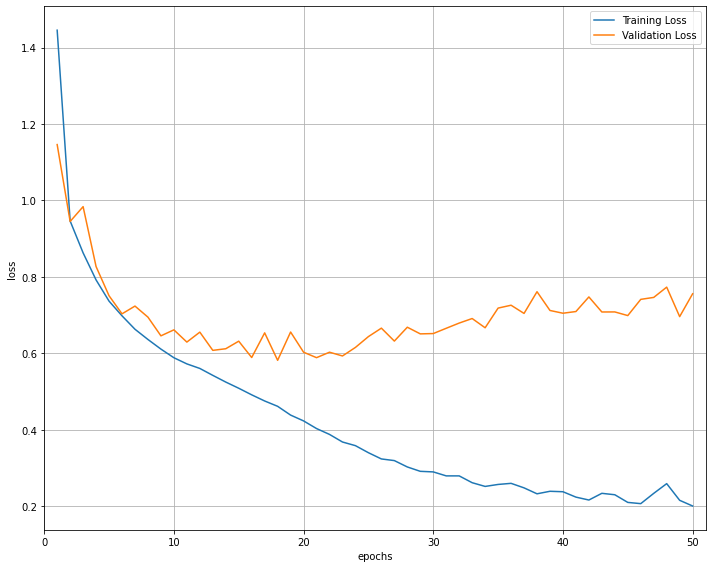
其后50epoch(第50-100epoch)的训练效果：Valid\_mIoU= 28%,test\_mIoU=26%

可以看出valid\_mIoU提升了不少，但test只提升了一点。为了比较test前后差异我把两次训练的结果拼接在下图，左三是前50epoch,左四是后50epoch：

可以看出，后50epoch中对第3，6幅图的mask质量有所提升，但在第2,4,5幅图的mask产出质量有所下降。





上一:前50epoch mIoU 上二:后50epoch mIoU

下一:前50epoch loss 下二:后50epoch loss

评价：这是在我使用Unet来做语义分割的多类分割的过程中mIoU表现最好的一个配置。

方案二：ResNet（18）+UNet

Resize=（256,256）

Batch\_size=32,

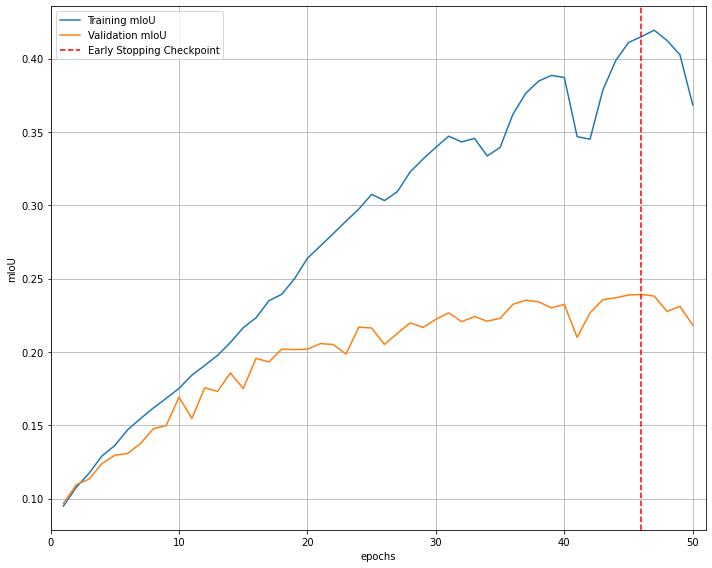
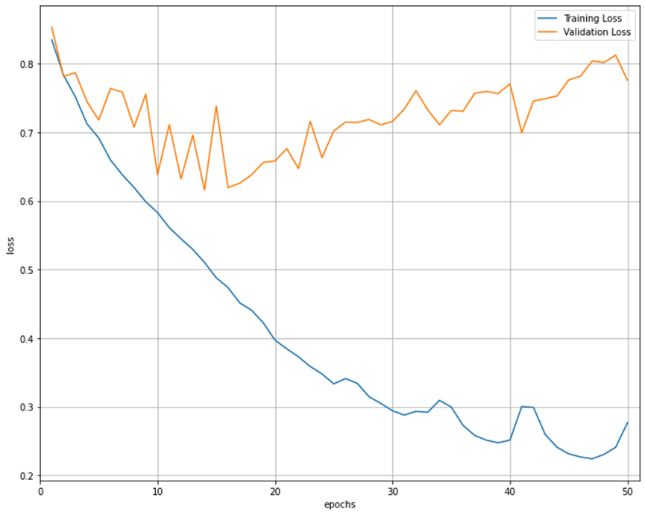
data\_augmentation=False，

loss= CrossEntry

最后一层就是conv2d

在体验过ResNet的优秀性能后，想把ResNet和Unet结合起来。在控制变量的思想下，我用的是Res18+Unet。

从Loss角度看，Res+Unet的表现很好，但从mIoU来看，其试验结果并没有显著提升，因此我并没有继续训练下去。可能的原因是Res18并不够深，如果换成Res50可能会好些。限于Cuda memory，与GPU使用时间上限，我并没有实现Res18+UNet与Res50+UNet的控制变量试验。



