人工智能　深度学习（Deep Learning）　图卷积神经网络 (GCN)

关注者 6,530　被浏览 549,393

如何理解 Graph Convolutional Network（GCN）？

期待大佬们深入浅出的讲解。

关注问题　写回答　邀请回答　添加评论　分享　···

37 个回答　默认排序

Johnny Richards
程序员

2,006 人赞同了该回答

最高票 @superbrother 答案从傅立叶变换出发到拉普拉斯矩阵最后再到GCN已经很详细和全面的讲解，清晰明确非常赞 。
然而，发表一点个人意见，我认为这个回答对于初学者不太友好，直接看可能容易晕，主要的困难在于，最开始的Fourier变换概念太多，而在触及**GCN的本质Laplacian矩阵**的时候，描述过于抽象，而Laplacian矩阵直观的数学意义和感受被隐藏进参考文献了，于是我在这里做个补充，权当抛砖引玉。
先说问题的本质：**图中的每个结点无时无刻不因为邻居和更远的点的影响而在改变着自己的状态直到最终的平衡，关系越亲近的邻居影响越大。**
要想理解GCN以及其后面一系列工作的实质，最重要的是理解其中的精髓**Laplacian矩阵**在干什么。知道了Laplacian矩阵在干什么后，剩下的只是**解法的不同**——所谓的Fourier变换只是将问题从空域变换到频域去解，所以也有直接在空域解的（例如GraphSage）。
为了让问题简单便于理解，先让我们忘记时域、频域这些复杂的概念，从一个最简单的物理学现象——热传导出发。

图（Graph）上的热传播模型

众所周知，没有外接干预的情况下，热量从温度高传播到温度低的地方并且不可逆，根据著名的牛顿冷却定律（Newton Cool's Law），热量传递的速度正比于温度梯度，直观上也就是某个地方A温度高，另外一个B地方温度低，这两个地方接触，那么温度高的地方的热量会以正比于他们俩温度差的速度从A流向B。

从一维空间开始

我们先建立一个一维的温度传播的模型，假设有一个均匀的铁棒，不同位置温度不一样，现在我们刻画这个铁棒上面温度的热传播随着时间变化的关系。预先说明一下，一个连续的铁棒的热传播模型需要列**温度对时间和坐标的偏微分方程**来解决，我们不想把问题搞这么复杂，我们把**空间离散化**，假设铁棒是一个**一维链条**，链条上每一个单元拥有一致的温度，温度在相邻的不同的单元之间传播，如下图：

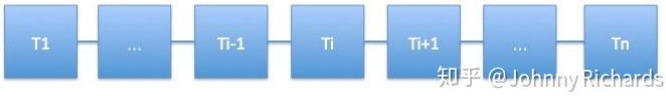


图1.一维离散链条上的热传播

对于第 i 个单元，它只和 $i-1$ 与 $i+1$ 两个单元相邻，接受它们传来的热量（或者向它们传递热量，只是正负号的差异而已），假设它当前的温度为 ϕ_i ，那么就有：

$$\frac{d\phi_i}{dt} = k(\phi_{i+1} - \phi_i) - k(\phi_i - \phi_{i-1})$$

k 和单元的比热容、质量有关是个常数。右边第一项是下一个单元向本单元的热量流入导致温度升高，第二项是本单元向上一个单元的热量流出导致温度降低。做一点微小的数学变换可以得到：

$$\frac{d\phi_i}{dt} - k[(\phi_{i+1} - \phi_i) - (\phi_i - \phi_{i-1})] = 0$$

注意观察第二项，它是两个差分的差分，在离散空间中，相邻位置的差分推广到连续空间就是**导数**，那么差分的差分，就是**二阶导数**！
所以，我们可以反推出铁棒这样的连续一维空间的热传导方程就是：

$$\frac{\partial \phi}{\partial t} - k\frac{\partial^2 \phi}{\partial x^2} = 0$$

同理，在高维的欧氏空间中，一阶导数就推广到**梯度**，二阶导数就是我们今天讨论的主角——**拉普拉斯算子**：

$$\frac{\partial \phi}{\partial t} - k\Delta\phi = 0$$

其中 Δ 这个符号代表的是对各个坐标二阶导数的加和，现在的主流写法也可以写作 ∇^2 。
综上所述，我们发现这样几个事实：
1、在欧氏空间中，某个点温度升高的速度正比于该点周围的温度分布，用拉普拉斯算子衡量。
2、拉普拉斯算子，是二阶导数对高维空间的推广。
那么，你肯定会问：你扯这么多有什么用呢？我还是看不到拉普拉斯算子和拉普拉斯矩阵还有GCN有半毛钱关系啊？
不要急，目前只是第一步，让我们把这个热传导模型推广导拓扑空间，你就会发现它们其实刻画的是同一回事了！

图(Graph)上热传播模型的推广

现在，我们依然考虑热传导模型，只是这个事情不发生在欧氏空间了，发生在一个Graph上面。这个图上的每个结点（Node）是一个单元，且这个单元只和与这个结点相连的单元，也就是有边（Edge）连接的单元发生热交换。例如下图中，结点1只和结点0、2、4发生热交换，更远的例如结点5的热量要通过4间接的传播过来而没有直接热交换。

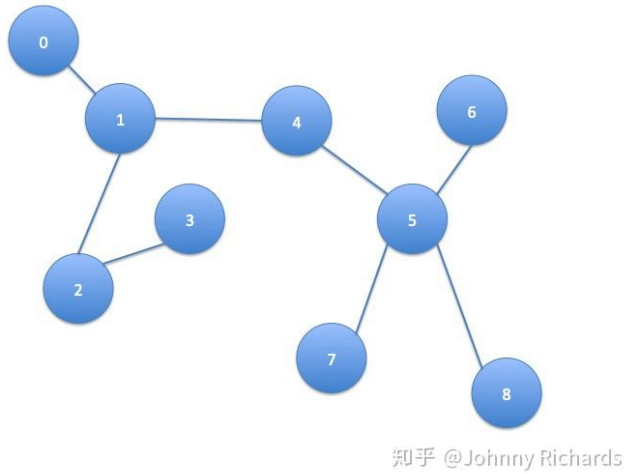


图2. 图上的热传播

我们假设热量流动的速度依然满足牛顿冷却定律，研究任一结点 i ，它的温度随着时间的变化可以用下式来刻画：

$$\frac{d\phi_i}{dt} = -k\sum_j A_{ij}(\phi_i - \phi_j)$$

其中 A 是这个图的邻接矩阵（Adjacency Matrix），定义非常直观： 对于这个矩阵中的每一个元素 A_{ij} ，如果结点 i 和 j 相邻，那么 $A_{ij} = 1$ ，否则 $A_{ij} = 0$ 。在这里，我们只讨论简单情况：

1、这张图是无向图， i 和 j 相邻那么 j 和 i 也相邻，所以 $A_{ij} = A_{ji}$ ，这是个对称阵。

2、结点自己到自己没有回环边，也就是 A 对角线上元素都是 0 。

所以不难理解上面这个公式恰好表示了只有相邻的边才能和本结点发生热交换且热量输入（输出）正比于温度差。

我们不妨用乘法分配律稍微对上式做一个推导：

$$\begin{aligned}\frac{d\phi_i}{dt} &= -k[\phi_i\sum_j A_{ij} - \sum_j A_{ij}\phi_j] \\ &= -k[deg(i)\phi_i - \sum_j A_{ij}\phi_j]\end{aligned}$$

先看右边括号里面第一项， $deg(\cdot)$ 代表对这个顶点求度（degree），一个顶点的度被定义为这个顶点有多少条边连接出去，很显然，根据邻接矩阵的定义，第一项的求和正是在计算顶点 i 的度。

再看右边括号里面的第二项，这可以认为是邻接矩阵的第 i 行对所有顶点的温度组成的向量做了个内积。

为什么要作上述变化呢，我们只看一个点的温度不太好看出来，我们把所有点的温度写成向量形式再描述上述关系就一目了然了。首先可以写成这样：

$$\begin{bmatrix}\frac{d\phi_1}{dt}\\ \frac{d\phi_2}{dt}\\ \cdots\\ \frac{d\phi_n}{dt}\end{bmatrix} = -k\begin{bmatrix}deg(1)\times\phi_1\\ deg(2)\times\phi_2\\ \cdots\\ deg(n)\times\phi_n\end{bmatrix} + kA\begin{bmatrix}\phi_1\\ \phi_2\\ \cdots\\ \phi_n\end{bmatrix}$$

然后我们定义向量 $\boldsymbol{\phi} = [\phi_1, \phi_2, \ldots, \phi_n]^T$ ，那么就有：

$$\frac{d\boldsymbol{\phi}}{dt} = -kD\boldsymbol{\phi} + kA\boldsymbol{\phi} = -k(D-A)\boldsymbol{\phi}$$

其中 $D = diag(deg(1), deg(2), \ldots, deg(n))$ 被称为度矩阵，只有对角线上有值，且这个值表示对应的顶点度的大小。整理整理，我们得到：

$$\frac{d\boldsymbol{\phi}}{dt} + kL\boldsymbol{\phi} = 0$$

回顾刚才在连续欧氏空间的那个微分方程：

$$\frac{\partial\phi}{\partial t} - k\Delta\phi = 0$$

二者具有一样的形式！我们来对比一下二者之间的关系：

- 相同点：刻画空间温度分布随时间的变化，且这个变化满足一个相同形式的微分方程。
- 不同点：**前者刻画拓扑空间有限结点**，用向量 $\boldsymbol{\phi}$ 来刻画当前状态，单位时间状态的变化正比于线性变换 $-L$ 算子作用在状态 $\boldsymbol{\phi}$ 上。**后者刻画欧氏空间的连续分布**，用函数 $\phi(\boldsymbol{x}, t)$ 来刻画当前状态，单位时间状态变化正比于拉普拉斯算子 Δ 作用在状态 ϕ 上。

