前言

我们在之前的博文[1]中曾经花了很长的篇幅介绍了视频理解与表征的一些内容，当然，由于篇幅原因，其实还是省略了很多内容的，特别是一些比较新的研究成果都没能进行介绍，在本文，我们继续我们的视频理解之旅。 **如有谬误，请联系指出，转载请联系作者，并且注明出处。谢谢。**

∇ 联系方式： **e-mail**: FesianXu@gmail.com **QQ**: 973926198 github: https://github.com/FesianXu

视频理解再回首

我们之前在[1]中进行过比较全面的关于视频理解的介绍，其中包括有基于多模态（RGB视频，骨骼点序列等）的动作识别，也有若干基于RGB视频的经典网络架构等，同时讨论了一些多视角动作识别和自监督视频理解的一些内容。我们在这篇博客中，暂时不会考虑其他模态的视频序列，而是主要会聚焦在基于RGB视频的理解上。一般来说，基于RGB视频的主要工作都集中在以下几个类别之中，如Fig 1.1所示。

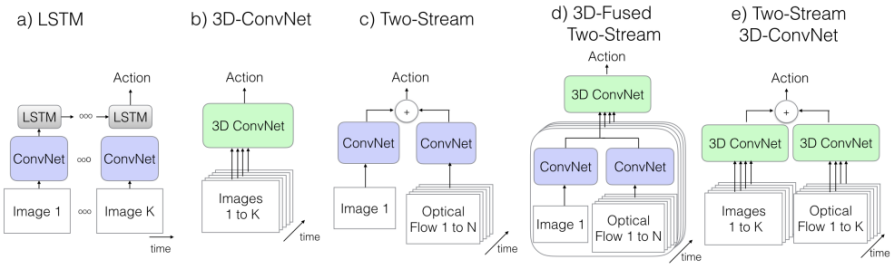

Fig 1.1 在视频理解任务中常见的五种网络结构。

这几种网络结构的优劣之处我们在[1]中做过较为详细的介绍，这里就不再赘述了。而在本文中，我们要再介绍几种比较有趣的网络构型，这几种网络结构和以上提到的网络结构存在较大的不同，值得玩味。

基于图结构的视频理解

我们发现Fig 1.1所示的这些模型都假设视频的发生是 **线性**的。这意味着我们的拍摄过程是一镜到底的，也就是说不存在有镜头的切换。这种类型的视频存在于自然视频拍摄过程中，占有目前视频存量的一定比例。但是，还有一大类的视频是 **非线性** 的。比如存在编导进行视频镜头的切换，典型的例子是体育比赛的直播场景，还有一些存在视频片段剪辑的场景也是非线性的。在这类型的视频中，若干场景或者事件可能交替发生。如Fig 2.1所示，我们发现场景的镜头聚焦的人物一直在变化，其中颜色框相同的代表是相同关注的人物或者事件，那么Fig 2.1的镜头下的事件发生顺序就变成：

```
A --> B --> C --> A --> D --> B --> D --> A --> D ...
```

这种存在明显的非线性镜头切换的视频，如果用普通的视频分类模型，比如R(2+1)D模型[2]进行识别，效果势必会大打折扣，因为网络设计并没有考虑这些镜头切换，而镜头切换导致的语义不连续对于特征提取来说是会造成负面影响的。

Fig 2.1 场景的镜头一直在切换，因此不是一个线性发生的视频。

为了显式地对这种非线性的事件序列进行建模，在[7]中作者提出了用图结构去组织视频中帧间的关系，如Fig 2.2所示，其中相同颜色的方块代表是相同场景的帧，通过图（graph）的方式可以对帧间的视觉关系进行组织，使得视觉相似的帧之间的连接强度更强，而视觉效果差别较大的帧之间连接强度更弱，从而有帧级别的关系图(Frame level graph)。当然，帧间的关系可以通过图池化（graph pooling），将相似的帧聚合成一个镜头（shot），进一步可以聚成事件（event）。最终形成整个视频级别的特征嵌入。

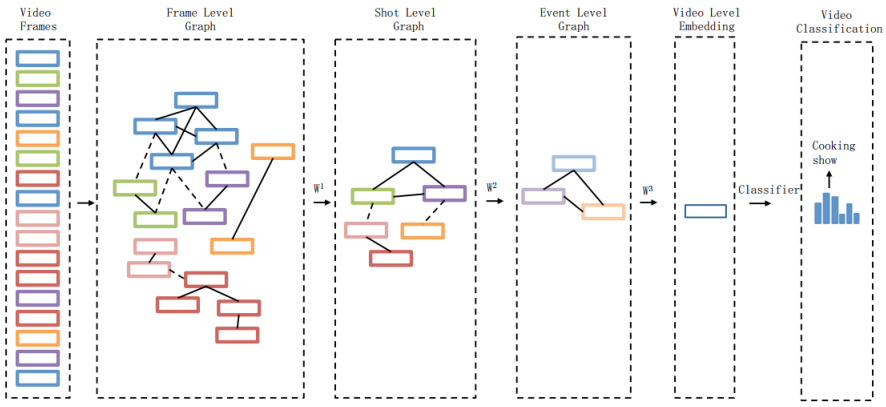

Fig 2.2 用图结构去组织同一个视频的帧间关系，其中相同颜色的表示同一个场景或者镜头下的视频帧。

我们发现，通过图的数据组织形式，我们给网络提供了学习非线性事件发生的能力。那么既然需要建立一个图结构数据，其图拓扑结构就很重要，我们可以通过计算帧特征之间的余弦相似度去衡量不同帧之间的相似程度，将其视为邻接矩阵。计算方式如(2.1)所示