方案三：UNet

Resize=（256,256）

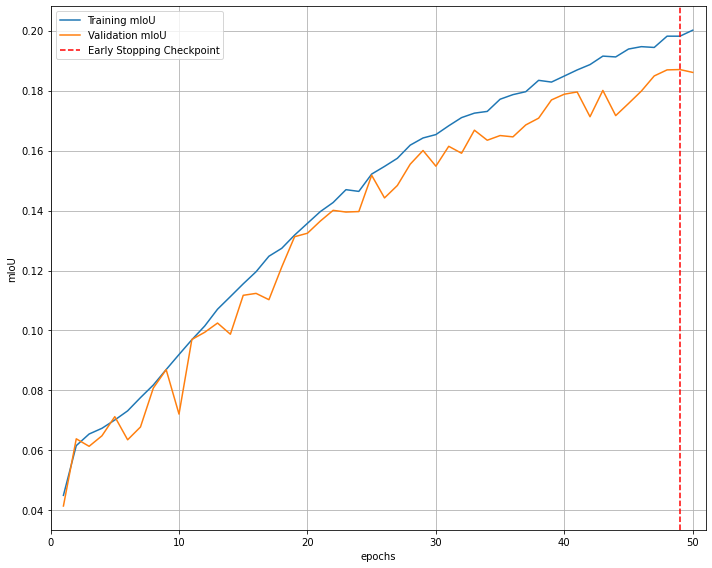
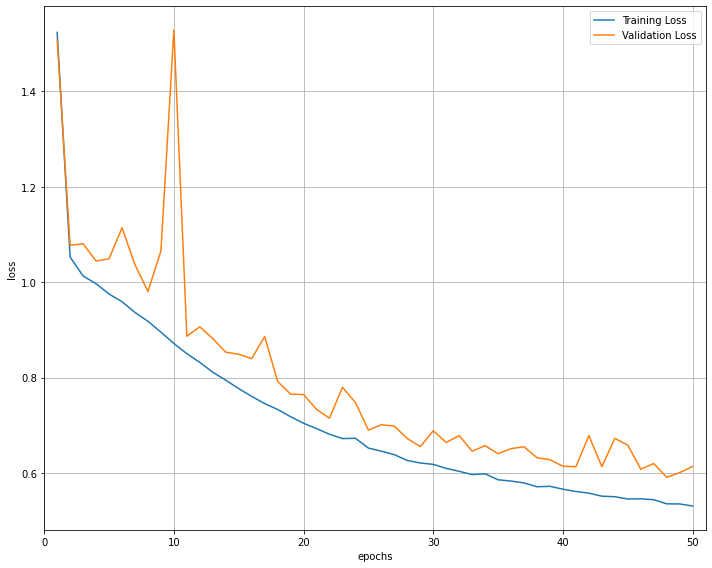
Batch\_size=32,

data\_augmentation=True，

loss= CrossEntry

最后一层就是conv2d

我采用的是随机rotation来做data augmentation。效果如下：



评价：时间性价比不划算，50epoch只到了未使用data augmentation的20epoch的效果。

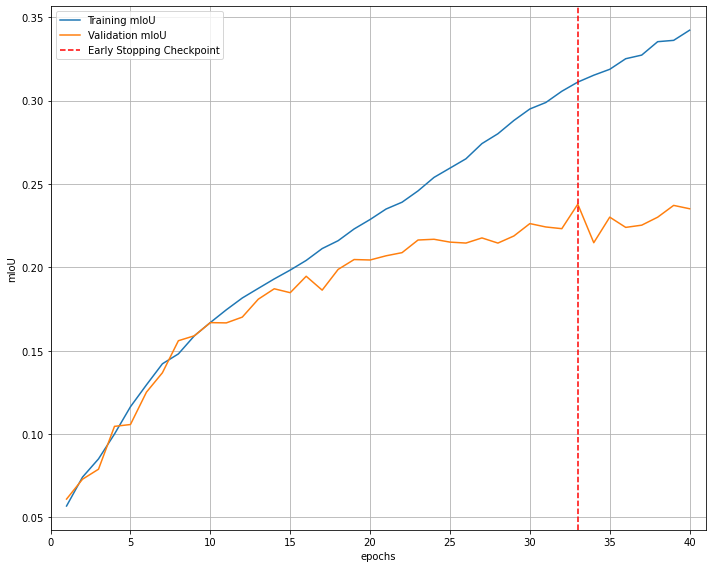
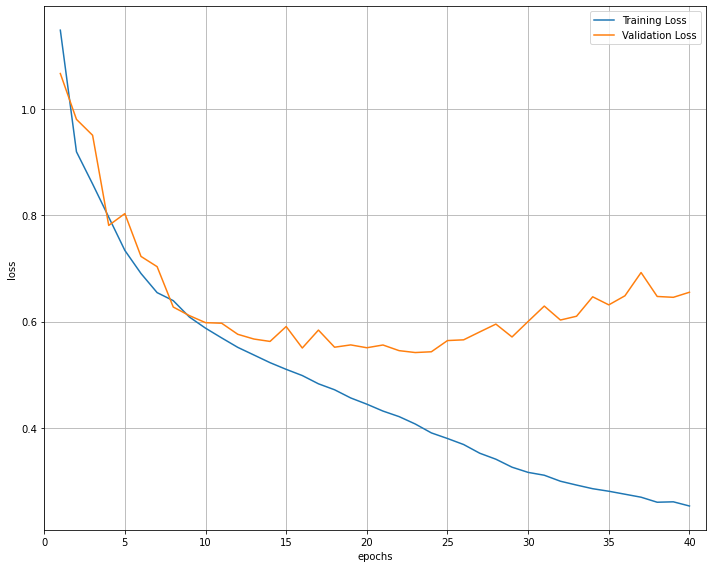
一组对比试验

