Graph Convolutional Networks

欧阳剑波

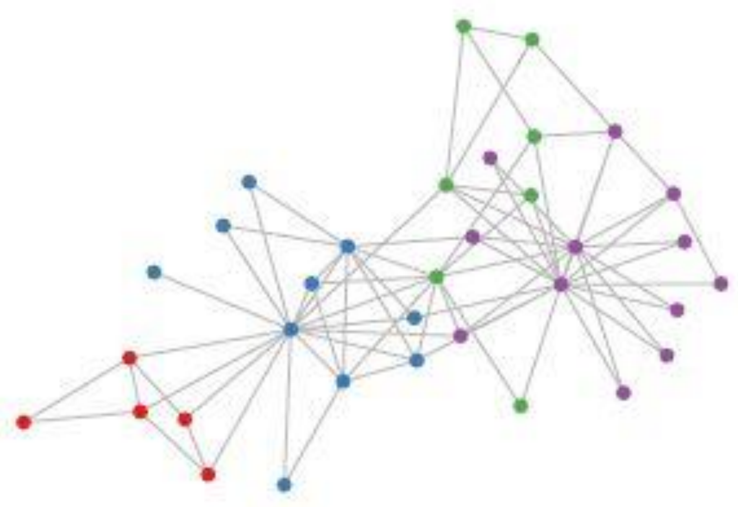
2018.11.30

Outline

- Overview
- Spectral domain method
- Spatial domain method

Non Euclidean Structure

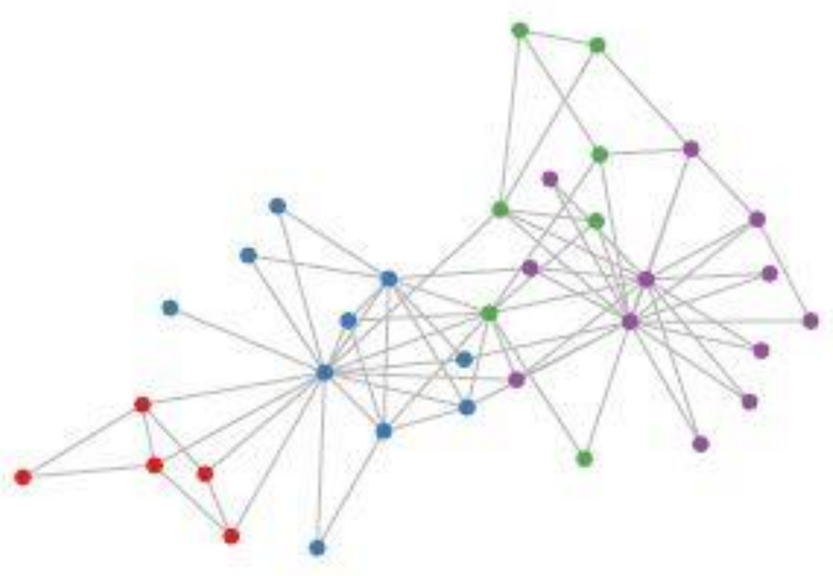
- Euclidean Structure
  - Video – 3D grid
  - Image - 2D grid
  - Voice - 1D grid
- Non Euclidean Structure
  - 社交网络
  - 信息网络
  - 化合物结构

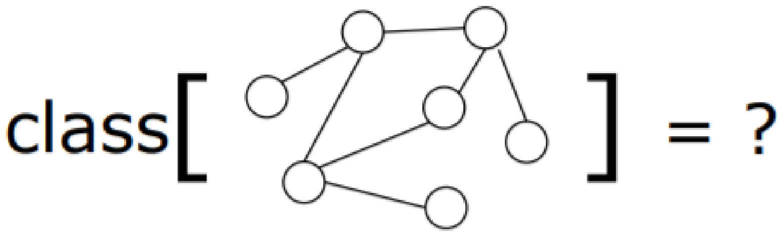


- CNN处理的数据是Euclidean Structure Data
- CNN无法处理Non Euclidean Structure Data
  - 拓扑图中每个顶点的相邻顶点数目都可能不同，无法用一个同样尺寸的卷积核来进行卷积运算

Task

- 提取拓扑图的空间特征
  - 预测graph中节点的标签
  - 预测graph的标签





Two direction

- Spectral domain
  - Based on the spectral graph theory
  - Convolution operation is defined in the Fourier domain
- Spatial domain
  - Define convolutions directly on the graph