不难发现，这就是**同一种变换、同一种关系在不同空间上面的体现**，实质上是一回事！

于是我们自然而然，可以把连续空间中的热传导，推广到图（Graph）上面去，我们把图上面和欧氏空间地位相同变换，以矩阵形式体现的 L 叫做拉普拉斯（Laplacian）矩阵。看，这正是 @superbrother 答案中所述的原始形式的拉普拉斯矩阵 $L = D - A$ 。

需要多嘴一句的是，本文开头所说的离散链条上的热传导，如果你把链条看成一个图，结点从左到右编号1，2，3...的话，也可以用图的热传导方程刻画，此时 D 除了头尾两个结点是1其他值都是2， A 的主对角线上下两条线上值是1，其他地方是0。

推广到GCN

现在**问题已经很明朗**了，只要你给定了一个空间，给定了空间中存在一种东西可以在这个空间上流动，两邻点之间流动的强度正比于它们之间的状态差异，那么**何止是热量可以在这个空间流动，任何东西都可以！**

自然而然，假设在图中各个结点流动的东西不是**热量**，而是**特征（Feature）**，而是**消息（Message）**，那么问题自然而然就被推广到了GCN。**所以GCN的实质是什么，是在一张Graph Network中特征（Feature）和消息（Message）中的流动和传播！这个传播最原始的形态就是状态的变化正比于相应空间（这里是Graph空间）拉普拉斯算子作用在当前的状态。**

抓住了这个实质，剩下的问题就是怎么去更加好的建模和解决这个问题。

建模方面就衍生出了各种不同的算法，你可以在这个问题上面复杂化这个模型，不一定要遵从牛顿冷却定律，你可以引入核函数、引入神经网络等方法把模型建得更非线性更能刻画复杂关系。

解决方面，因为很多问题在频域解决更加好算，你可以通过Fourier变换把空域问题转化为频域问题，解完了再变换回来，于是便有了所有Fourier变换中的那些故事。

扯了这么多，总结一下，问题的本质就是：

1. 我们有张图，图上每个结点刻画一个实体，物理学场景下这个实体是某个有温度的单元，它的状态是温度，广告和推荐的场景下这个实体是一个user，一个item，一个ad，它的状态是一个embedding的向量。
2. 相邻的结点具有比不相邻结点更密切的关系，物理学场景下，这个关系是空间上的临近、接触，广告和推荐场景下这个是一种逻辑上的关系，例如用户购买、点击item，item挂载ad。
3. 结点可以传播热量/消息到邻居，使得相邻的结点在温度/特征上面更接近。

本质上，这是一种Message Passing，是一种Induction，卷积、傅立叶都是表象和解法。

最后再补充**说明几点事实**：

热/消息传导方程的数值可迭代求解性（机器学习上的可操作性）

我们可以把原方程写成这样：

$$\frac{d\phi}{dt} = -kL\phi$$

机器学习中，时间是离散的，也就是左边对时间的求导变成了离散的一步步迭代。好在这个方程天生似乎就是上帝为了我们能够迭代求解而设计的。右边用拉普拉斯算子作用一次到全局的状态上，就能把状态更新一步！

实际解决的过程中，可以发挥机器学习搬砖工懂得举一反三的优良精神，首先，不一定要全局操作，我们可以batchify操作一部分结点，大家轮着来，其次，我们可以只考察这个点的一阶和二阶邻居对当前点作Message Passing，这个思想就是对拉普拉斯算子作特征分解，然后只取低阶的向量，因为矩阵的谱上面能量一般具有长尾效应，头几个特征值dominate几乎所有能量。

Laplacian算子的另一个性质

Laplacian矩阵/算子不仅表现的是一种二阶导数的运算，另一方面，它表现了一种加和性，这个从图上热/消息传播方程最原始的形态就能一目了然：

$$\frac{d\phi_i}{dt} = k\sum_j A_{ij}(\phi_j - \phi_i)$$

可见，每个结点每个时刻的状态变化，就是所有邻居对本结点差异的总和，也就是所有的邻居把message pass过来，然后再Aggregate一下，这正是GraphSage等空域算法的关键步骤**Aggregate思想**的滥觞。

在实际建模中，我们的Aggregate不一定是加和，作为一个熟练的机器学习搬砖工，我们懂得可以把Aggregate推广成各种操作例如Sum Pooling，例如LSTM，例如Attention，以求刷效果，发paper :)

两种空间下的Fourier分解/ 特征分解对比（卷积为什么能从欧氏空间推广到Graph空间）

最后，因为有以上分析作基础，我们可以解释一下傅立叶变换是怎么推广到图空间的。

首先，在欧氏空间上，迭代求解的核心算子 Δ 算子的特征函数是： $e^{-j\omega t}$ ，因为：

$$\Delta e^{-j\omega t} = \omega^2 e^{-j\omega t}$$

其次，在图空间上，对应地位的 L 对应的不是特征函数，而是它的特征值和特征向量，分别对应上面的 ω^2 和 $e^{-j\omega t}$ 。

唯一的区别是， $e^{-j\omega t}$ 是定义内积为函数积分的空间上的一组基，有无穷多个，当然这无穷多个是完备的。而 L 的特征空间是有限维的，因为 L 的对称性和半正定性，所有特征值都是实数且特征向量都是实向量，特征空间维度等于图中的结点数也就是 L 的阶。

注意，实对称矩阵的特征空间的所有基能够张出整个线性空间且它们两两正交，所以无论是拉普拉斯算子 Δ 还是拉普拉斯矩阵 L ，它们的特征空间是一个满秩且基两两正交的空间，所以把欧氏空间的 $e^{-j\omega t}$ 推广到图空间的 L 的这一组特征向量。**正是同一个关系（Message Passing），同一种变换，在不同空间下的体现罢了！**

拉普拉斯矩阵还有另外两个变种，本质上就是用度对周围邻居的影响做了不同的归一化策略，具体可参考wikipedia：

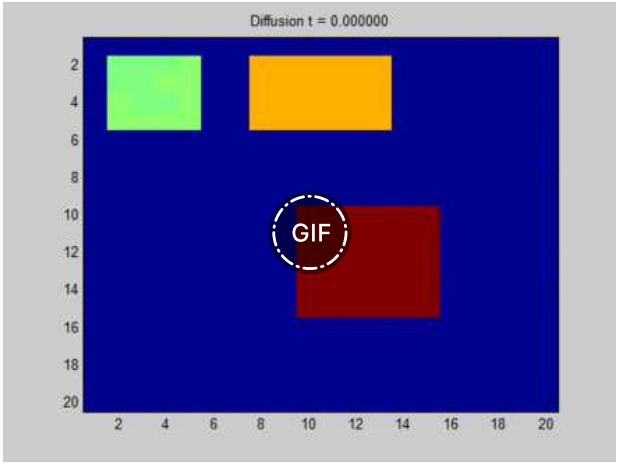
https://en.wikipedia.org/wiki/Laplacian_matrix
en.wikipedia.org

故事还没有结束，让我们来探讨点更有意思的事情～Laplacian matrix故事还没有结束，让我们来探讨点更有意思的事情～

宇宙热寂说、恒温热源和半监督学习

感谢你饶有兴致能够继续听我絮叨完。乍看这个标题，你一定认为我在扯淡，前面两个概念和后面一个八竿子打不着。不用着急，且听我慢慢道来，as the saying goes：因果有差距，联想来搭桥～

如果你仔细的思考，你会本文常挂在嘴边的热力学传导模型有个宿命论的结论：假设这个图是个连通图，每个结点到任一其他结点都有路可走，那么这样的事情肯定会发生：如果一个结点温度高于周围邻居，那么它的热量会持续流出到邻居，此间它自己的热量流失，温度就会降低，直到它和周围邻居一样，最终所有结点都达到一个平衡的、一样的温度。
我想说的是，你的猜想非常正确而且能够经过严格的数学证明，只要解上面这个方程（并不难）就能验证你的结论。
问题的本质在于，我们之前所讨论的热传导场景是一个**孤立系统的无源**场景，每一个结点的热量都有一个初始禀赋，没有外界干预它、给它热量，它每传给周围一些热量，自己就要减少，伴随自身温度的降低。那么根据**能量守恒定律**，整个系统的能量是定值，并且随着热传导，系统的热量被均匀分配到每个参与其中的结点。这正是**热力学第一定律**和**热力学第二定律**在这张图上的完美体现。这也是这张图上的“宇宙热寂说”，这个孤立的小宇宙，最终处处温度一样，平衡状态下没有热量的流动。
Wikipedia[1]对无源系统有个动态图仿真，能够增加直观认识，盗图放在这里：



无源系统最终出现“宇宙热寂”现象，所有点平衡态温度趋同，图片来自wikipedia [1]

如果我们在这张图上引入一个东西叫做**恒温热源**，事情就会发生改变。所谓恒温热源，就是这样一个存在，系统中有些点开了外挂，它们接着系统外面的某些“供热中心”，使得它们温度能够保持不变。这样，这些结点能够源源不断的去从比它温度高的邻居接受热量，向温度比它低的邻居释放热量，每当有热量的增补和损失，“供热中心”便抽走或者补足多余或者损失的热量，使得这个结点温度不发生变化。
那么最终的平衡的结果不是无源情况下所有结点温度一样，而是整个图中，高温热源不断供热给周围的邻居，热量传向远方，低温热源不断持续吸收邻居和远方的热量，最终状态下每条边热量稳恒流动（不一定为0），流量不再变化，每个结点都达到**终态的某个固有温度**。这个温度具体是多少，取决于整张图上面**恒温热源的温度和它们的分布**。
恒温热源的存在，正是我们跨向半监督学习的桥梁。
我们把问题向广告和推荐场景去类比，在这个场景下，某些结点有着明确的label，例如某个广告被点击，某个广告的ctr是多少，某个item被收藏，这些带着label的结点有着相对的确定性——它们可以被看作是这个结点的温度，这些**有标签的结点正是恒温热源**。那么，这些图的其他参与者，那些等待被我们预测的结点正是这张图的**无源部分**，它们被动的接受着那些或近或远的恒温热源传递来的Feature，改变着自己的Feature，从而影响自己的label。
让我们做个类比：

