

图卷积梳理

袁铭潮

1. 卷积

介绍图卷积之前必须对离散卷积（也就是 CNN 中所使用的卷积计算）有明确的认识。

从数学上讲，卷积是对两个实变函数的一种数学运算。泛涵分析中，卷积（Convolution）是通过两个函数 f 和 g 生成第三个函数的一种数学算子，表示函数 f 和 g 经过反转和平移的重叠部分函数值乘积对重叠长度的积分，是一种特殊的线性运算。

连续为： $(f * g)(n) = \int_{-\infty}^{\infty} f(\tau)g(n - \tau)d\tau$ 。

离散为： $(f * g)(n) = \sum_{\tau=-\infty}^{\infty} f(\tau)g(n - \tau)d\tau$ 。

把二元函数 $U(x, y) = f(x)g(y)$ 卷成一元函数 $V(t)$ ，也可以看作是一种降维的方法。

由于式子中的两个函数或者说两个变量是平等的，一种可取的办法就是沿直线 $x + y = t$ 卷起来，得到 $V(t) = \int_{x+y=t} U(x, y)dx$ 。

上文中卷积定义的两个式子的共同特点是式中： $n = \tau + (n - \tau)$

若是令 $x = \tau$ ， $y = n - \tau$ ，那么 $x + y = n$ 就是与 x 轴交点从左往右，斜率为 -1 的直线。

卷积运算有两个效应：

(1) 展宽效应：加入函数只在一个有限区间内不为零，这个区间可称为函数的宽度。一般来说，卷积函数的宽度等于被卷函数宽度之和。

(2) 平滑效应：被卷函数经过卷积运算，其细微结构在一定程度上被消除，函数本身的起伏振荡变得平缓圆滑。

2. 离散卷积的应用

由于应用和篇幅的限制，本文仅对离散卷积举例说明。

卷积关系最重要的一种情况，是在信号与线性系统或数字信号处理中的卷积定理。利用该定理，可以将

时间域或空间域中的卷积运算等价于频率域的相乘运算，从而利用 FFT 等快速算法，实现有效计算。

但我们关注最多的还是图像处理领域的应用。

用一个模板和一幅图像进行卷积，对于图像上的一个点，让模板的原点和该点重合，然后模板上的点和图像上对应的点相乘，然后各点的积相加，就得到了该点的卷积值。对图像上的每个点都这样处理。由于大多数模板都是对称的，所以模板不旋转。卷积是一种积分运算，用来求两个曲线重叠区域面积。可以看作加权求和，可以用来消除噪声、特征增强。

把一个点的像素值用其周围的点的像素值的加权平均代替。

图像处理中常见的 mask 运算都是卷积，广泛应用于图像滤波。

3. 卷积网络

卷积网络（Convolutional network），也叫卷积神经网络（CNN），是一种专门用来处理具有类似网格结构的数据的神经网络。例如时间序列数据（可以看作在时间轴上有规律采样形成的一维网格）和图像数据（看作二维像素网格）。

这里回顾一下前文中的卷积定义： $(f * g)(n) = \sum_{\tau=-\infty}^{\infty} f(\tau)g(n - \tau)d\tau$ 。

在卷积网络中，卷积的第一个参数通常叫作输入（input），第二个参数叫作核函数（kernel function），输出有时被称作特征映射。

在机器学习中，输入通常是多维数组的数据，而核通常是由学习算法优化得到的多维数组的参数。这些多维数组通常称为张量。通常假设在存储了数值的有限点集以外，这些函数的值都为零，实际操作中通过对有限个数组元素的求和来实现无限求和。

卷积运算不局限于一维，如果把一张二维图像 I 作

为输入，二维的核为 K ，则式子可以改写为：

$$I * K(i, j) = \sum_m \sum_n I(m, n) K(i - m, j - n)$$

根据卷积的交换性质，等价写作：

$$K * I(i, j) = \sum_m \sum_n I(i - m, j - n) K(m, n)$$

进行交换的原因是使得核 K 的尺寸更小一些，实现更加简单。

离散的卷积可以看作矩阵的乘法。卷积通常对应着一个非常稀疏的矩阵，这是因为核的大小通常要远小于输入图像的大小。任何一个使用矩阵乘法但是不依赖矩阵结构的特殊性质的神经网络算法，都适用于卷积运算。

4. 机器学习使用卷积的原因是什么？

4.1. 稀疏连接

传统的神经网络通过矩阵乘法来建立输入与输出的连接关系，其中，参数矩阵中每一个单独的参数都描述两个输入单元与一个输出单元间的交互。这意味着每一个输出单元与每一个输入单元都产生交互。