$$\mathrm{A}_{i,j} = \frac{\sum_{d=0}^{D-1}(f_{i,d} \times f_{j,d})}{\sqrt{\sum_{d=0}^{D-1} f_{i,d}^2} \times \sqrt{\sum_{d=0}^{D-1} f_{j,d}^2}} \tag{2.1}$$

其中的$\mathbf{F} \in \mathbb{R}^{N \times D}$表示具有$N$帧并且每一帧提取出来的特征维度为$D$的帧特征矩阵，而$f_{i,d}$是该矩阵的第$i$行第$d$列。计算出来的邻接矩阵的可视化结果如Fig 2.3所示，其中我们发现的确可以把某些相同场景或者镜头下的帧进行聚合。

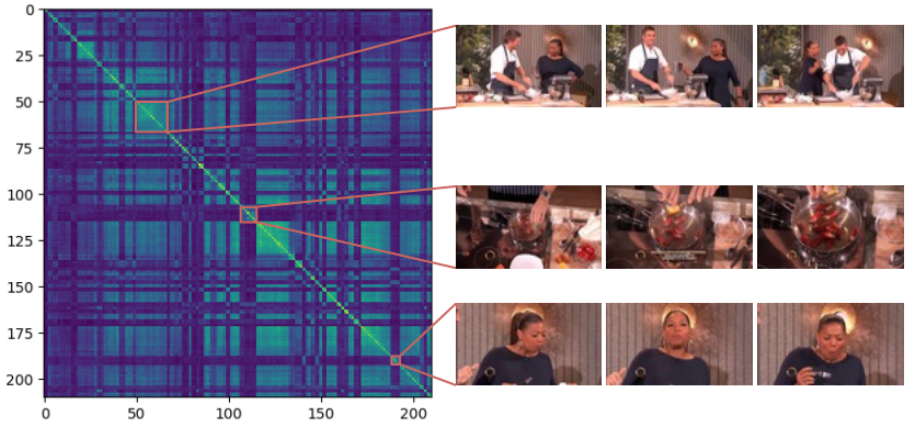

Fig 2.3 通过余弦相似度计算，可以计算视频帧间的视觉相似程度，这个相似程度可以视为是图的邻接矩阵。

我们曾在博文[3,4,5]介绍过图卷积网络以及非欧几里德结构数据，我们在这里将要涉及到部分以前讨论过的知识，读者又有需要可以移步详读。因为我们现在有了每帧的特征，也有了图的邻接矩阵，因此我们可以用式子(2.2)，以消息传递（message passing）的方式去表达图卷积，有：

$$h^l = G(\mathrm{A}, h^{l-1}, W^l) \tag{2.2}$$

按照我们之前在[3,4,5]中介绍的，我们知道$G(\cdot)$是一个消息传递函数，可以表示为一种特殊形式的矩阵乘法（存在矩阵的加权），有：

$$G = \sigma(\sqrt{\overline{D}}\mathrm{A}\sqrt{\overline{D}}h^{l-1}W^l) \tag{2.3}$$

其中的标准化后的邻接矩阵A的度数矩阵，具体计算可以参考[5]。

然而，只是根据(2.3)对图进行卷积操作无法进行场景，镜头的聚合，我们的图结构仍然只是静态的。因此我们需要引入图池化操作去进行帧信息的“浓缩”，我们也可以理解为是视频不同层次，粒度信息的聚合。我们可以采用的图池化操作有很多，比如存在导数，可以端到端学习的`DiffPool`[6]，也有单纯的`AveragePool`和引入了局部自注意机制的`AveragePool`。基于`DiffPool`的方式能提供更为灵活的，端到端的池化方式，但是在文章[7]中，作者只是探索了后两种池化方式。我们暂且先关注到这两种方式。

1. `Average Pool`：该方法的motivation很简单，就是即便是存在镜头场景切换，某个帧的前后若干连续帧也是大概率是连续的，也即是具有“局部连续”的特性。那么我们只需要设置一个超参数K作为池化核大小，在K帧内进行聚合即可，因此有：

$$p^l[i,d] = \frac{\sum_{k=0}^{K-1} h^{l-1}[i \times K + k][d]}{K}, d \in [0, D] \tag{2.4}$$