图的傅里叶变换

- 传统傅里叶变换 $F(\omega)=\mathcal{F}[f(t)]=\int f(t)e^{-i\omega t}dt$
- 拉普拉斯算子 $\Delta=\sum_{i=1}^{n}\frac{\partial^2}{\partial x_i^2}$
- 特征方程 $\Delta g=\lambda g$
- e^{-jwt}是特征方程的解 $\Delta e^{-iwt}=\frac{\partial^2}{\partial x_i^2}e^{-iwt}=-w^2e^{-iwt}$
- 傅里叶变换是时域信号与拉普拉斯算子特征函数乘积的积分

拉普拉斯矩阵的谱分解

- 图G=(V,E)
- 拉普拉斯矩阵L = D - A ,对称矩阵
  - D：顶点的度矩阵(对角矩阵)
    - ✓ 度：某个顶点的度是图中与该顶点相连的边的数目
  - A：图的邻接矩阵
    - ✓ 若节点i与j相连，则A(i, j)=1，否则为0
- 归一化拉普拉斯矩阵 $L^{sys} = D^{-1/2} L D^{-1/2}$
- 谱分解 $L = U \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} U^{-1}$ $U = (\overrightarrow{u_1}, \overrightarrow{u_2}, \cdots, \overrightarrow{u_n})$

  - U：列向量为单位特征向量的矩阵
  - $\begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$ 是n个特征值构成的对角阵

拉普拉斯矩阵的谱分解

- U是正交矩阵，即 $UU^T = E$
- 谱分解写为：

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

图的傅里叶变换

- 傅里叶变换是时域信号与拉普拉斯算子特征函数乘积的积分
- 推广到图：图上的傅里叶变换是时域信号与图的拉普拉斯方程的特征向量的和

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

- 写成矩阵形式

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

$$\hat{f} = U^T f$$

图的傅里叶逆变换

- 传统的傅里叶逆变换 $\mathcal{F}^{-1}[F(\omega)] = \frac{1}{2\Pi}\int F(\omega)e^{i\omega t}d\omega$

- 迁移到图

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

- 矩阵形式

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

$$f = U\hat{f}$$

卷积定理

- 函数卷积的傅里叶变换是函数傅里叶变换的乘积
- 对非欧式结构的数据无法直接做卷积运算，所以现在傅里叶域做乘积，再做逆变换
- 卷积核h在图上的傅里叶变换 $\hat{h}(\lambda_l)=\sum_{i=1}^{N}h(i)u_l^*(i)$
- 写成对角矩阵 $\begin{pmatrix}\hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n)\end{pmatrix}$
- 最终形式

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

第1代谱卷积

$$(f*h)_G = U \begin{pmatrix} \hat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{pmatrix} U^T f \quad \Rightarrow \quad y_{output} = \sigma\left(U \begin{pmatrix} \theta_1 & & \\ & \ddots & \\ & & \theta_n \end{pmatrix} U^T x \right)$$

- 缺点
  - 计算复杂度高
  - 没有体现传统CNN的空间局部性，每次卷积要考虑所有的节点
  - 卷积核有n个参数

Spectral Networks and Deep Locally Connected Networks on Graphs_ICLR_2014

第2代谱卷积

$$y_{output} = \sigma\left(U\begin{pmatrix}\theta_1 & & \\ & \ddots & \\ & & \theta_n\end{pmatrix}U^T x\right) \Rightarrow y_{output} = \sigma\left(U\begin{pmatrix}\sum_{j=0}^{K}\alpha_j\lambda_1^j & & \\ & \ddots & \\ & & \sum_{j=0}^{K}\alpha_j\lambda_n^j\end{pmatrix}U^T x\right)$$

- 推导可得

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

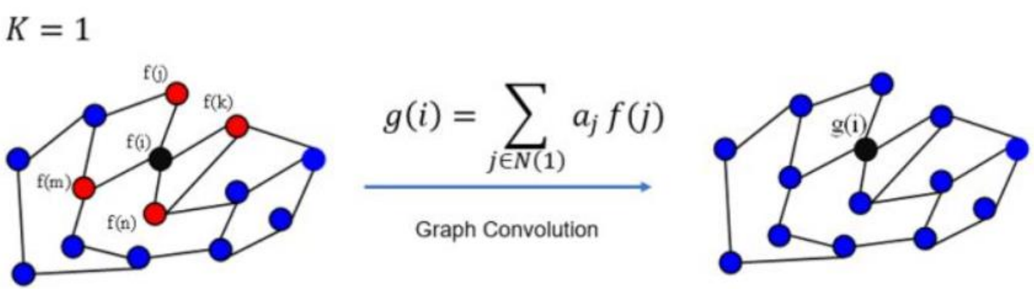
$$y_{output} = \sigma\left(\sum_{j=0}^{K}\alpha_j L^j x\right)$$