- **温度** 好比 **label** 是状态量
- **恒温热源 好比 有明确 label的结点**，它们把自己的Feature传递给邻居，让邻居的Feature与自己趋同从而让邻居有和自己相似的label
- 结点的**热能**就是**Feature**，无标签的、被预测的结点的Feature被有明确label的结点的Feature传递并且影响最终改变自己

需要说明的一点是，无论是有源还是无源，当有新的结点接入已经平衡的系统，系统的平衡被打破，新的结点最终接受已有结点和恒温热源的传递直到达到新的平衡。**所以我们可以不断的用现有的图去预测我们未见过的结点的性质，演化和滚大这个系统。**
如果关注有源情况下的热传导方程定量分析，可以看wikipedia这一条：
这里不是我们的重点，就不再赘述了。最后，我们讨论一下优化的过程，以期待发现什么。

Heat equation - Wikipedia
en.wikipedia.org

关于优化过程和优化结果的讨论Heat equation - Wikipedia关于优化过程和优化结果的讨论

从物理和从机器学习角度去看拉普拉斯热传导微分方程注定是不一样的。
从物理学角度去研究它，更多的是在搞明白**边界条件**和**初始条件**的情况下，探索和了解**后续状态的演化过程**。
而机器学习更多关心的是结果，毕竟我们要的是**最终**的**参数**和**特征**，用来使用和预测结果。然而，关注一些过程方面的东西，能够帮助我们更加深入的建立起原始热传导和前沿的GNN算法的联系。
我们并不靠解这个微分方程得到状态和时间的显式关系来获得对过程的认知，这个对我们当下讨论的话题意义不大，想了解怎么解可以看[1]，做法大概就是以所有特征向量为基展开状态向量，然后就能像解决简单常微分方程的办法积个分解出来，是一个exponential的形式。我们用另外一个方式来认知这个过程。
为求简单，我们研究上文所说的原始的无源热传导方程：

$$\frac{d\boldsymbol{\phi}}{dt} = -k\boldsymbol{L}\boldsymbol{\phi}$$

为了好解，我们把时间也离散化，微分变差分近似是这样：

$$\boldsymbol{\phi}_{t+1} - \boldsymbol{\phi}_t = -k\boldsymbol{L}\boldsymbol{\phi}_t$$

所以自然：

$$\boldsymbol{\phi}_{t+1} = \boldsymbol{\phi}_t - k\boldsymbol{L}\boldsymbol{\phi}_t$$

这实质上是一个机器学习，或者优化问题中常见的样子：

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha\nabla f(\boldsymbol{w}_t)$$

只是状态的变化量从梯度算子（可能还有动量、历史状态、Hessian矩阵、正则化操作等等，这里不赘述）作用于loss function变成了拉普拉斯算子作用于当前状态。

我们这回先不整体的考察状态向量 $\boldsymbol{\phi}$ 的演化，倒退到早些时候所说的每个结点 $\boldsymbol{i}$ 的状态 $\boldsymbol{\phi_i}$ 的变化，有：

$$\begin{aligned}
{\phi^{(i)}}_{t+1} &= {\phi^{(i)}}_t - k\sum_j A_{ij}({\phi^{(i)}}_t - {\phi^{(j)}}_t) \\
&= (1 - k \times deg(i)){\phi^{(i)}}_t - k\sum_j A_{ij}{\phi^{(j)}}_t \\
&= (1 - k \times deg(i)){\phi^{(i)}}_t - k\sum_{j \in Neighbor(i)} {\phi^{(j)}}_t
\end{aligned}$$

到这里已经很明了了，每个结点的下一个状态，是当前状态的常数倍（第一项），和所有邻居的加和（第二项）的融合。

我们来看看GraphSage核心算法的过程，会发现原理上二者的惊人相似性：

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \rightarrow 2^{\mathcal{V}}$

Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1...K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
  end
  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

GraphSage算法[3]，截自原论文

看最里面的循环节，就是迭代过程，核心两步：

1、Aggregate邻居结点的状态，作为邻居结点的一个“和”状态。

2、将这个“和”状态和当前结点的状态通过Concat操作和带非线性激活的全连接层融合到一起更新当前结点的状态。

原文中的配图很好的解释了这个过程：

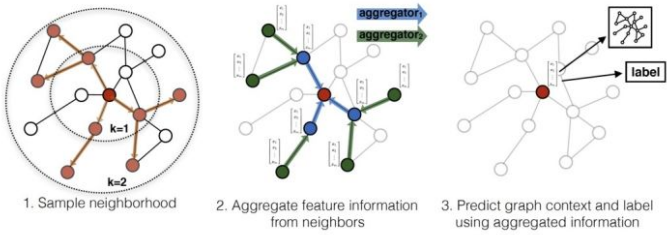


Figure 1: Visual illustration of the GraphSAGE sample and aggregate approach.

GraphSage算法[3]采样和Aggregate过程

GraphSage这个过程和原始的热传导过程是这么的相似，我们可以发现：

1、原始的热传导用所有直接相邻的邻居的简单加和作为邻居“和”状态，GraphSage用一个Aggregator作用于所有采样邻居（可以是二阶）得到“和”状态。本质是Aggregate，只是后者更加非线性，建模能力更强。

2、原始的热传导把邻居“和”状态和当前结点状态简单加权叠加作为融合态，GraphSage使用Concat操作加上神经网络非线性融合一把。本质是把邻居的“和”和本地的现有状态糅合得到本地新状态，依然只是后者更加非线性，建模能力更强。

3、原始热传导使用所有一阶邻居。GraphSage会采样邻居，因为实际上邻居可能很多全部算很慢，同时它还会采样高阶邻居。本质是主动融合几何上相近的单元。这里可能存在一个隐患，在高度非线性下，对周围邻居的采样的Aggregate是不是对周围所有邻居的Aggregate的**无偏估计**，本人知识有限，有待进一步考证。

所有的所有，还是**“相邻”、“叠加”与“融合”**三个关键词，只是解法更新、迭代、升级了。

理解了Laplacian变换和图上的热传播的关系，搞明白了Message Passing和有源无源状态下的均衡状态，再去看浩如烟海的paper和高票答案，相信你一定会有更深刻的理解！祝你好运～

参考文献

[1] Laplacian matrix
[2] Cannon J R. The one-dimensional heat equation[M]. Cambridge University Press, 1984.
[3] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs[C]//Advances in Neural Information Processing Systems. 2017: 1024-1034.
[4] Heat equation - Wikipedia

编辑于 2019-04-03

superbrother
清华大学 土木工程博士在读

5,232 人赞同了该回答

从CNN到GCN的联系与区别——GCN从入门到精（fang）通（qi）

1 什么是离散卷积？CNN中卷积发挥什么作用？

了解GCN之前必须对离散卷积（或者说CNN中的卷积）有一个明确的认识：

如何通俗易懂地解释卷积？这个链接的内容已经讲得很清楚了，**离散卷积本质就是一种加权求和。**

如图1所示，CNN中的卷积本质上就是利用一个共享参数的过滤器（kernel），**通过计算中心像素点以及相邻像素点的加权和来构成feature map实现空间特征的提取**，当然加权系数就是卷积核的权重系数。

那么卷积核的系数如何确定的呢？是随机化初值，然后根据误差函数通过反向传播梯度下降进行迭代优化。这是一个关键点，**卷积核的参数通过优化求出才能实现特征提取的作用，GCN的理论很大一部分工作就是为了引入可以优化的卷积参数。**

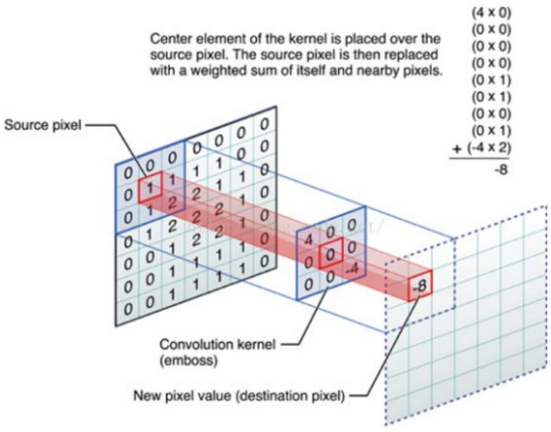


图1 CNN中卷积提取feature map示意图

注：这里的卷积是指深度学习（CNN）中的卷积，与数学中定义的卷积运算严格意义上是有区别的。两者的区别与联系可以见我的另一个回答。

哪位高手能解释一下卷积神经网络的卷积核？
www.zhihu.com

2 GCN中的Graph指什么？为什么要研究GCN？

CNN是Computer Vision里的大法宝，效果为什么好呢？原因在上面已经分析过了，可以很有效地提取空间特征。但是有一点需要注意：**CNN处理的图像或者视频数据中像素点（pixel）是排列成成很整齐的矩阵**（如图2所示，也就是很多论文中所提到的Euclidean Structure）。

