本周主要将寒假全部编写完成 faster rcnn 之后的调试工作，将应用于单张图像的 GPU nms 操作修改成应用在连续帧的 tubes 之间的 GPU nms 操作，提高了网络的训练速度，并且在 UCFSports 上面做了几个的实验，但是发现效果并不是很好。越折腾越觉得这种现象迷。。

faster rcnn 基本思路：（整个过程中需要注意到的细节很多，刚开始看别人的代码有些操作不是很能理解，直到踩坑才意识到被忽视部分的重要性。）

- Feature map 的提取采用 resnet101
- 该方法的前面 rpn 网络基本和 SSD 相同，使用最后提取出的 feature map 上面的每一个点映射回原图，将在该点上预设的每一个 anchor 进行分类和回归。
- Rpn 网络的结果根据其二分类结果，取前景打分大于 0.01 的 anchor 进行 nms，最后 nms 剩下大约 200 个左右的 roi 送入 roi pooling 和最后的分类回归网络
- 根据 rpn 提出的 roi 和 ground truth 进行匹配（为了保证训练过程中至少有一个正样本，将 ground truth 也加入到 rpn 提取出的 roi 中，同时为了有负样本，在准备数据的阶段找出部分 iou 位于 0.1-0.3 之间的加入到 rpn rois 中），找出 iou>0.5 的作为正样本，iou<0.3 的为负样本，按照正负样本 1:3 的比例为每一个 batch 找出正负样本集。并为所有的正样本生成回归的目标值用于计算 loss 进行训练。
- 因为我的实验中要一次性堆叠 6 帧，所以只能一帧一帧单独进行 roi pooling 操作，将 rpn 输出的 roi tubes 结果分配到每一帧，这样 roi pooling 的结果也会随着 rpn 回归的结果在 feature map 上面的位置有所变化。
- 最后将 roi pooling 的结果送入回归和打分网络。

这周的实验主要是在 UCFSports 上面进行:

输入尺寸	Rpn 网络		Roi 处理方式	最终 map
	Recall	Map		
320*320	0.9769	0.9489	Roi pooling	0.7298
			Roi align	0.7898

从上表可以看出 rpn 网络提出的 proposals 的质量还是不错的，但是最终的 map 效果并不理想，可以看出使用 roi align 的效果比 pooling 的效果要好很多，320*320 的图像提取出的 feature map 仅有 20*20 的大小，而 roi pooling 的输出则有 7*7，从这一点上看有可能是 feature map 太小了。网络对验证集的推理结果中，**其定位基本上没有什么问题**，甚至在经过最后一层网络的定位后，单独考虑 rpn 网络的 recall 还会有提升，但是在**分类中的效果非常感人**。

之后**换用 640*640 的图**，这样提取出的 feature map 有 40*40，原本期待这样能有一个比较好的表现，但是该方案不仅速度奇慢，一张卡也只能装下一张图，rpn 网络的 recall 只有 0.7 不到，效果奇差。

为了解决 feature map 过小导致的特征损失较多的问题，考虑在 roi 的时候不仅选用最后一个 feature map，同时向前多用一个 feature map，这样对同一帧的两张分别为 20*20 和 40*40 的 feature map 进行 roi，希望通过这种方式能够保留更多的特征，提高检测效果。

向前多取一张 40*40 的 feature map，得到的结果:

输入尺寸	Rpn 网络		Roi 处理方式	最终 map
	Recall	Map		
320*320	0.961	0.925	Roi pooling	0.7745