- 优势
  - 卷积核参数量为K，k<<n
  - 不再需要做谱分解

Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering_NIPS_2016

第2代谱卷积

- 优势
  - 卷积核具有良好的空间局部性
  - 拉普拉斯矩阵的性质
    - ✓ 若两节点i，j不相连，则L(i, j)=0
    - ✓ 若d(m,n)>s, 则$L^s(m,n) = 0$
  - 卷积核的感受野大小为K，每次卷积会将中心顶点K-hop neighbor上的feature进行加权求和



Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering_NIPS_2016

第2代谱卷积

- 切比雪夫多项式逼近
- 切比雪夫展开式

$$T_0(x) = 1$$
$$T_1(x) = x$$
$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$$

- 最终形式

$$g_{\theta'} \star x \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{L}) x$$

$$\tilde{L} = \frac{2}{\lambda_{max}} L - I_N$$

Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering_NIPS_2016

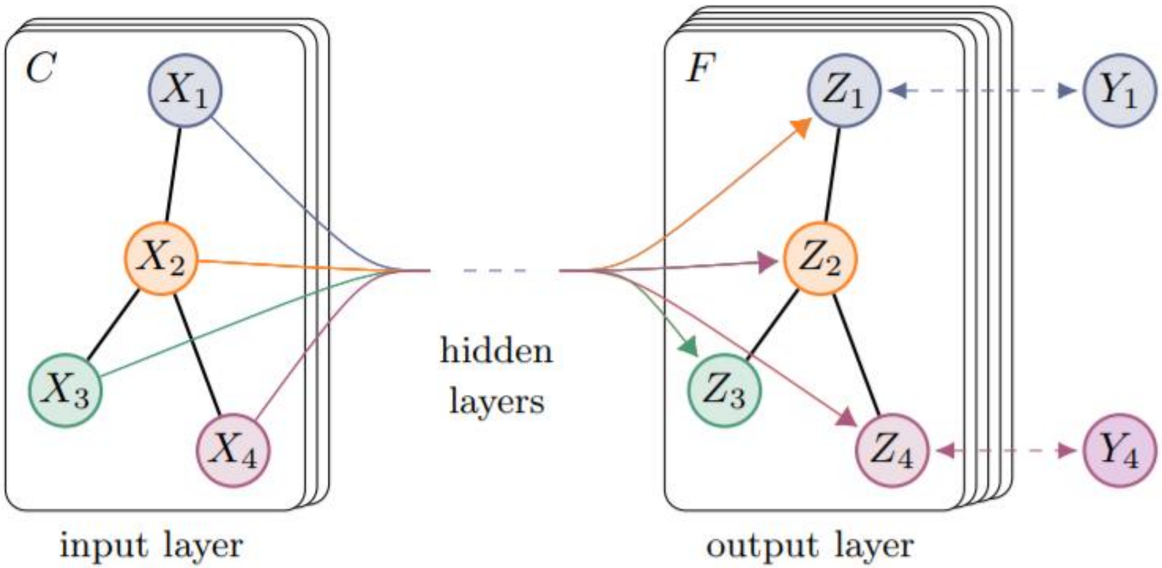
第3代谱卷积

- 二代卷积的基础上，令K=1，卷积只考虑直接领域，类似于3*3卷积核
- 为了简化运算，定义 $\lambda_{max} \approx 2$
- 得到一阶近似 $g_{\theta'} \star x \approx \theta'_0 x + \theta'_1 (L - I_N) x = \theta'_0 x - \theta'_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x$
- 进一步简化 $\theta = \theta'_0 = -\theta'_1$ 得到 $g_\theta \star x \approx \theta \left(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \right) x$
- 最终形式 $Z = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta,$

$$\tilde{A} = A + I_N \text{，} \quad \tilde{D}_{ii} = \Sigma_j \tilde{A}_{ij}$$

第3代谱卷积

□ 实验



$$\mathcal{L} = -\sum_{l \in \mathcal{Y}_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$