|  |  |
| --- | --- |
| 方案四：UNet | 方案五：UNet |
| Resize=（448,448） | Resize=（256,256） |
| Batch\_size=8 | Batch\_size=8 |
| data\_augmentation=False | data\_augmentation=False |
| loss= CrossEntry | loss= CrossEntry |
| 最后一层就是conv2d | 最后一层就是conv2d |

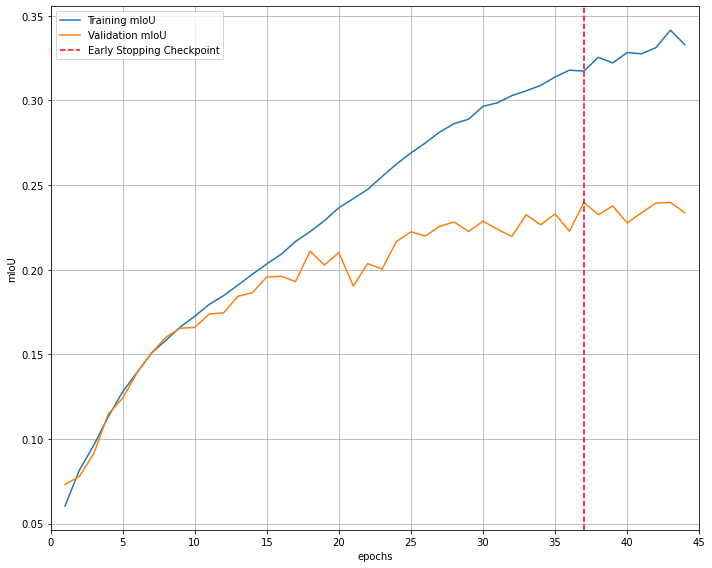
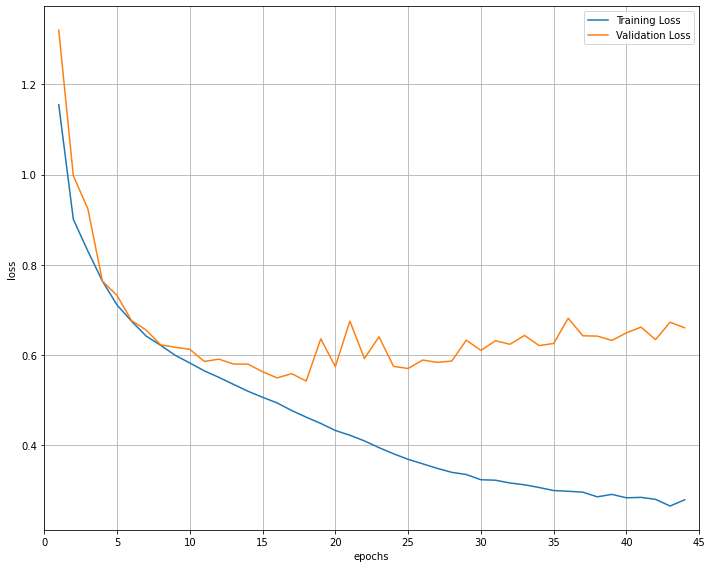
方案四与方案五是为了尝试resize时使用较大的size对结果的的影响。

受限于cuda memory，为了控制变量batch\_size我选择了8。

下面是比较的的结果



图一：size=488，loss 图二：size=488，mIoU



图三：size=488，loss 图四：size=488，mIoU

经过比较，发现resize的size改大后，loss与mIoU并未有突出的提升。推测的原因是对变多的像素点，模型没有更好地学习，所以size=448的与size=256在loss和mIoU的差异不大。

从耗时来看，size=448在计算loss与mIoU所花的时间更长。

更据网上一些论文使用的输入resize结果推测，size放大后，模型在mIoU的表现可能有所提高，但所花的时间很长，如果是为了冲刺比赛，在兼顾效率的同时可以把resize放大些，但像我在平时训练学习的过程中256的size就够用了。