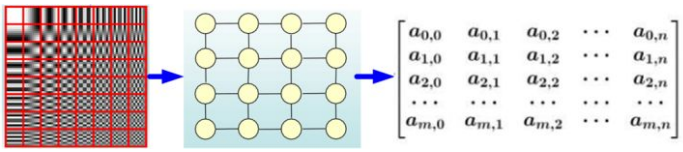


图2 图像矩阵示意图（Euclidean Structure）

与之相对应，科学研究中还有很多Non Euclidean Structure的数据，如图3所示。社交网络、信息网络中有很多类似的结构。

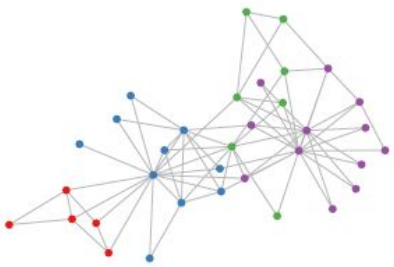


图3 社交网络拓扑示意（Non Euclidean Structure）

实际上，这样的网络结构（Non Euclidean Structure）就是图论中抽象意义上的拓扑图。

所以，**Graph Convolutional Network中的Graph是指数学（图论）中的用顶点和边建立相应关系的拓扑图。**

那么为什么要研究GCN？原因有三：

(1)CNN无法处理Non Euclidean Structure的数据，学术上的表达是传统的离散卷积（如问题1中所述）**在Non Euclidean Structure的数据上无法保持平移不变性。**通俗理解就是在拓扑图中每个顶点的相邻顶点数目都可能不同，那么当然无法用一个同样尺寸的卷积核来进行卷积运算。
(2)由于CNN无法处理Non Euclidean Structure的数据，又**希望在这样的数据结构（拓扑图）上有效地提取空间特征来进行机器学习**，所以GCN成为了研究的重点。
(3)读到这里大家可能会想，自己的研究问题中没有拓扑结构的网络，那是不是根本就不会用到GCN呢？其实不然，**广义上来讲任何数据在赋范空间内都可以建立拓扑关联，谱聚类就是应用了这样的思想**（谱聚类（spectral clustering）原理总结）。**所以说拓扑连接是一种广义的数据结构，GCN有很大的应用空间。**
综上所述，GCN是要为除CV、NLP之外的任务提供一种处理、研究的模型。

3 提取拓扑图空间特征的两种方式

GCN的本质目的就是用来提取拓扑图的空间特征，那么实现这个目标只有graph convolution这一种途径吗？当然不是，在vertex domain(spatial domain)和spectral domain实现目标是两种最主流的方式。
(1)vertex domain(spatial domain)是非常直观的一种方式。顾名思义：提取拓扑图上的空间特征，那么就把每个顶点相邻的neighbors找出来。这里面蕴含的科学问题有二：
a.按照什么条件去找中心vertex的neighbors，也就是如何确定receptive field?
b.确定receptive field，按照什么方式处理包含不同数目neighbors的特征?
根据a,b两个问题设计算法，就可以实现目标了。推荐阅读这篇文章Learning Convolutional Neural Networks for Graphs（图4是其中一张图片，可以看出大致的思路）。

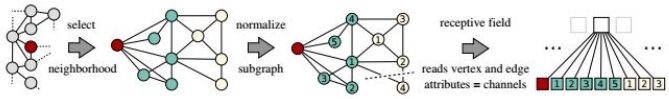


图4 vertex domain提取空间特征示意

这种方法主要的缺点如下：
c.每个顶点提取出来的neighbors不同，使得计算处理必须针对每个顶点
d.提取特征的效果可能没有卷积好
当然，对这个思路喜欢的读者可以继续搜索相关文献，学术的魅力在于百家争鸣嘛！
(2)spectral domain就是GCN的理论基础了。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。从整个研究的时间进程来看：首先研究GSP（graph signal processing）的学者定义了graph上的Fourier Transformation，进而定义了graph上的convolution，最后与深度学习结合提出了Graph Convolutional Network。
认真读到这里，脑海中应该会浮现出一系列问题：
Q1 什么是Spectral graph theory?
Spectral graph theory请参考这个，简单的概括就是**借助于图的拉普拉斯矩阵的特征值和特征向量来研究图的性质**
Q2 GCN为什么要利用Spectral graph theory?
这应该是看论文过程中读不懂的核心问题了，要理解这个问题需要大量的数学定义及推导，没有一定的数学功底难以驾驭（我也才疏学浅，很难回答好这个问题）。
所以，先绕过这个问题，来看Spectral graph实现了什么，再进行探究为什么?

4 什么是拉普拉斯矩阵？为什么GCN要用拉普拉斯矩阵?

Graph Fourier Transformation及Graph Convolution的定义都用到图的拉普拉斯矩阵，那么首先来介绍一下拉普拉斯矩阵。
对于图 $G=(V,E)$，其Laplacian 矩阵的定义为 $L=D-A$，其中 L 是Laplacian 矩阵， D 是顶点的度矩阵（对角矩阵），对角线上元素依次为各个顶点的度， A 是图的邻接矩阵。看图5的示例，就能很快知道Laplacian 矩阵的计算方法。

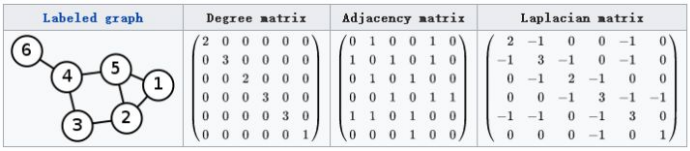


图5 Laplacian 矩阵的计算方法

这里要说明的是：**常用的拉普拉斯矩阵实际有三种**：
No.1 $L=D-A$ 定义的Laplacian 矩阵更专业的名称叫**Combinatorial Laplacian**
No.2 $L^{sys}=D^{-1/2}LD^{-1/2}$ 定义的叫**Symmetric normalized Laplacian**，很多GCN的论文中应用的是这种拉普拉斯矩阵
No.3 $L^{rw}=D^{-1}L$ 定义的叫**Random walk normalized Laplacian**,有读者的留言说看到了**Graph Convolution与Diffusion相似之处**，当然从Random walk normalized Laplacian就能看出了两者确有相似之处（**其实两者只差一个相似矩阵的变换**，可以参考Diffusion-Convolutional Neural Networks，以及下图）
不需要相关内容的读者可以略过此部分

5.2 RELATION WITH SPECTRAL GRAPH CONVOLUTION

Proof. The spectral graph convolution utilizes the concept of normalized graph Laplacian $\boldsymbol{L} = \boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{D}-\boldsymbol{W})\boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^\intercal$. ChebNet parametrizes f_θ to be a K order polynomial of $\boldsymbol{\Lambda}$, and calculates it using stable Chebyshev polynomial basis.

$$\boldsymbol{X}_{:,t} \star_G f_{\boldsymbol{\theta}} = \boldsymbol{\Phi}\left(\sum_{k=0}^{K-1}\theta_k\boldsymbol{\Lambda}^k\right)\boldsymbol{\Phi}^\intercal\boldsymbol{X}_{:,t} = \sum_{k=0}^{K-1}\theta_k\boldsymbol{L}^k\boldsymbol{X}_{:,t} = \sum_{k=0}^{K-1}\tilde{\theta}_k T_k(\tilde{\boldsymbol{L}})\boldsymbol{X}_{:,t} \tag{4}$$

where $T_0(x)=1, T_1(x)=x, T_k(x)=xT_{k-1}(x)-T_{k-2}(x)$ are the basis of the Cheyshev polynomial. Let λ_{max} denote the largest eigenvalue of $\boldsymbol{L}$, and $\tilde{\boldsymbol{L}} = \frac{2}{\lambda_{max}}\boldsymbol{L} - \boldsymbol{I}$ represents a rescaling of the graph Laplacian that maps the eigenvalues from $[0, \lambda_{max}]$ to $[-1, 1]$ since Chebyshev polynomial forms an orthogonal basis in $[-1, 1]$. Equation 4 can be considered as a polynomial of $\tilde{\boldsymbol{L}}$ and we will show that the output of ChebNet Convolution is *similar* to the output of diffusion convolution up to constant scaling factor. Assume $\lambda_{max} = 2$ and $\boldsymbol{D}_I = \boldsymbol{D}_O = \boldsymbol{D}$ for undirected graph.

$$\tilde{\boldsymbol{L}} = \boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{D}-\boldsymbol{W})\boldsymbol{D}^{-\frac{1}{2}} - \boldsymbol{I} = -\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}} \sim -\boldsymbol{D}^{-1}\boldsymbol{W} \tag{5}$$

$\tilde{\boldsymbol{L}}$ is *similar* to the negative random walk transition matrix, thus the output of Equation 4 is also similar to the output of Equation 2 up to constant scaling factor. □

Diffusion Graph Convolution与Spectral Graph Convolution相似性证明

其实维基本科对Laplacian matrix的定义上写得很清楚，**国内的一些介绍中只有第一种定义。**这让我在最初看文献的过程中感到一些的困惑，特意写下来，帮助大家避免再遇到类似的问题。

为什么GCN要用拉普拉斯矩阵?

拉普拉斯矩阵矩阵有很多良好的性质，这里写三点我感触到的和GCN有关之处

(1)拉普拉斯矩阵是对称矩阵，可以进行特征分解（谱分解），这就和GCN的spectral domain对应上了

(2)拉普拉斯矩阵只在中心顶点和一阶相连的顶点上（1-hop neighbor）有非0元素，其余之处均为0

（3）通过拉普拉斯算子与拉普拉斯矩阵进行类比（详见第6节）

两者的关系可以参考我的另一个文章：

superbrother：拉普拉斯矩阵与拉普拉斯算子的关系

zhuanlan.zhihu.com

5 拉普拉斯矩阵的谱分解（特征分解）

GCN的核心基于拉普拉斯矩阵的谱分解，文献中对于这部分内容没有讲解太多，初学者可能会遇到不少误区，所以先了解一下特征分解。

矩阵的谱分解，特征分解，对角化都是同一个概念（特征分解_百度百科）。

不是所有的矩阵都可以特征分解，其充要条件为n阶方阵存在n个线性无关的特征向量。

但是拉普拉斯矩阵是半正定对称矩阵（半正定矩阵本身就是对称矩阵，半正定矩阵_百度百科，此处这样写为了和下面的性质对应，避免混淆），有如下三个性质：

- **对称矩阵一定n个线性无关的特征向量**
- **半正定矩阵的特征值一定非负**
- **对阵矩阵的特征向量相互正交，即所有特征向量构成的矩阵为正交矩阵。**

由上可以知道拉普拉斯矩阵一定可以谱分解，且分解后有特殊的形式。

对于拉普拉斯矩阵其谱分解为：

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

其中 $\boldsymbol{U} = (\vec{u_1}, \vec{u_2}, \cdots, \vec{u_n})$ ，是**列向量**为单位特征向量的矩阵，**也就说** $\vec{u_l}$ **是列向量。**

$\begin{pmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_n\end{pmatrix}$ 是n个特征值构成的对角阵。

由于 $\boldsymbol{U}$ 是正交矩阵，即 $\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{E}$

所以特征分解又可以写成：

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

文献中都是最后导出的这个公式，但大家不要误解，特征分解最右边的是特征矩阵的逆，只是拉普拉斯矩阵的性质才可以写成特征矩阵的转置。

其实从上可以看出：整个推导用到了很多数学的性质，在这里写得详细一些，避免大家形成错误的理解。

6 如何从传统的傅里叶变换、卷积类比到Graph上的傅里叶变换及卷积?