然而，卷积网络具有稀疏连接的特征。这是核的大小远小于输入大小所达到的，这意味着需要存储的参数更少，不仅减少了模型的存储需求，还提高了统计效率，也意味着需要的计算量更少。

例如 m 个输入和 n 个输出，矩阵乘法需要 $m \times n$ 个参数，对于每个例子时间复杂度为 $O(m \times n)$ 。如果限制每个输出拥有的连接数为 k ，那么稀疏连接方法只需要 $k \times n$ 个参数以及 $O(k \times n)$ 的时间复杂度，实际应用中，保持 k 比 m 小几个数量级，能够取得很好的效率提升。

4.2. 参数共享

在传统的神经网络中，当计算一层的输出时，权重矩阵的每一个元素只是用一次。在卷积神经网络中，核的每一个元素都作用在输入的每一个位置上，卷积运算的参数共享保证了我们只需要学习一个参数集合，而不是对每一位置都学习一个单独的参数集合。

对于卷积，参数共享的特殊形式使得神经网络层具有对平移等变的性质，但是对其他一些变换并不是天然等变的，比如图像的缩放或者旋转变换。

一些不能被传统矩阵乘法定义的神经网络处理的特殊数据，可能通过卷积神经网络来处理，比如一维的音频波形数据、二维的使用傅里叶变换预处理过的音频数据、三维的体积数据等。

5. 图卷积

CNN 可以很有效地提取空间特征，但是 CNN 的局限性在于其所处理的数据必须是排列整齐的矩阵，也就是很多论文中提到的欧几里德结构的数据。

与之相对的，科学研究中还有很多非欧几里德结构的数据，例如社交网络、信息网络等等，这样的网络结构就是图论中抽象意义上的拓扑图。所以图卷积中的图是指图论中的用顶点和边建立相应关系的拓扑图。

那为什么要研究图卷积呢？

1、CNN 无法处理非欧几里德结构的数据，严谨的表述是在非欧几里德结构的数据上无法保持平移不变性。个人的理解是，图中每个顶点的相邻顶点数不同，无法使用同样尺寸的卷积核来进行卷积运算。

2、由于 CNN 无法处理非欧几里德结构的数据，又希望在这样的数据结构上有效地提取空间特征来进行机器学习，需求推动 GCN 的研究。

3、广义上，任何数据在赋范空间内都可以建立拓扑关联，所以拓扑连接是一种广义的数据结构，因此 GCN 的应用范围很广。

GCN 的本质目的是提取拓扑图上的空间特征，目前主流的方式有空域和频域两种，分别对应两种思路。

6. 基于空域的图卷积

基于空域的图卷积的思路是找出可在相邻节点数量变化的情况下进行特征抽取的卷积核。

问题是非常直观的，就是把每个顶点相邻的顶点找出来。

这里面的包含的科学问题有两个：

(1) 按照什么条件去找中心顶点的相邻顶点，也就是如何确定感受野？

(2) 确定感受野，按照什么方式处理包含不同数量相邻节点的特征？

基于空域的图卷积的主要缺点是：

(1) 每个顶点的相邻顶点数不同，使得计算处理必须针对每个顶点。

(2) 提取特征的效果可能没有卷积好。

所以基于空域的图卷积方面的研究比较少，个人目前看过的论文中使用基于空域图卷积的论文仅仅只有一篇。

7. 基于频域的图卷积

基于频域的图卷积的思路是把非欧式空间转换成欧式空间，把图当作信号进行处理，通过傅里叶变换得到频域上的欧式空间，再进行卷积。

这种思路就是希望借助图谱的理论来实现拓扑图上的卷积操作，从研究进程看，首先是研究图信号处理的学者定义了图上的傅里叶变换，进而定义了图上的卷积，最后与深度学习结合提出了图卷积网络。

这里涉及两个平时不太接触的知识：傅里叶变换和拉普拉斯算子。

7.1. 傅里叶变换和拉普拉斯算子

维基百科的说法是，傅里叶变换会将一个在空域上定义的函数分解成频域上的若干频率成分。

傅里叶变换对于卷积的重要性在于一个公式：

$$f * g(t) = F^{-1}[F[f(t)] \odot F[g(t)]]$$

这里的 F^{-1} 指的是傅里叶逆变换， \odot 是逐点乘积。

上述公式的直观含义是：空（时）域卷积等于频域卷积，解释为如果要计算两个函数的卷积，可以先将它们通过傅里叶变换变换到频域中，将两个函数在频域中相乘，然后通过傅里叶逆变换变换回空（时）域，得到的就是两个函数的卷积结果。