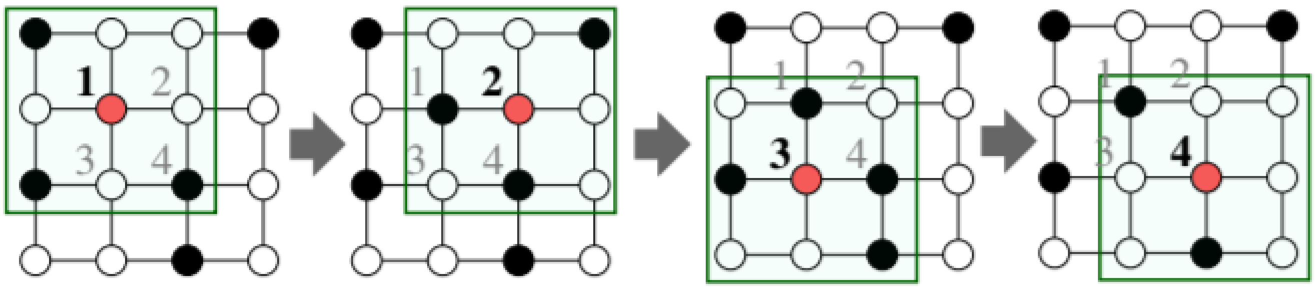
Semi-Supervised Classification with Graph Convolutional Networks_ICLR_2017

Spatial domain

- Define convolutions directly on the graph
  - 图的节点有不同数量的邻节点
- 选取固定大小的邻域做卷积

Learning Convolutional Neural Networks for Graphs

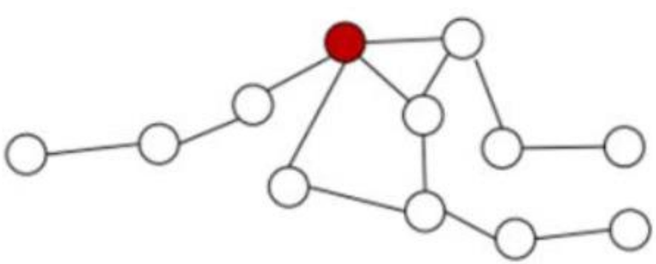
- 将image看作特殊的graph



- 需要体现出邻域的空间位置信息
- 算法流程
  - 选出合适的节点
  - 为每个节点建立邻域，构成子图
  - 将节点及其邻域节点特征表示为feature map

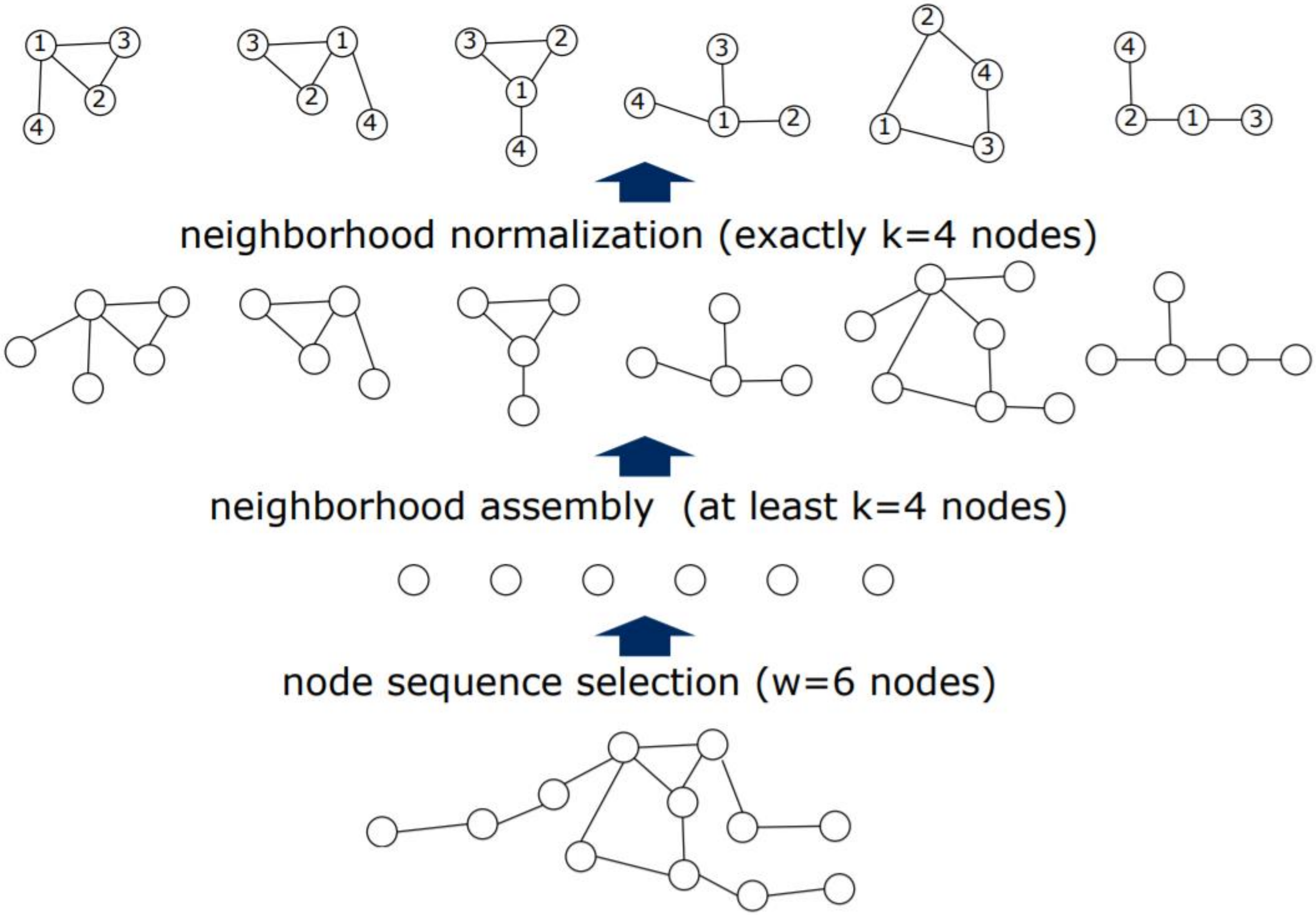
Learning Convolutional Neural Networks for Graphs