把传统的傅里叶变换以及卷积迁移到Graph上来，核心工作其实就是把拉普拉斯算子的特征函数 $e^{-i\omega t}$ **变为Graph对应的拉普拉斯矩阵的特征向量。**

(1)推广傅里叶变换

想亲自躬行的读者可以阅读The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains这篇论文，下面是我的理解与提炼：

(a)Graph上的傅里叶变换

传统的傅里叶变换定义为：

$$F(\omega) = \mathcal{F}[f(t)] = \int f(t) e^{-i\omega t} dt$$

信号 $f(t)$ 与基函数 $e^{-i\omega t}$ 的积分，**那么为什么要找 $e^{-i\omega t}$ 作为基函数呢？从数学上看， $e^{-i\omega t}$ 是拉普拉斯算子的特征函数（满足特征方程），ω 就和特征值有关。**

广义的特征方程定义为：

$$AV = \lambda V$$

其中 A 是一种变换， V 是特征向量或者特征函数（无穷维的向量）， λ 是特征值。

$e^{-i\omega t}$ 满足：

$$\Delta e^{-i\omega t} = \frac{\partial^2}{\partial t^2} e^{-i\omega t} = -\omega^2 e^{-i\omega t}$$

当然 $e^{-i\omega t}$ 就是变换 Δ 的特征函数， ω 和特征值密切相关。

那么，可以联想了，**处理Graph问题的时候，用到拉普拉斯矩阵**（拉普拉斯矩阵就是离散拉普拉斯算子，想了解更多可以参考Discrete Laplace operator），**自然就去找拉普拉斯矩阵的特征向量了。**

L 是拉普拉斯矩阵， V 是其特征向量，自然满足下式

$$LV = \lambda V$$

离散积分就是一种内积形式，仿上定义Graph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Graph上的 N 维向量， $f(i)$ 与Graph的顶点一一对应， $u_l(i)$ 表示第 l 个特征向量的第 i 个分量。那么特征值（频率） λ_l 下的， f 的Graph 傅里叶变换就是与 λ_l 对应的特征向量 u_l 进行内积运算。

注：上述的内积运算是在复数空间中定义的，**所以采用了 $u_l^*(i)$ ，也就是特征向量 u_l 的共轭。**Inner product更多可以参考Inner product space。

利用矩阵乘法将Graph上的傅里叶变换推广到矩阵形式：

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

即 f 在Graph上傅里叶变换的矩阵形式为： $\hat{f} = U^T f \qquad (a)$

式中： U^T 的定义与第五节中的相同

(b)Graph上的傅里叶逆变换

类似地，**传统的傅里叶逆变换是对频率 ω 求积分：**

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

迁移到Graph上变为对特征值 λ_l 求和：

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

利用矩阵乘法将Graph上的傅里叶逆变换推广到矩阵形式：

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

即 f 在Graph上傅里叶逆变换的矩阵形式为： $f = U\hat{f} \qquad (b)$

式中： U 的定义与第五节中的相同

(2)推广卷积

在上面的基础上，利用**卷积定理**类比来将卷积运算，推广到Graph上。

卷积定理：函数卷积的傅里叶变换是函数傅立叶变换的乘积，即对于函数 $f(t)$ 与 $h(t)$ 两者的卷积是其函数傅立叶变换乘积的逆变换：

$$f * h = \mathcal{F}^{-1}\left[\hat{f}(\omega)\hat{h}(\omega)\right] = \frac{1}{2\Pi} \int \hat{f}(\omega)\hat{h}(\omega) e^{i\omega t} d\omega$$

类比到Graph上并把傅里叶变换的定义带入， f 与卷积核 h 在Graph上的卷积可按下列步骤求出：

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

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

两者的傅立叶变换乘积即为：$\begin{pmatrix} \hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{pmatrix} U^T f$

再乘以 U 求两者傅立叶变换乘积的逆变换，则求出卷积：

$$(f * h)_G = U \begin{pmatrix} \hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{pmatrix} U^T f \qquad (1)$$

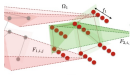
式中： U 及 U^T 的定义与第五节中的相同

注：很多论文中的Graph卷积公式为：

$$(f * h)_G = U((U^T h) \odot (U^T f)) \qquad (2)$$

$\odot$ 表示Hadamard product（哈达马积），对于两个维度相同的向量、矩阵、张量进行对应位置的逐元素乘积运算。

其实式(2)与式(1)是完全等价的，详细的证明见我的另一篇文章。



这里主要推(1)式是为了和后面的deep learning相结合。

这部分理论也是我看了很久才想明白的，在此记录下来，如果是想继续探究理论的朋友，可以作为“入门小引”，毕竟理论还很深！对于喜欢实践的朋友，也能初步了解理论基础，也是“开卷有益”。

7 为什么拉普拉斯矩阵的特征向量可以作为傅里叶变换的基？特征值表示频率？

(1)为什么拉普拉斯矩阵的特征向量可以作为傅里叶变换的基？

傅里叶变换一个本质理解就是：**把任意一个函数表示成了若干个正交函数（由sin,cos 构成）的线性组合。**

图6 傅里叶逆变换图示

通过第六节中(b)式也能看出，**graph傅里叶变换也把graph上定义的任意向量 f ，表示成了拉普拉斯矩阵特征向量的线性组合，即：**

$$f = \hat{f}(\lambda_1)u_1 + \hat{f}(\lambda_2)u_2 + \cdots + \hat{f}(\lambda_n)u_n$$

那么：**为什么graph上任意的向量 f 都可以表示成这样的线性组合？**

原因在于 $(\overrightarrow{u_1}, \overrightarrow{u_2}, \cdots, \overrightarrow{u_n})$ 是graph上 n 维空间中的 n 个线性无关的正交向量（原因参看第五节中拉普拉斯矩阵的性质），由线性代数的知识可以知道：**n 维空间中 n 个线性无关的向量可以构成空间的一组基，而且拉普拉斯矩阵的特征向量还是一组正交基。**

(2)怎么理解拉普拉斯矩阵的特征值表示频率？

在**graph空间上无法可视化展示“频率”**这个概念，那么从**特征方程**上来抽象理解。

将拉普拉斯矩阵 L 的 n 个**非负实特征值，从小到大排列为** $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ ，而且最小的特征值 $\lambda_1 = 0$ ，因为 n 维的全1向量对应的特征值为0（由 L 的定义就可以得出）：

$$L \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = 0$$

从特征方程的数学理解来看：

$$Lu = \lambda u$$

在由Graph确定的 n 维空间中，越小的特征值 λ_l 表明：拉普拉斯矩阵 L 其所对应的基 u_l 上的分量、“信息”越少，那么当然就是可以忽略的低频部分了。

其实**图像压缩**就是这个原理，**把像素矩阵特征分解后，把小的特征值（低频部分）全部变成0，PCA降维**也是同样的，**把协方差矩阵特征分解后，按从大到小取出前K个特征值对应的特征向量作为新的“坐标轴”。**

Graph Convolution的理论告一段落了，下面开始Graph Convolution Network

8 Deep Learning中的Graph Convolution

Deep learning 中的Graph Convolution直接看上去会和第6节推导出的图卷积公式有很大的不同，但是万变不离其宗，(1)式是推导的本源。

第1节的内容已经解释得很清楚：Deep learning 中的Convolution就是要设计含有trainable共享参数的kernel，从(1)式看很直观：graph convolution中的卷积参数就是 $diag(\hat{h}(\lambda_l))$ 。

- **第一代GCN**

Spectral Networks and Locally Connected Networks on Graphs中**简单粗暴地把** $diag(\hat{h}(\lambda_l))$ **变成了卷积核** $diag(\theta_l)$，也就是：

$$y_{output} = \sigma\left(Ug_\theta(\Lambda)U^T x\right) \qquad (3)$$

（为避免混淆，本文中称 $g_\theta(\Lambda)$ 是卷积核， $Ug_\theta(\Lambda)U^T$ 的运算结果为卷积运算矩阵）

$$g_\theta(\Lambda) = \begin{pmatrix} \theta_1 & & \\ & \ddots & \\ & & \theta_n \end{pmatrix}$$

式（3）就是标准的第一代GCN中的layer了，其中 $\sigma(\cdot)$ **是激活函数，** $\Theta = (\theta_1, \theta_2, \cdots, \theta_n)$ **就跟三层神经网络中的weight一样是任意的参数，通过初始化赋值然后利用误差反向传播进行调整，** x **就是graph上对应于每个顶点的feature vector（由特数据集提取特征构成的向量）。**

第一代的参数方法存在着一些弊端：主要在于：

（1）每一次前向传播，都要计算 $U, diag(\theta_l)$ **及** U^T **三者的矩阵乘积，特别是对于大规模的graph，计算的代价较高，也就是论文中** $\mathcal{O}(n^2)$ **的计算复杂度**