傅里叶变换的公式：

$$\hat{f}(t) = \int f(x) \exp^{-2\pi i x t} dx$$

\hat{f} 是 f 经过傅里叶变换后的结果，其中 $i = \sqrt{-1}$ ， t 是任意实数。

对于傅里叶变换公式，我们所关注的是 $\exp^{-2\pi i x t}$ 的物理意义，这也是在图傅里叶变换的关键所在，这个式子实际上是拉普拉斯算子 Δ 的广义特征函数。

拉普拉斯算子的物理意义是空间二阶导，准确定义是：标量梯度场中的散度，一般可用于描述物理量的

流入流出。比如说在二维空间中的温度传播规律，一般可以用拉普拉斯算子来描述。

根据拉普拉斯算子的定义推导，根据特征向量的定义有： $Ax = \lambda x$ ，则可得拉普拉斯算子作用在 $\exp^{-2\pi i x t}$ 上满足：

$$\Delta \exp^{-2\pi i x t} = \frac{\partial^2}{\partial t^2} \exp^{-2\pi i x t} = -4\pi^2 x^2 \exp^{-2\pi i x t}$$

到此可以得出傅里叶变换的式子，本质是将函数 $f(t)$ 映射到以 $\{\exp^{-2\pi i x t}\}$ 为基向量的空间中。

7.2. 图傅里叶变换

对图进行傅里叶变换的问题在于怎么找到对应的拉普拉斯算子和 $\exp^{-2\pi i x t}$ 。

目前所使用的图傅里叶变换的方案是用图的拉普拉斯矩阵替换拉普拉斯算子，拉普拉斯矩阵的特征向量替换 $\exp^{-2\pi i x t}$ 。

这里给出拉普拉斯矩阵 L 的定义：

有 n 个顶点的无向图 $G = (V, E)$ ，根据 G ，得到度矩阵 D 和邻接矩阵 A ， $L = D - A$ 。

Labeled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

图 1. 拉普拉斯矩阵计算方法

根据瑞利熵可以得到对称归一化的拉普拉斯矩阵 $L^{sys} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ ，其中 I 为单位矩阵。

GCN 经典论文中使用的就是这种拉普拉斯矩阵。

为什么要使用拉普拉斯矩阵？

首先，拉普拉斯矩阵是对称矩阵，可以进行谱分解，特征向量互相正交。

$$L = U \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} U^{-1} \rightarrow L = U \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} U^T$$

其中 $U = (U_1, U_2, \dots, U_n)$ 。

其次，拉普拉斯矩阵只在中心顶点和一阶相连的顶点上有非零元素，其余均为零。

这里还需要指出拉普拉斯矩阵的局限性，拉普拉斯矩阵是基于无向图定义的，也就意味着，对于有向图，GCN 是无能为力的。

7.3. 图傅里叶变换公式

把传统的傅里叶变换以及卷积迁移到图上，最核心的工作就是把拉普拉斯算子的特征函数编程图对应的拉普拉斯矩阵的特征向量。

根据 $Ax = \lambda x$ ，相同形式的 $LU = \lambda U$ 。采用离散积分来定义图上的傅里叶变换：

$$F(\lambda_l) = \hat{f}(\lambda_l) = \sum_{i=1}^N f(i) u_l^*(i)$$

f 是图上的 N 维向量， $f(i)$ 与顶点一一对应， $u_l(i)$ 表示第 l 个特征向量的第 i 个分量，在特征值 λ_l 下， f 的傅里叶变换就是与其对应的特征向量进行内积运算，由于本次内积是定义在复数空间，所以用的是 $u_l^*(i)$ 表示特征向量的共轭。

利用矩阵乘法将上式推广到矩阵形式：

$$\begin{pmatrix} \hat{f}(\lambda_1) \\ \hat{f}(\lambda_2) \\ \vdots \\ \hat{f}(\lambda_N) \end{pmatrix} = \begin{pmatrix} u_1(1) & \dots & u_1(N) \\ u_2(1) & \dots & u_2(N) \\ \vdots & \ddots & \vdots \\ u_N(1) & \dots & u_N(N) \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ \vdots \\ f(N) \end{pmatrix}$$

传统傅里叶变换的逆变换是对频率进行积分，这里的频率为 ω ，等价于前文中的 $2\pi x$ ，可以将公式改写为：

$$\mathcal{F}^{-1}[F(\omega)] = \frac{1}{2\pi} \int F(\omega) \exp^{i\omega t} d\omega$$