- 选出合适的节点
  - 对输入图选定节点个数为w
  - 对图中的节点进行排序
    - ✓ 中心化：某节点与其余所有节点的距离之和越小，越处于图的中心



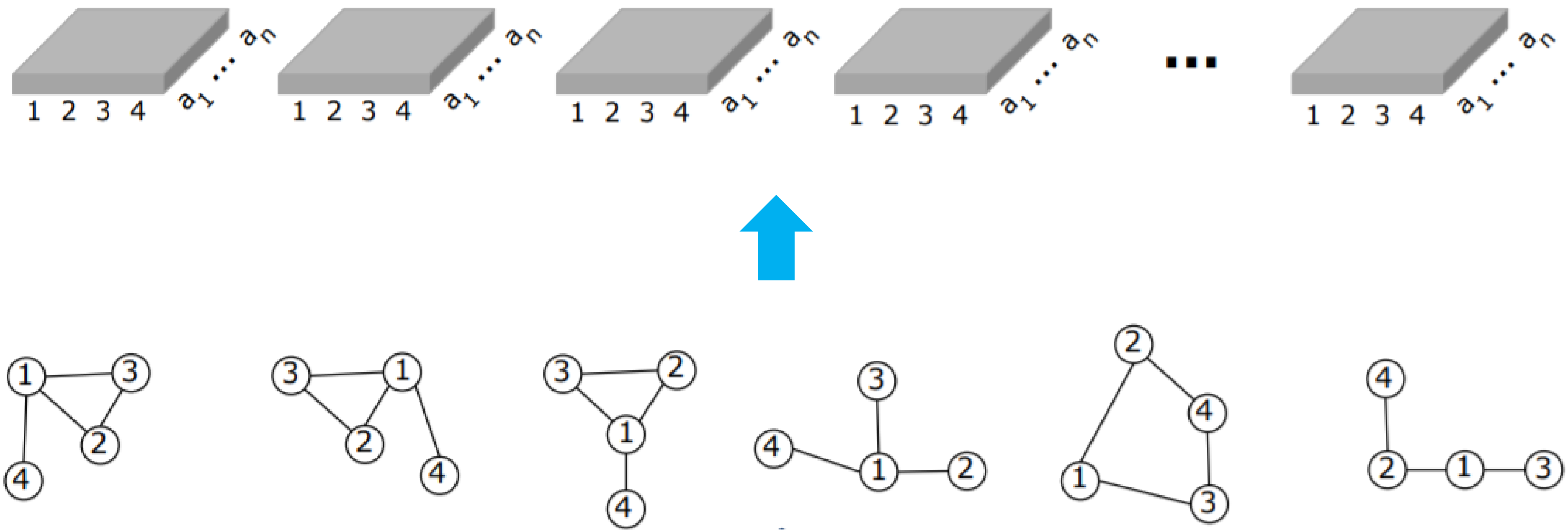
Learning Convolutional Neural Networks for Graphs