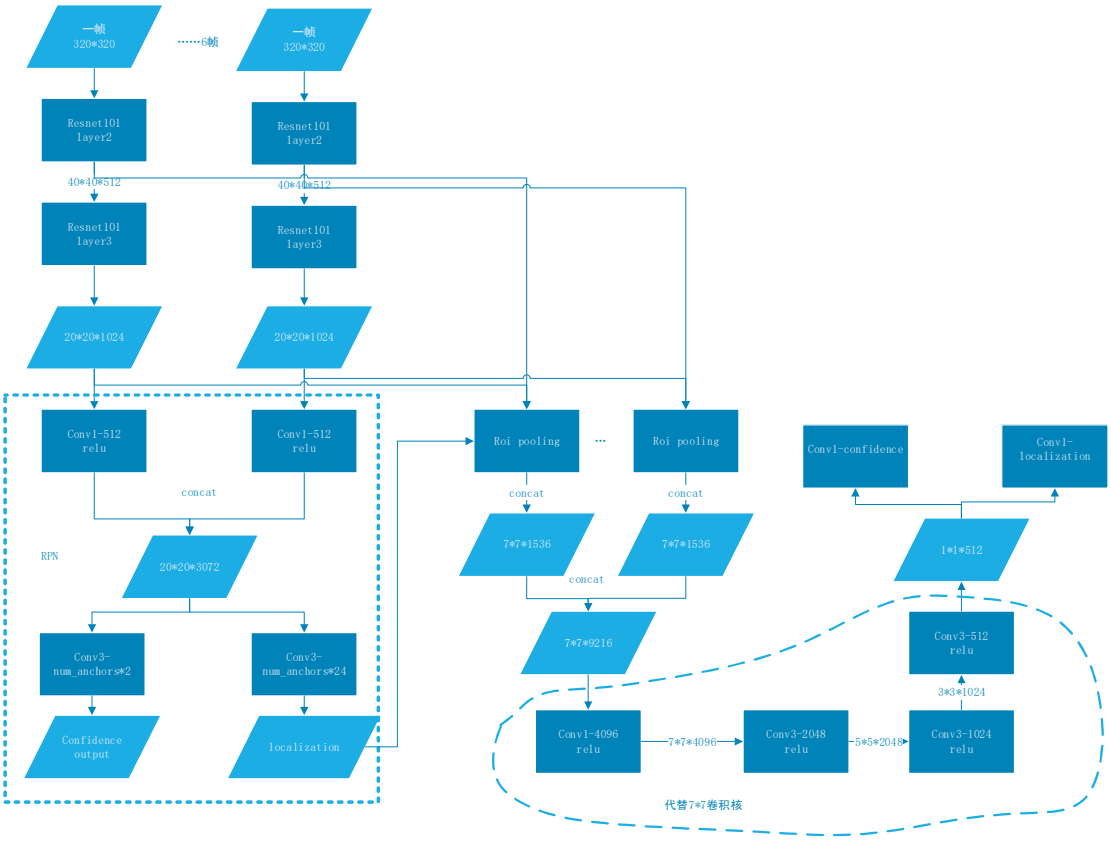
			Roi align	0.7703

可以看出，最终 map 仅仅使用 roi pooling 就可以达到之前 roi align 的效果，但是就算使用 roi align 也没有什么提升，注意到该方法在使用的过程中在**训练集上面的 map 高达 0.97，**所以显然在这个实验中过拟合很严重了。

希望找出出现这种结果的原因，考虑到在网络参数自己新加的层的初始化的过程中似乎是顺手自己写了参数的初始化函数，使用 pytorch 自带的初始化参数训练后发现很难收敛，同样有过拟合现象。尤其是最后的分类效果非常感人，我觉得这个网络这样的结果不应该啊。

现在对于这种现象的出现有点迷，这会不会是那一篇文章的作者不用 faster rcnn 而选用 SSD 的原因？

网络详细的结构图如下图所示，不知道是不是后面自己添加的层设计上有不合理的地方？



实验过程中注意到因为视频中人的移动，其实 6 帧连续取的时候有一小部分视频最大 iou 仅 0.2 不到，这个 iou 可以说是很小了，虽然数量不多，但是这种情况的出现说明这种方法存在的局限性。

另外在寻找正样本的过程中有个问题，就是如果只有一部分也认为完全是背景？也就是说 iou<0.4 就认为是负样本，我觉得这种设定是不合理的，虽然那个 anchor 的位置不太对，但是把这个**完全作为负样本这种一刀切**的方法，是不太能够接受的。带着这种想法逛到过一个人脸检测相关的文章，文中认为可以直接考虑回归 iou，而不是给出概率，我认为这样考虑就比较合理，但是只能够针对人脸检测这种特定的任务，因为人脸检测不需要分类。不过如果可以**用在 rpn 中的话**我认为应该还是会有效果的。