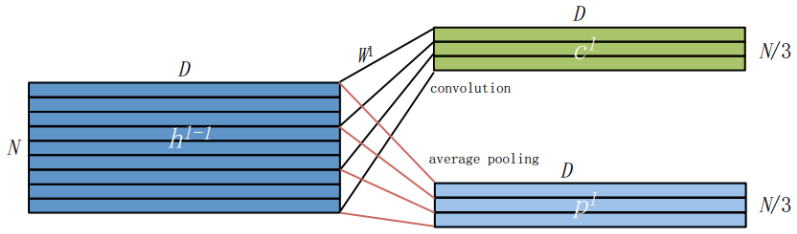

Fig 2.4 本文采用的average pool方法示意图

该池化过程如Fig 2.4所示，其中$k=3$。

2. `self-attention based pool`：基于局部自注意机制的池化，其会对局部连续的帧进行加权，需要学习出加权系数$\alpha^l[i]$。

$$\begin{aligned}\alpha^l[i] &= \text{softmax}(h^{l-1}[i\times K:(i+1)\times K]W^l_{\text{att}}+b^l)\\ p^l_{\text{att}}[i] &= \alpha^l[i]\otimes h^{l-1}[i\times K:(i+1)\times K]\end{aligned} \tag{2.4}$$

其中$W^l_{\text{att}}\in\mathbb{R}^{D\times 1}$，其中$\otimes$是克罗内克积(kronecker product)[8]，即是

$$A\otimes B=\begin{bmatrix}a_{11}B & \cdots & a_{1n}B\\ \vdots & \ddots & \vdots\\ a_{m1}B & \cdots & a_{mn}B\end{bmatrix} \tag{2.5}$$

也就是说，对于原输入的每K个帧，我们通过注意力矩阵W^l_{att}和偏置b^l共同学习得到一个权重系数$\alpha^l[i]$，这个权重系数负责这K个帧的权重。整个过程如Fig 2.5所示，其中$K=3$。

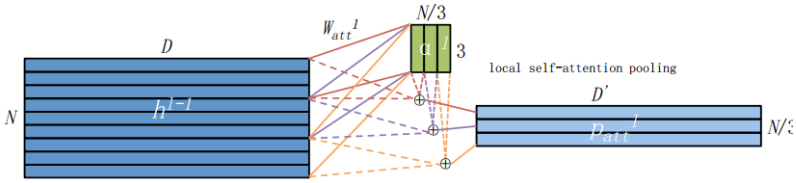

Fig 2.5 本文采用的self-attention based pooling，其根据输入特征学习出权重矩阵，将其给输入特征进行加权。

一般而言，在图卷积中我们用全连接网络进行节点关系的组织，如公式(2.3)所示，其中的$h^{l-1}W^l$其实本质上就是一个全连接操作。那么在文章[7]中，作者用卷积操作取代了全连接操作，有：

$$c^l[i]=h^{l-1}[i\times K:(i+1)\times K]W^l \tag{2.6}$$

其中$W^l\in\mathbb{R}^{D\times D_2}$，$D_2$表示下一层的特征数量，本文中有$D_2=D$。图示如Fig 2.6所示。

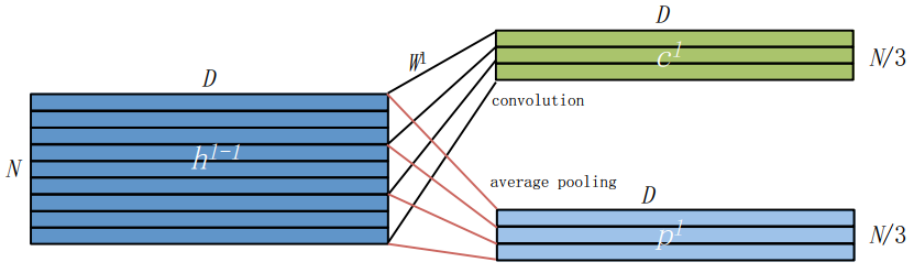

Fig 2.6 绿色区域表示是用“卷积操作”取代“全连接操作”进行图卷积的过程。

当然，因为在进行图池化过程中，图拓扑势必发生了变化，我们需要更新发生了变化之后的图结构的邻接矩阵，重新通过公式(2.1)计算即可，当然需要把输入的f替换成p^{l-1}。

Reference

[1]. https://blog.csdn.net/LoseInVain/article/details/105545703

[2]. Tran D, Wang H, Torresani L, et al. A closer look at spatiotemporal convolutions for action recognition[C]//Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2018: 6450-6459.

[3]. https://blog.csdn.net/LoseInVain/article/details/88373506

[4]. https://blog.csdn.net/LoseInVain/article/details/90171863

[5]. https://blog.csdn.net/LoseInVain/article/details/90348807

[6]. Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., & Leskovec, J. (2018). Hierarchical graph representation learning with differentiable pooling. In *Advances in neural information processing systems* (pp. 4800-4810).

[7]. Mao, Feng, et al. "Hierarchical video frame sequence representation with deep convolutional graph network." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

[8]. https://zh.wikipedia.org/wiki/%E5%85%8B%E7%BD%97%E5%86%85%E5%85%8B%E7%A7%AF