- 选出节点的邻域大小为k

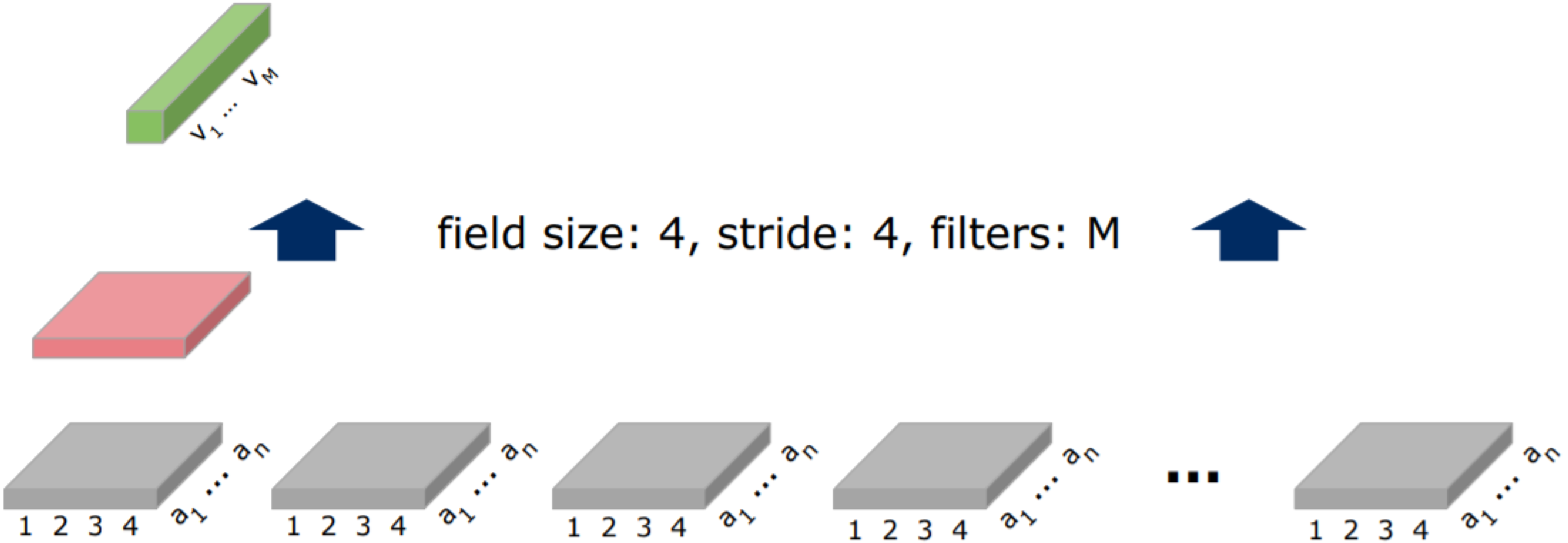


Learning Convolutional Neural Networks for Graphs

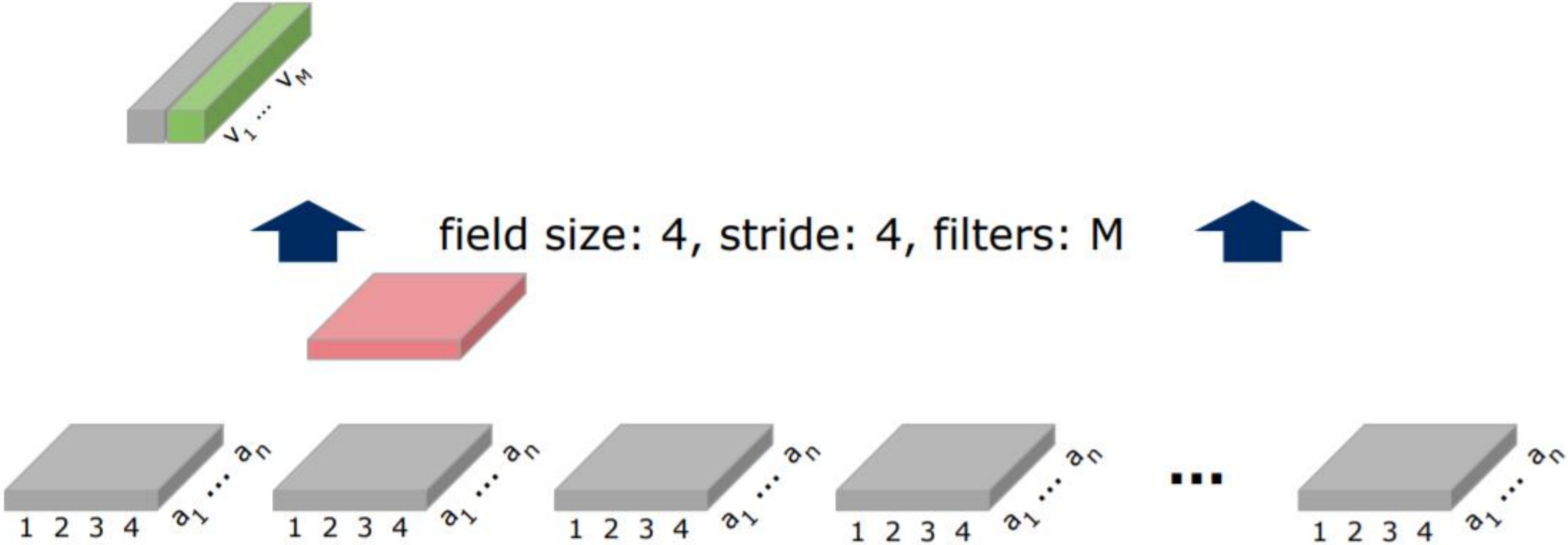