（2）卷积核不具有spatial localization（这个在第9节中进一步阐述）

（3）卷积核需要 n **个参数**

由于以上的缺点第二代的卷积核设计应运而生。

- **第二代GCN**

(Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering)，把 $\hat{h}(\lambda_l)$ **巧妙地设计成了** $\sum_{j=0}^{K} \alpha_j \lambda_l^j$ ，也就是：

$$y_{output} = \sigma\left(Ug_\theta(\Lambda)U^T x\right) \qquad (4)$$

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

上面的公式仿佛还什么都看不出来，下面利用矩阵乘法进行变换，来一探究竟。

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

进而可以导出：

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

上式成立是因为 $L^2 = U\Lambda U^T U\Lambda U^T = U\Lambda^2 U^T$ 且 $U^T U = E$

各符号的定义都同第五节。

(4)式就变成了：

$$y_{output} = \sigma\left(\sum_{j=0}^{K-1} \alpha_j L^j x\right) \qquad (5)$$

其中 $(\alpha_0, \alpha_1, \cdots, \alpha_{K-1})$ **是任意的参数，通过初始化赋值然后利用误差反向传播进行调整。**

式(5)所设计的卷积核其优点在于在于：

（1）卷积核只有 K **个参数，一般** K **远小于** n**，参数的复杂度被大大降低了。**

（2）矩阵变换后，神奇地发现不需要做特征分解了，直接用拉普拉斯矩阵 L **进行变换。然而由于要计算** L^j**，计算复杂度还是** $\mathcal{O}(n^2)$

（3）卷积核具有很好的spatial localization，特别地， K **就是卷积核的receptive field，也就是说每次卷积会将中心顶点K-hop neighbor上的feature进行加权求和，权系数就是** α_k

更直观地看， ***K*=1就是对每个顶点上一阶neighbor的feature进行加权求和，如下图所示：**

图7 k=1 的graph convolution示意

同理，***K*=2的情形如下图所示：**

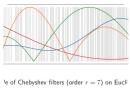
图8 k=2 的graph convolution示意

注：上图只是以一个顶点作为实例，GCN每一次卷积对所有的顶点都完成了图示的操作。

- **利用Chebyshev多项式作为卷积核**

在GCN领域中，利用Chebyshev多项式作为卷积核是非常通用的形式。针对这部分内容，我专门整理了如下的文章。

superbrother：Chebyshev多项式作为GCN卷积核
zhuanlan.zhihu.com



上述的内容已经阐明了GCN的整体思路，其他的一些形式都是“万变不离其宗”。例如Semi-Supervised Classification with Graph Convolutional Networks一文中的GCN形式，其实是二阶Chebyshev多项式推导出的特例。

9 在GCN中的Local Connectivity和Parameter Sharing

CNN中有两大核心思想：网络局部连接，卷积核参数共享。这两点内容的详细理解可以看我的这个回答。

如何理解卷积神经网络中的权值共享？
www.zhihu.com

那么我们不禁会联想：这两点在GCN中是怎样的呢？以下图的graph结构为例来探究一下

graph结构示意

- **GCN中的Local Connectivity**

(a)如果利用第一代GCN，根据式（3）卷积运算矩阵（ $Ug_\theta(\Lambda)U^T$ ） 即为

第一代卷积核示意（需要放大观看）

这个时候，可以发现这个卷积核没有**local的性质，因为该卷积核得到的运算矩阵在所有位置上都有非0元素。以第一个顶点为例，如果考虑一阶local关系的话，那么卷积核中第一行应该只有[1,1],[1,2],[1,5]这三个位置的元素非0。换**句话说，这是一个global全连接的卷积核。
(b)如果是第二代GCN，根据式（5）当 $K=0$ 卷积运算矩阵即为

$$\begin{bmatrix} \alpha_0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_0 \end{bmatrix}$$

当 $K=1$ 卷积运算矩阵即为

$$\begin{bmatrix} \alpha_0+2\alpha_1 & -\alpha_1 & 0 & 0 & -\alpha_1 & 0 \\ -\alpha_1 & \alpha_0+3\alpha_1 & -\alpha_1 & 0 & -\alpha_1 & 0 \\ 0 & -\alpha_1 & \alpha_0+2\alpha_1 & -\alpha_1 & 0 & 0 \\ 0 & 0 & -\alpha_1 & \alpha_0+3\alpha_1 & -\alpha_1 & -\alpha_1 \\ -\alpha_1 & -\alpha_1 & 0 & -\alpha_1 & \alpha_0+3\alpha_1 & 0 \\ 0 & 0 & 0 & -\alpha_1 & 0 & \alpha_0+\alpha_1 \end{bmatrix}$$

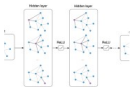
当 $K=3$ 卷积运算矩阵即为

$$\begin{bmatrix} \alpha_0+2\alpha_1+6\alpha_2 & -\alpha_1-4\alpha_2 & \alpha_2 & \alpha_2 & -\alpha_1-4\alpha_2 & 0 \\ -\alpha_1-4\alpha_2 & \alpha_0+3\alpha_1+12\alpha_2 & -\alpha_1-5\alpha_2 & 2\alpha_2 & -\alpha_1-5\alpha_2 & 0 \\ \alpha_2 & -\alpha_1-5\alpha_2 & \alpha_0+2\alpha_1+6\alpha_2 & -\alpha_1-5\alpha_2 & 2\alpha_2 & \alpha_2 \\ \alpha_2 & 2\alpha_2 & -\alpha_1-5\alpha_2 & \alpha_0+3\alpha_1+12\alpha_2 & -\alpha_1-6\alpha_2 & -\alpha_1-4\alpha_2 \\ -\alpha_1-4\alpha_2 & -\alpha_1-5\alpha_2 & 2\alpha_2 & -\alpha_1-6\alpha_2 & \alpha_0+3\alpha_1+12\alpha_2 & \alpha_2 \\ 0 & 0 & \alpha_2 & -\alpha_1-4\alpha_2 & \alpha_2 & \alpha_0+\alpha_1+2\alpha_2 \end{bmatrix}$$

看一下图的邻接结构，卷积运算矩阵的非0元素都在localize的位置上。

- **GCN中的Parameter Sharing**

相关内容比较多，我专门写了一篇文章，感兴趣的朋友可以阅读一下。



10 GCN的可解释性

前面的内容，已经介绍了GCN的基本原理以及一些特性的理解。**这章节的内容是我个人的部分研究工作，将GCN应用于大规模交通路网速度预测问题中，对空间相关性的建模结果进行解释。**感兴趣的朋友可以阅读一下我们的论文。

如果这个回答的内容对于大家的研究工作有帮助，也希望引用我们的论文。

Zhang, Z., Li, M., Lin, X., Wang, Y., & He, F. (2019). Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies. *Transportation Research Part C: Emerging Technologies*, *105*, 297-322.

11 关于有向图问题

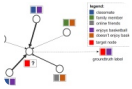
前面的理论推导都是**关于无向图**，如果是**有向图问题，最大的区别就是邻接矩阵会变成非对称矩阵**，这个时候不能直接定义**拉普利矩阵及其谱分解（拉普拉斯矩阵本身是定义在无向图上的）。这个时候有两条思路解决问题：**

（a）要想保持理论上的完美，就需要重新定义图的邻接关系，保持对称性

比如MotifNet: a motif-based Graph Convolutional Network for directed graphs 提出利用**Graph Motifs定义图的邻接关系。**

（b）如果只是为了应用，有其他形式的GCN或者GAT可以处理有向图

简而言之，要想简单地处理有向图问题，那就换成一种逐顶点（node-wise）的运算方式，可以参考下面这篇文章中的3.2及3.3节。

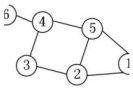


值得说明的是：GAT作者写道“It is worth noting that, as Kipf & Welling (2017) and Atwood & Towsley (2016), our work can also be reformulated as a particular instance of MoNet (Monti et al., 2016). ”

也就是说本质上这些模型都可以认为是在重新定义了图的邻接关系后，再进行基本的卷积运算。

12 GCN的过度平滑问题

公式（5）给出的卷积核中，含有拉普拉斯矩阵的 j 次幂。**当 j 的取值较大时，GCN学习到的特征可能会存在过度平滑现象，即每个顶点的输出特征都十分相似。这个问题的原因及解决方法可以参考我下面的回答。**



13 GCN的相关资料

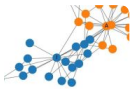
————————GCN的综述与论文合辑————————

清华大学孙茂松组：图神经网络必读论文列表
mp.weixin.qq.com

thunlp/GNNPapers
github.com

————————动手教程————————

图卷积网络到底怎么做，这是一份极简的Numpy实现
mp.weixin.qq.com



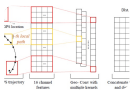
————————华丽的分割线————————

还写了一些关于机器学习比较热点的文章，大家有需要可以看看，一起学习进步！

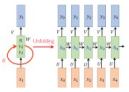
graph convolution network有什么比较好的应用task?
www.zhihu.com

superbrother：xgboost如何使用MAE或MAPE作为目标函数?
zhuanlan.zhihu.com

CNN模型可以输入离散特征吗?
www.zhihu.com



LSTM如何来避免梯度弥散和梯度爆炸?
www.zhihu.com



编辑于 2020-03-30

知乎用户
f(f(f...f(喵~))) = 萌

265 人赞同了该回答

实际上只是为了做半监督的话，不是做图生成的话，图卷积思路非常简单：

我有 N 个样本有标签，M 个样本没有标签，这 N+M 的样本之间有各种连接关系，用 adj 矩阵描述为 A。我需要通过学习 N 个有标签的样本，推断出 M 个没标签样本的标签。
和传统标签传播思路不一样的是，图卷积本质不是传播标签，而是在传播特征，图卷积将不知道标签的特征，传染到已知标签的特征节点上，利用已知标签节点的分类器旁敲侧击其属性。具体来说：
给定一个有标签节点 i，设其特征为 f_i，搜索其最近邻的节点，$f_j \quad j \in S_i$，紧邻节点有可能有标签，也有可能没有，不过不用理，反正通常它的特征是知道的，我们定义混合特征：
$f_i^* = \sum_{j \in S} (l_{ii} w_{ii} f_i + l_{ij} w_{ij} f_j)$，其中 w 为训练参数，l 为图的拉普拉斯量，其实就是一个衰减因子，你紧邻节点的无关程度越高，特征融合过程自然要减少考虑其贡献，风牛马不相及嘛~ 然后将做好的混合特征送到分类器当中 train，进行传统的分类任务。train 好以后，inference 的过程就用不知道标签的节点上的混合特征，送分类器，然后预测。当然，我们可以多垒几层，就变成了二度特征传染，三度特征传染。。

