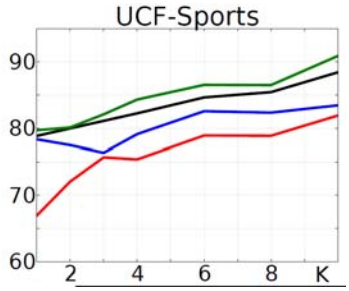
这周因为家里临时有点急事临时请假回去了两天，回来的路上感冒了，一直头疼，这几天精神状态比较差，干活效率比较低。

这周只完成了 map 的计算部分程序编写和调试，因为计算速度的问题（没占到显卡），目前只完成了 UCFSports 这个数据集 map 的计算，最终的结果是 0.8764，和作者论文上的数据基本相对应（图中的绿色曲线，K 的取值为 6）。UCF101 还在算…



数据集		K	map
UCFSports	RGB+FLOW(late)	6	0.8764

对于本文提到的方法考察 ap，是以每一帧为单位来考虑的。计算 ap 的过程和最终生成的 tubes 无关。

其关键思路总结如下：

- 在前面的预测中，每连续的 6 帧就有一个预测结果，所以其中会有多个帧是重复多次预测的，将各个预测结果分配到每一帧，这样每一帧上对应的同意 ground truth 就会有多个预测结果。
- 依次对每一帧上的所有预测结果利用 nms 将其合并
- 依次遍历每一个 label，对于预测的结果中存在该 label 的帧，要相应的在 ground truth 表中寻找到该帧当前 label 的 ground truth，然后计算 iou 来确定是否预测正确。（这里要想清楚的一点是要如何迅速找到拥有当前指定 label 的帧以及 ground truth，对于预测的结果，可以直接根据预测的 label 来进行寻找，但是对于 ground truth 则要求指定视频并且指定帧，指定 label，一个方便的方法就是将 ground truth 的结果以 key 为（视频 id，帧号，label），存储为字典，可以实现唯一且方便的查找。）
- 回到定义：
  - 召回率：所有 ground truth 预测出来了多少
  - 精度：预测的结果中有多少是正确的
- 所以 ground truth 的总数是不会改变的，初始化时将当前 label 下的 ground truth 数量定义为 fn，错误的正样本数量定义为 fp，正确的正样本数量定义为 tp，如果预测正确则 fn-1,tp+1,如果预测错误则 fp+1 那么

$$\text{precision} = \frac{tp}{fp + tp}$$

$$\text{recall} = \frac{tp}{tp + fn}$$

- 最后求取 precision-recall 曲线的面积即可求得 ap，将所有 label 的 ap 取平均值即可得到 map