- 组成feature map



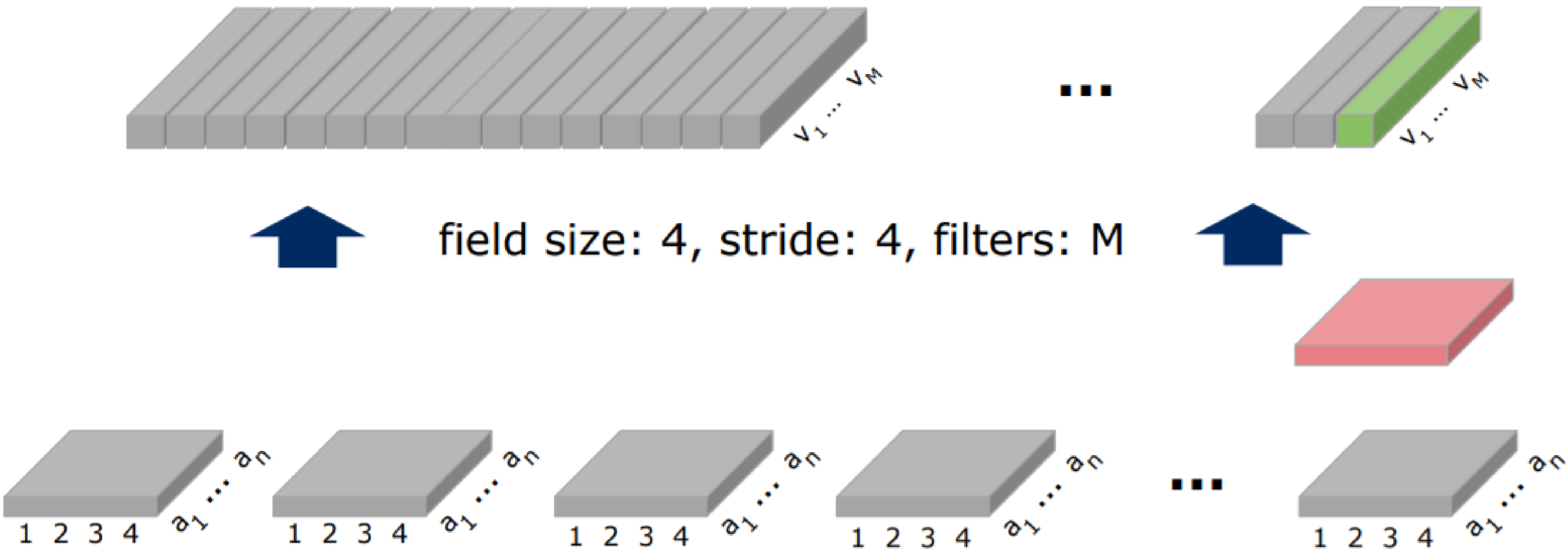
Learning Convolutional Neural Networks for Graphs