我说点不一样的观点，反正我一直怀疑图卷积是否有必要严格遵循卷积的数学定义，一直怀疑。实际上，我们传统图像的卷积都不是严格数学意义上的卷积。一开始，图卷积确实要满足卷积的数学性质而搞了一堆复杂的谱展开。其实，后期被 max welling 等人简化成一阶展开，用于半监督学习，每层图卷积网络刚好对应为近邻匹配，效果居然比做复杂的谱展开要好，这个事实不得不令人怀疑复杂的数学要求是否存在画蛇添足。

编辑于 2018-03-31

Alva
帝国理工学院 助理研究员

368 人赞同了该回答

最近入坑Geometric Deep Learning，相对于直接回答这个问题，我觉得给一个全面的介绍会更有帮助，上面的答主无非是以下几篇tutorials的翻译，对于初学者更应该看源材料。我整理了一些不错的论文和tutorials，对于想要迅速进入这个领域，看以下这些资料足矣。如果有什么想法，欢迎跟我讨论。

Tutorials:

1. Geometric Deep Learning on Graphs and Manifolds

NIPS 2017
nips.cc

全面的介绍了目前用DL处理Non-Euclidean doman, i.e. manifolds, graphs三类主要的方法，可以用很精简的话概括：
Specral methods for fixed graphs：最初的核心思想是，通过类比的方法定义Non-Euclidean卷积及与频域的关系，先做傅里叶变换在频域相乘，再做逆变换。经过几年的发展，现在已经不需要在做傅里叶变换，只需要用到graph laplacian。而不同框架核心的差别在与convolutional layer里filter的设计，旨在维持stationarity, O(1) complexity in each layer, 消除O(n^2) Fourier basis的计算。最新的需求在于解决fourier basis domain-dependent, in another word, filter are isotropic.
Spatial methods for variable graphs: 核心思想是，将相邻vertex看作集合，通过取平均值去刻画相邻点的关系。(**目前这种方法仍然是绝对主流80-90%**)
spatial methods for manifolds：核心思想是从geodesic polar coordinates的角度去设计空域的卷积。
Tutorials里也介绍了各种相应的应用。

2. IPAM Workshop on New Deep Learning Techniques
Convolutional Neural Networks on Graphs (by Xavier Bresson): 以Spectral的方法为主，介绍了这个方向的发展，以及目前在处理的问题，即怎样使spectral CNN不再受限于domain-dependent Fourier basis。

Convolutional Neural Networks on Graphs
www.ipam.ucla.edu

Deep Geometric Matrix Completion : a Geometric Deep Learning approach to Recommender Systems (by Federico Monti): 介绍目前性能最优秀的俩类架构。**CayleyNet**, 把ChabyNet的Filter设计的更为复杂了一些，介绍了一个相关的应用处理citation network；**MoNet framework** 基于Geodesic polar coordinates构造了一种spatial convolution，其中patch operator是gaussian mixture model.，介绍了相关的应用处理3D shape correspondence.

Deep Geometric Matrix Completion: a Geometric Deep Learning approach t...
www.ipam.ucla.edu

Deep functional maps: intrinsic structured prediction for dense shape correspondence (by **Michael Bronstein**) : 主要介绍目前spatial methods的发展以及最新进展状况。

Deep functional maps: intrinsic structured prediction for dense shap...
www.ipam.ucla.edu

总之，对于想要迅速入坑这个领域的，建议优先看上述tutorials；下面几个相关的tutorials也比较有意思，介绍如何处理community detection, graph embedding, multi-agent communication.

On Computational Hardness with Graph Neural Networks
www.ipam.ucla.edu

Large-scale Graph Representation Learning
www.ipam.ucla.edu

Deep Architecture for Sets and Its Application to Multi-agent...

www.ipam.ucla.edu

Papers:
我主要按三条线整理paper以及相关的应用（根据时间先后顺序），1. Spectral methods for fixed graphs （not limited to fixed graphs for the most state of the art techniques to some extent in 2018）; 2. Spatial methods for arbitrary graphs (many interdiscipline applications) ; 3. Spatial methods for manifolds (charting-based)。 如果没有时间的话，直接关注加粗的工作。

1. Spectral methods for fixed graphs

1.1 Papers

- J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral Networks and Deep Locally Connected Networks on Graphs, ICLR 2014 (Vanilla Spectral graph ConvNets, **the first spectral CNN**)
- M. Henaff, J. Bruna, Y. LeCun, Deep Convolutional Networks on Graph-Structured Data, 2015 (SplineNets)
- M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering, NIPS 2016 (**ChebyNet**)
- T. Kipf, M. Welling, Semi-supervised Classification with Graph Convolutional Networks, ICLR 2017 (**Simplification of ChebyNet**)
- (Similar work) J. Atwood and D. Towsley. Diffusion-convolutional neural networks. NIPS, 2016
- R. Levie*, F. Monti*, X. Bresson, M. M. Bronstein, CayleyNets: Graph convolutional neural networks with complex rational spectral filters, 2017 (**CayleyNets**)
- F. Monti, X. Bresson, M. M. Bronstein, Geometric matrix completion with recurrent **multi-graph neural networks**, NIPS 2017 (combining a Multi-Graph CNN (MGCNN) and a recurrent neural network (RNN))

1.2 Applications
Citation networks:
M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering, NIPS 2016
T. Kipf, M. Welling, Semi-supervised Classification with Graph Convolutional Networks, ICLR 2017
Recommender systems (matrix completion):
F. Monti, X. Bresson, M. M. Bronstein, Geometric matrix completion with recurrent multi-graph neural networks, NIPS 2017 (combining a Multi-Graph CNN (MGCNN) and a recurrent neural network (RNN)) (sRGCNN (Cheb))
R. Levie*, F. Monti*, X. Bresson, M. M. Bronstein, CayleyNets: Graph convolutional neural networks with complex rational spectral filters, 2017 (CayleyNet framework) (sRGCNN (Cayley))
Medical Imaging:
Sofia Ira Ktena, Sarah Parisot, Enzo Ferrante, Martin Rajchl, Matthew Lee, Ben Glocker, Daniel Rueckert, Distance Metric Learning using Graph Convolutional Networks: Application to Functional Brain Networks, 2017
1.3 Key issues
Poor generalization across different domains: Spectral kernels are isotropic; Only undirected graphs.
Papers those try to solve these problems:
L. Yi, H. Su, X. Guo, L. Guibas, **SyncSpecCNN: Synchronized Spectral CNN for 3D Shape Segmentation** , CVPR 2017
Xavier Bresson, Thomas Laurent, Residual Gated Graph ConvNets, arXiv:1711.07553, 2017 [**edge gates**]
P. Velickovi ˇ c, G. Cucurull, A. Casanova, A. Romero, ´ P. Lio, and Y. Bengio. Graph attention networks. In ICLR, 2018 [**attention**]
J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, G. E. Dahl, Neural Message Passing for Quantum Chemistry, ICML 2017 [**Differentiate graph edges and graph vertices (e.g. different atoms and atom connections)**]

2. Spatial methods for arbitrary graphs
2.1 Papers

- F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, Trans. Neural Networks 20(1):61-80, 2009 (**first neural networks on graphs**)
- Yujia Li, Daniel Tarlow, Marc Brockschmidt, Richard Zemel, Gated Graph Sequence Neural Networks, ICLR 2016 (**Gated Recurrent Unit**)
- S. Sukhbaatar, A. Szlam, R. Fergus. Learning Multiagent Communication with Backpropagation. NIPS, 2016 [**Multi-agent communication**]
- D. Duvenaud, et al, Convolutional Networks on Graphs for Learning Molecular Fingerprints, NIPS, 2015 (**graph-wise classification, similar to image recognition**)
- D. Marcheggiani and I. Titov. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. arXiv preprint arXiv:1703.04826, 2017.
- M. B. Chang, et al, A Compositional Object-Based Approach to Learning Physical Dynamics, ICLR, 2017
- Peter W. Battaglia, et al, Interaction Networks for Learning about Objects, Relations and Physics, NIPS, 2016 [**Interaction networks**]
- J. Bruna and X. Li. Community Detection with Graph Neural Networks. arXiv preprint arXiv:1705.08415, 2017 [**Community Detection**]
- Xavier Bresson, Thomas Laurent, **Residual Gated Graph ConvNets,** arXiv:1711.07553, 2017
- J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, G. E. Dahl, Neural Message Passing for Quantum Chemistry, ICML 2017 (**Modern**

GNN/MPNNs reach "Chemical Precision" on currently available datasets)

- P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. **Graph attention networks**. In ICLR, 2018

2.2 Applications

Particle physics and Chemistry:

Peter W. Battaglia, et al, Interaction Networks for Learning about Objects, Relations and Physics, NIPS, 2016

