本周主要看了Action Tubelet Detector for Spatio-Temporal Action Localization和Online real-time multiple spatiotemporal action localisation and prediction这两篇论文。其中第一篇论文中对anchor cuboids和action tubes这两个概念的理解较为吃力，因为根本无法理解action tubes存在的意义是什么，为什么要这么做。因为第一篇论文中提到其使用的算法来自第二篇论文，为了理解其思想，所以又跑去看了第二篇企图理解贪心算法。

论文 Online real-time multiple spatiotemporal action localisation and prediction 中对贪心算法生成 action tube:

对 tubes 的基本要求：允许 tubes 的数量随时间变化，也允许 tubes 随时开始或者停止

i、 连续检测的不完整的 action tubes 和 detection boxes 的 IOU 大于一个阈值

ii、 一个 action tube 中没有两个动作

iii、 Action tube 的 label 随时间更新。

算法的输入是融合后的从帧中提取的 boxes 以及它们对应每一个类别的打分。在每一帧中，帧中存在的每一个类都会用 nms 找出帧中前 n 个打分最高的 boxes。在视频的第一帧，nc(1)=n，即 action tubes 每一个类 c 的 action tubes 的个数被初始化为 n。算法会随着时间的增加给 tubes 每次增加一个 box，使 tubes 逐渐加长。Tubes 的数量 nc(t)随着时间变化，因为新的 tubes 会被加入，旧的可能被终止。

对于视频第一帧，对每一个动作类分别在图中拥有的 detection boxes 的个数，就作为起始的 action tubes 的开始。

对于后面每一个新考虑的帧，都要对每一个类进行如下操作:

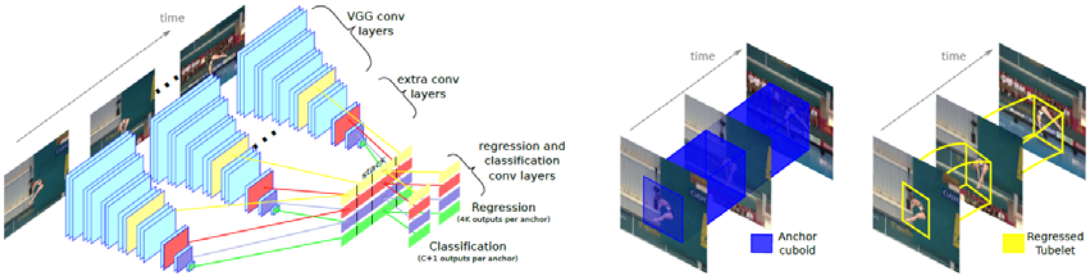
- 对属于当前类的已经存在的 action tubes，根据每一个 tube 里面的 detection boxes 的当前类的得分的平均值，对其进行降序排序。
- 遍历属于当前类的所有 action tubes，并去当前帧中寻找和当前 action tube 最匹配的 detection boxes:
  - 只要当前 detection box 和 tube 的最后一个 box 的 IOU 值大于了一个阈值，那么久将其加入候选列表。
  - 如果候选列表中的个数不为 0，那么寻找候选列表中在当前类的得分最高的加入 action tubes，同时将该 detection box 从该帧中删除。
  - 如果候选列表中没有候选项，则继续程序继续运行，也不增加新的 tube，如果后面连续 k 帧都没有找到匹配的，则这个 action tube 结束。
- 使用 viterbi dynamic programming 在线更新 tube 的 label。

Action Tubelet Detector for Spatio-Temporal Action Localization:

核心思想：使用同一个 SSD 网络对一系列帧分别进行一次卷积操作，将在不同尺度上得到的 feature map 分别进行叠加，最后分别通过一个分类和一个回归卷积网络得到视频中对动作的分类和出现的位置，这就是 tubelet，得到 tubelet 之后再使用贪心算法将每张图上的 tubelet 连接成 action tubes，以得到最终结果。

其中，最后使用贪心算法将 tubelet 连接成 action tubes 的操作是独立于卷积网络之外的后处理，它考虑到了 action tube 中一系列 bounding boxes 之间的联系，通过后处理可以有效提高识别率。**从本质上来说，尽管对 SSD 的输入是连续的 k 帧，但是 SSD 网络最后的操作是通过对这一系列堆叠的帧一起进行分析，实现了对每张图片上对每一个动作的预测，但是不能保证最后的卷积网络对这一系列堆叠帧进行处理的过程中一定能学到这些连续帧之间的联系。使用 action tubes 其实是通过一种较为暴力的方法来找出这一段视频中的人**

的连续的动作！将这一段视频中所有动作的打分做一个平均，是可以抑制其中某一帧出现误识别。假如连续的 10 帧中第 5 帧判断出错，那么其他的帧就会帮助纠正这个错误。



对 anchor cuboid 的理解：文中有提到，假设在 K 帧间 spatial extent is fixed over time（意思应该是被圈中的人在这 k 帧中的位置几乎不会发生变化）。将 SSD 的 anchor box 扩展为 anchor cuboids，在 SSD 的操作过程中，我们在每一个 feature map 上的每一个点所应用的不同 aspect ratio 和 scale 的 anchor boxes 都是可以映射回原图的一部分区域，那么可以考虑将对应的区域扩展到全部的 K 帧，在截取的这一段连续的 K 帧内，因为人的动作都是连续的，可以认为人不出去这个 anchor box。在对每一帧都分别进行了回归操作之后，每一帧上的 anchor cuboid 又会去追踪实际的目标，最后在每一帧上都形成 tubelet。

Action tubes 的生成：

为每一个动作类输入 10 个经过 nms 之后的最高得分的 tubelets，之后的操作与另一篇论文中的贪心算法几乎相同。其目的是将每张图上属于同一个人的同一个动作通过一种抽象的逻辑联系起来。

但是我认为有一条改进的思路就是可以再设定一个 IOU 的阈值，这个阈值并不只是针对一个类，而是说，假设其中有某一帧判断出错，将这一块区域识别成其他的动作类，但是发现这一块区域和上一帧和下一帧都及其吻合，例如 IOU>0.9，基于前面的假设（每一个 detection box 中只有一个动作），这样的情况我认为是可以考虑将其进行纠正的。只是不知道这种错误出现的可能性有多大。

准确率（precision）：正确预测为正占全部预测为正的比例（给出的结果有多少是正确的）

召回率（recall）：正确预测为正占全部正样本的比例（正确的结果又多少被给出了）

针对某一个类，预测输出的 score，将 score 排序，其前 K 个样本预测为正样本，可以计算（P,R），调节阈值 K 可以得到 P-R 曲线，通过计算 P-R 曲线和坐标轴围成的面积即可得到该类的 AP，那么对所有的类取均值即可得到 mAP

通常情况下，召回率越高准确率越低，但是如果一个网络不仅召回率高，而且准确率高，那么围成的面积为 1，也就是 AP 为 1，这当然是极好的。可以理解 mAP 可以用于计算一个网络的综合性能。