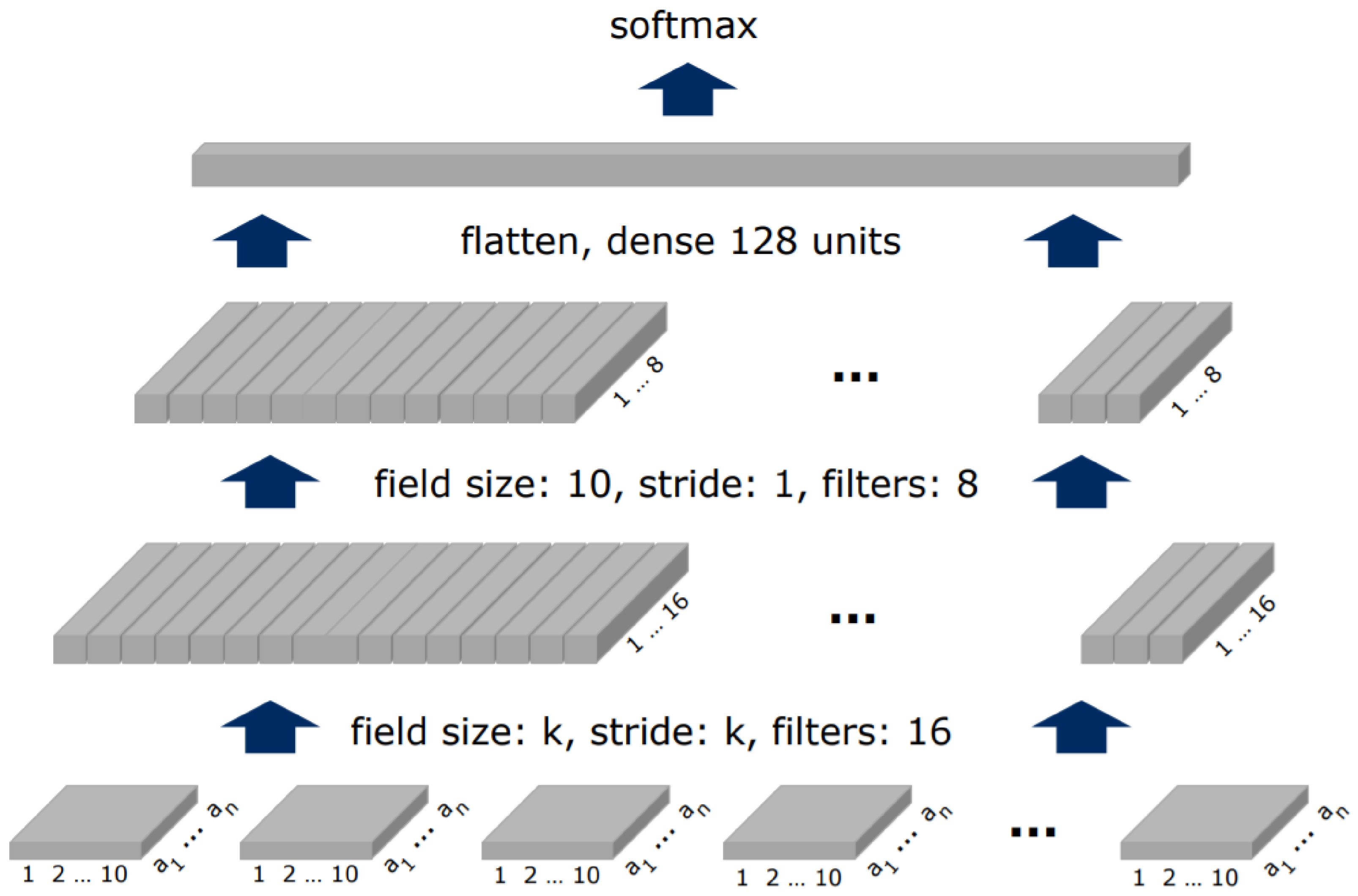
Learning Convolutional Neural Networks for Graphs



Learning Convolutional Neural Networks for Graphs



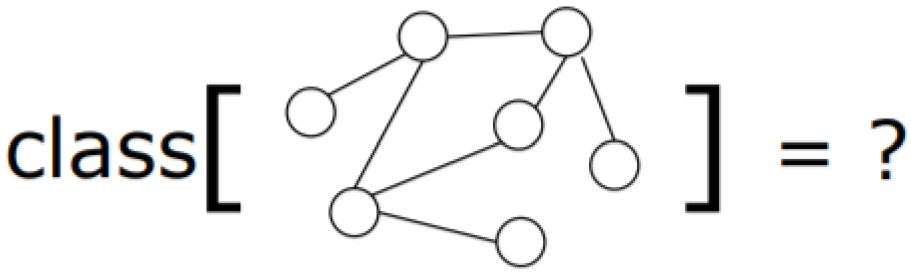
Learning Convolutional Neural Networks for Graphs



Learning Convolutional Neural Networks for Graphs

- Train

- Test



Summary

- 谱图卷积在傅里叶域做卷积变换，有完整的理论支持。
- 空间卷积更加灵活，核心点在于选取定量的邻域。