迁移到图上变为对特征值 λ_l 求和：

$$f(i) = \sum_{l=1}^N \hat{f}(\lambda_l) u_l(i)$$

同样推广到矩阵形式：

$$\begin{pmatrix} f(1) \\ f(2) \\ \vdots \\ f(N) \end{pmatrix} = \begin{pmatrix} u_1(1) & \dots & u_1(N) \\ u_2(1) & \dots & u_2(N) \\ \vdots & \ddots & \vdots \\ u_N(1) & \dots & u_N(N) \end{pmatrix} \begin{pmatrix} \hat{f}(\lambda_1) \\ \hat{f}(\lambda_2) \\ \vdots \\ \hat{f}(\lambda_N) \end{pmatrix}$$

7.4. 图上的卷积

参照 CNN，图上的卷积的目标是求的输入函数 f 和卷积核 h 之间的卷积结果。

f 的傅里叶变换为 $\hat{f} = U^T f$ 。

卷积核 h 的傅里叶变换写成对角矩阵的形式为：

$$\begin{pmatrix} \hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{pmatrix}$$

$\hat{h}(\lambda_l) = \sum_{i=1}^N h(i) u_l^*(i)$ 是根据需要设计的卷积核 h 在图上的傅里叶变换。

两者乘积为：

$$\begin{pmatrix} \hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{pmatrix} U^T f$$

再乘以 U 求傅里叶变换的乘积的逆变换，则为卷积：

$$(f * h)_G = U \begin{pmatrix} \hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{pmatrix} U^T f$$

很多论文中的形式为： $(f * h)_G = U((U^T h) \odot (U^T f))$ ，与推出的式子是等价的。

8. 图卷积网络的研究现状

GCN 提出时，使用的方式极为简单粗暴，将 $\hat{h}(\lambda_l)$ 替换成 θ_l ，整体公式为：

$$y_{output} = \sigma \left(U \begin{pmatrix} \theta_1 & & \\ & \ddots & \\ & & \theta_n \end{pmatrix} U^T x \right)$$

这种方式替换方式简单，但是存在一些弊端，每次向前传播都需要计算三个矩阵的乘积，对于大规模的图，计算代价较高，论文中给出的复杂度为 $O(n^2)$ ，需要 n 个参数，且卷积核不具有局部性。

ICLR2017 的一篇文章对 GCN 的卷积核进行了优化，使用切比雪夫多项式的 K 阶截断来近似 $\hat{h}(\lambda_l)$ 。

$$y_{output} = \sigma \left(U \begin{pmatrix} \sum_{j=0}^{j=K} \alpha_j \lambda_1^j & & \\ & \ddots & \\ & & \sum_{j=0}^{j=K} \alpha_j \lambda_n^j \end{pmatrix} U^T x \right)$$

这个式子貌似看上去更加复杂了，其实不然：

$$\begin{pmatrix} \sum_{j=0}^{j=K} \alpha_j \lambda_1^j & & \\ & \ddots & \\ & & \sum_{j=0}^{j=K} \alpha_j \lambda_n^j \end{pmatrix} = \sum_{j=0}^K \alpha_j \Lambda^j$$

$$U \sum_{j=0}^K \alpha_j \Lambda^j U^T = \sum_{j=0}^K \alpha_j U \Lambda^j U^T = \sum_{j=0}^K \alpha_j L^j$$

由公式 $L^2 = U\Lambda U^T U\Lambda U^T = U\Lambda^2 U^T$ ，进一步简化：

$$y_{output} = \sigma\left(\sum_{j=0}^K \alpha_j L^j x\right)$$

相较于最初的图卷积版本，有三个优势：

- 1、卷积核只有 K 个参数，一般 K 远小于 N ，参数复杂度降低了。
- 2、根据最后的公式，可以发现不需要进行谱分解了，但是由于要计算 L^j ，时间复杂度依旧是 $O(n^2)$ 。
- 3、卷积核具有很好的局部性， K 就是卷积核的感受野。

目前最新的研究成果是利用小波变换进行图卷积，大大降低了时间复杂度，具体论文未曾拜读。

9. 总结

到基于切比雪夫多项式的图卷积已经基本与 CNN 中的卷积效用相同了。于是，其局限性遍被凸显出来两，利用拉普拉斯矩阵模拟卷积的方式，由于拉普拉斯矩阵本身的限制，导致不能兼容有向图。

对于有向图的图卷积目前理论上尚未完备，仅有少数几篇论文涉及，且方法复杂效果普通。这个方向的研究突破，可能对图卷积产生一次巨大的推动，图卷积全面替代卷积并非不可能。