M. B. Chang, et al, A Compositional Object-Based Approach to Learning Physical Dynamics, ICLR, 2017

J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, G. E. Dahl, Neural Message Passing for Quantum Chemistry, ICML 2017

Molecule design:

D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gomez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, R. P. Adams, Convolutional Networks on Graphs for Learning Molecular Fingerprints, NIPS 2015 (molecular fingerprints using graph CNNs)

J. Bruna and X. Li. Community Detection with Graph Neural Networks. arXiv preprint arXiv:1705.08415, 2017.

Quadratic Assignment Problem:

Alex Nowak, Soledad Villar, Afonso S. Bandeira, Joan Bruna, A Note on Learning Algorithms for Quadratic Assignment with Graph Neural Networks, 2017

3. Spatial methods for manifolds

3.1 Papers

- J. Masci, D. Boscaini, M. M. Bronstein, P. Vandergheynst, Geodesic convolutional neural networks on Riemannian manifolds, 3dRR 2015 (**Geodesic CNN framework, the first intrinsic version** of convolutional neural networks on manifolds applying filters to local patches represented in geodesic polar coordinates.
- D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, Learning shape correspondence with anisotropic convolutional neural networks, NIPS2016 (**Anisotropic CNN framework,** used anisotropic heat kernels as an alternative way of extracting intrinsic patches on manifolds)
- F. Monti*, D. Boscaini*, J. Masci, E. Rodolà, J. Svoboda, M. M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model CNNs, CVPR 2017 (**MoNet framework**)
- O. Litany, T. Remez, E. Rodolà, A. M. Bronstein, M. M. Bronstein, Deep Functional Maps: Structured Prediction for Dense Shape Correspondence, 2017 (**FMNet framework**)

3.2 Applications

3D shape correspondence:

J. Masci, D. Boscaini, M. M. Bronstein, P. Vandergheynst, Geodesic convolutional neural networks on Riemannian manifolds, 3dRR 2015 (Geodesic CNN framework) - local descriptor learning

D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, Learning shape correspondence with anisotropic convolutional neural networks, NIPS2016 (Anisotropic CNN framework) - correspondence learning

F. Monti*, D. Boscaini*, J. Masci, E. Rodolà, J. Svoboda, M. M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model CNNs, CVPR 2017 (MoNet framework)

O. Litany, T. Remez, E. Rodolà, A. M. Bronstein, M. M. Bronstein, Deep Functional Maps: Structured Prediction for Dense Shape Correspondence, 2017 (FMNet framework)

3D shape synthesis:

Or Litany, Alex Bronstein, Michael Bronstein, Deformable Shape Completion with Graph Convolutional Autoencoders, 2017

Ilya Kostrikov, Joan Bruna, Surface Networks, 2017

Martin Simonovsky, Nikos Komodakis, GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders, ICLR 2018

编辑于 2018-04-12

知乎用户

321 人赞同了该回答

恰好被分到相关的paper presentation，写一下自己的总结和理解。（适合有ML基础的GCN小白 :)

1. 为什么要做图卷积

CNN已经在CV中广泛应用，就不赘述CNN的基本模型了，主要的 intuition 是用加权平均的方法（kernel / filter）提取特征值。类比到graph，GCN 的 core 也是提取拓扑图的空间特征。

传统的CNN应用在 **Euclidean Domain Data** 上，也就是规整的grid structure，比如图像，音频，句子。

然而现实中有很多都是 **Non-Euclidean Domain Data**，节点的链接方式可能很不同，比如我们的社交网络或者蛋白质结构。不能直接应用传统的CNN。

对于Euclidean Domain Data, **CNN的主要 Assumption** 是 Compositional，（CNN 提取compositional features），包括:
1. Locality (局域链接)
2. stationary (转移不变)
3. multi-scale (hierarchical 多层次)
同样的，**GCN的 Assumption** 也是，Non-Euclidean Domain Data 是 locally stationary 和 manifest hierarchical 的。那么如何在图上定义Compositionality，也就是如何做convolution和pooling就成为下一个问题。

2. 图卷积的分类

2.1. Fixed Graph

解决的问题：给一个不变的graph，和相应的 signals，进行分析，比如提取特征进行分析。具体例子可以是，事件在社交网络上的传播，脑信号在Brain connectivity network上。

主流是基于 Spectral Graph Theory 的（当然还有上面的答主提到的vertex domain，就不展开啦）。下面会展开讲。
这种方法很大的**限制**是，只能应用在fixed graph上。因为graph 中一个小小的改变，就可以改变 Fourier mode 中的representation，而对齐 Fourier mode又是一个很难的问题 （graph matching problem）。所以学到的 spectral filters is not transferable.

2.2. Variable Graph

解决的问题：给很多个graphs（可以是不同的size），以及信号，进行分析，比如对图的分类（e.g.蛋白质或其他分子结构分类）。

主要方法分为 Graph CNN 和 Graph RNN. 后面会讲他们的结果比较。

3. Spectral Convolution Net for Fixed Graph

Spectral Graph Theory 简单来讲就是用 adjacency 和 Laplacian matrix 的 eigenvector，eigenvalue 分析图的性质。

3.1. Fourier Transform

Fourier Transform on Euclidean spaces:

其实就是把一个function 分解为几个基础函数的组合，类似于投射到一个不同坐标系的空间，比如从 Cartesian coordinate 到 Spherical coordinate.

Fourier Transform on Graph 也是相同的道理，把graph 上的function 分解为几个basis 的组合。我们选择 Laplacian Matrix 的 eigenvectors 作为 Fourier basis，下面会展开讲。

3.2. Graph Laplacians

定义：大致有三种：
Notation: A is edge weights, and D is the sum of all edge weights of the same node (diagonal matrix).
a. Unnormalized Laplacian: $\Delta = D - A \rightarrow$ **Intuition:** Approximation of the minimization of RatioCut, optimize an objective relative to the number of nodes in each cluster
b. Normalized Laplacian: $\Delta = I - D^{-1/2}AD^{-1/2} \rightarrow$ **Intuition:** Approximation of the minimization of NCut, optimize the objective relative to the volume of each cluster
c. Random Walk Laplacian: $\Delta = I - D^{-1}A \rightarrow$ **Intuition:** Minimize the inter-cluster transit time of the random walker (most of the time stay in the same cluster)

Laplacian Matrix 的性质

因为graph是undirected的，所以 A （weights matrix) 是 symmetric的，D (degree matrix) 是 diagonal matrix （symmetric），所以 $\Delta = D - A$ 也是symmetric的。

我们decompose 他到eigenvector和eigenvalues： $\Delta = \Phi^T \Lambda \Phi$ ：

a. **Eigenvectors** are **real** and **orthonormal** (self-adjointness) 这一点在 Fourier Transform里很重要

b. **Eigenvalues** are **non-negetive** (posistive-semidefinite)

Intuition of Laplacian Matrix

Laplacian matrix is the discrete version of the Laplacian operator over graphs.

在Euclidean space中, Laplace operator is the divergence of the gradient of a function： $\Delta f = \nabla . \nabla f = div(grad(f))$ 在连续空间里，也就是二阶导。那让我们类比到graph中。

Graph 中的 function就是其上的signals.

Gradient 就是每个edge上的difference： $e = (u, v), grad(f)|_e = f(u) - f(v)$

Divergence, 在Euclidean space, at a point gives the net outward flux of a vector field， 同理，divergence gives the net outward flow of f at every node.

即 $(\Delta f)_i = \frac{1}{b_i} \sum_{j:(1,j)\in\varepsilon} a_{ij}(f_i - f_j) \rightarrow$ **Intuition:** difference between f and its local average (second derivative on graphs). 此表达式是从Laplacian Matrix 的定义推出。

那么为什么要用 Laplacian matrix 呢?

Laplacian operator 在连续函数中是二阶导，体现了函数的 smoothness ，在graph上也是同理。

Laplacian matrix 的第一个 eigenvector 是可以找到的最smooth的function，第二个 eigenvector 是 orthogonal to 第一个 eigenvector 的 functions 中最smooth 的，以此类推。正如前面提到的，所有的 eigenvectors 都是 orthogonal 的。

在 graph 上做 Fourier Transform，想找到 smoothest orthogonal basis on graph 作为 Fourier basis，所以取 Laplacian 的 eigenvectors。

之所以做 Fourier Transform，是因为 Convolution 在 Fourier mode 中更容易一些。

3.4. Convolution on Graph

Fourier Convolution Theory: $F\{h * f\} = F\{h\} \cdot F\{f\}$

Project the given signal to the eigenfunction of the graph Laplacian, multiply the obtained spectrum from some set of spectral coefficients that are also under the same Fourier mode. Then finally project them back to the original domain (inverse Fourier transform).

就是把 function 和 filter 都先转换到 Fourier mode，相乘后再 inverse Fourier transform 到原来的空间。

3.5. Pooling / Coarsening

Goals: Pool similar local features (max/average pooling); Series of pooling layers create invariance to global geometric deformation

Solution: Same as graph partitioning (NP hard) -> Need approximation

Balanced cuts -> Heavy-Edge Matching (HEM) **Intuition:** find the neighbors that locally maximize the normalized association for each vertice

3.6. Limitations

1. Definition of Laplacian matrix on directed graph is unclear.
2. Fourier model is not robust to perturbation (change one edge -> change the topology of the graph). Align Fourier mode is hard, not

guarantee good generalization (graph matching problem). -> only works with fixed size graph

4. ConvNet for Variable Graph

To Do

5. GCN 的迁移学习

To Do

Update:

之前一直在赶ddl和QE，最近开始做GCN相关；贴两篇私以为很有用的survey， 结构和应用都很详细而且 up to date（给清华老师打call

1. Deep Learning on Graphs: A Survey
arxiv.org/pdf/1812.0420...
2. Graph Neural Networks: A Review of Methods and Applications
arxiv.org/pdf/1812.0843...

---

p.s. 有什么错误或者typo望

编辑于 2018-12-24