Graph Convolutional Neural Networks

沈华伟

shenhuawei@ict.ac.cn

微博: 沈华伟_ICT

中国科学院计算技术研究所

智源社区AI周刊

每周一封新邮件，AI动态悉数知

学术 | 行业 | 政策 | 数据 | 产品 | 求职

即刻订阅，了解人工智能领域过去一周内，**值得关注的新思想、新动向和新成果**

扫码订阅

北京智源人工智能研究院
BEIJING ACADEMY OF ARTIFICIAL INTELLIGENCE

智源AI社群

人工智能技术交流群

智能架构与芯片　自然语言处理　智能信息检索与挖掘

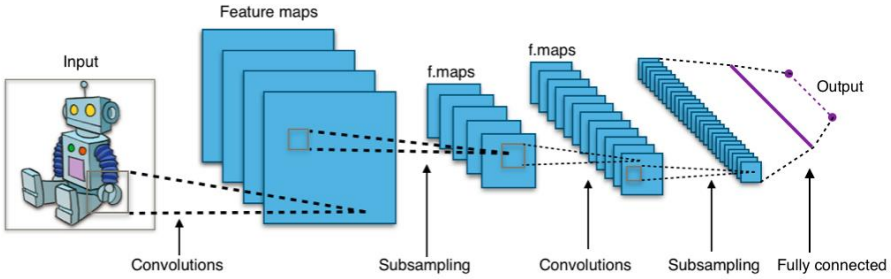
人工智能的数理基础　机器学习　脑科学与 AI　人工智能伦理

AI 医疗　自动驾驶　数据认识与认知推理

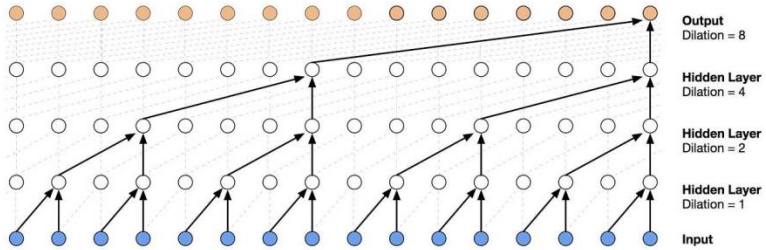
扫描二维码关注公众号
「北京智源人工智能研究院」
在对话界面
回复“微信群”获取进群方法

Convolutional Neural Network

- Convolutional neural network (CNN) gains great success on Euclidean data, e.g., image, text, audio, and video
  - Image classification, object detection, machine translation



Convolutional neural networks on image



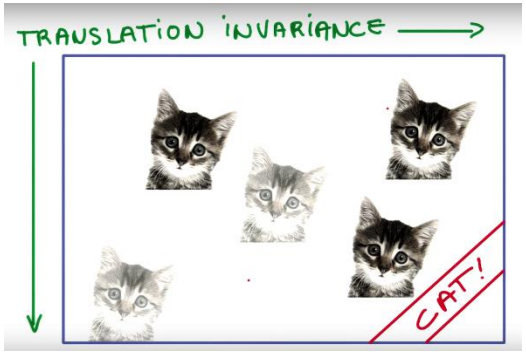
Temporal convolutional network

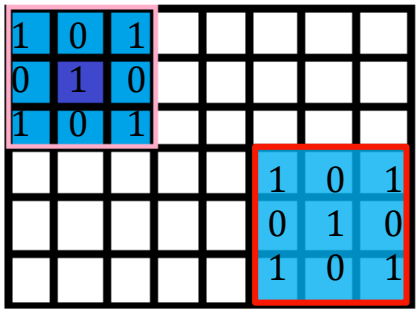
- The power of CNN lies in
  - its ability to learn local stationary structures, via localized convolution filter, and compose them to form multi-scale hierarchical patterns

M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, P. Vandergheynst. Geometric deep learning: going beyond Euclidean data. IEEE Signal Processing Magazine, 18-42, 2017.

Convolutional Neural Network

- **Localized convolutional filters are translation- or shift-invariant**
  - **Which are able to recognize identical features independently of their spatial locations**





X-Shape

Template Matching

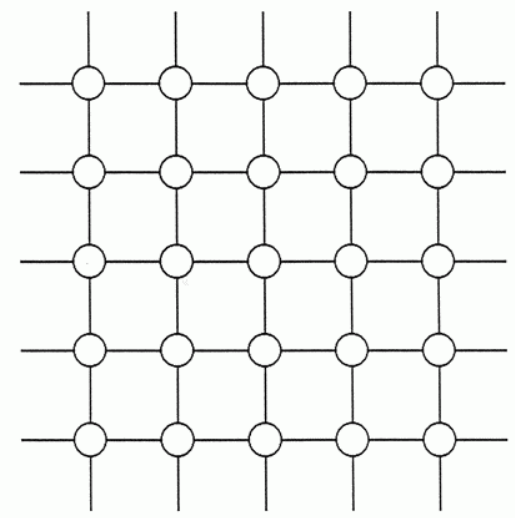
1	0	1
0	1	0
1	0	1

- **One interesting problem is how to generalize convolution to non-Euclidean domain, e.g., graph?**
  - **Irregular structure of graph poses challenges for defining convolution for graph data**

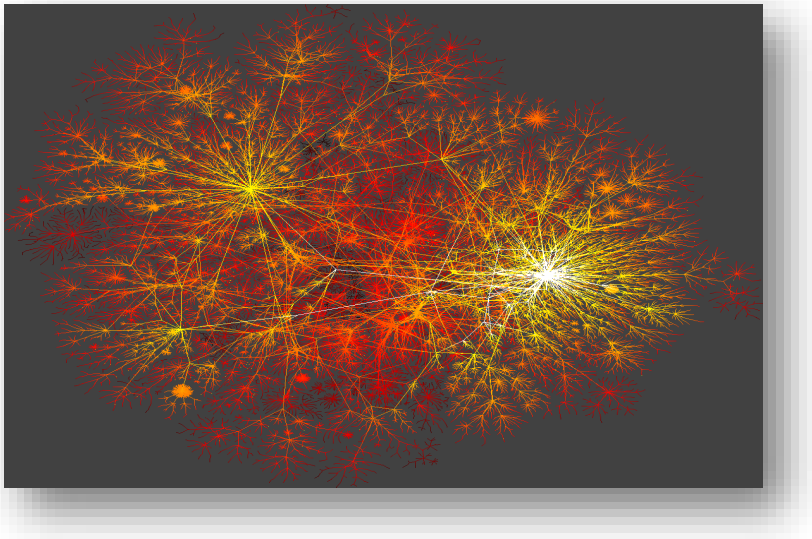
LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. Nature, 521(7553):436, 2015

From CNN to graph CNN

- Convolution is well defined in Euclidean data, grid-like network
- Not straightforward to define convolution on irregular network, widely observed in real world



Grid-like network



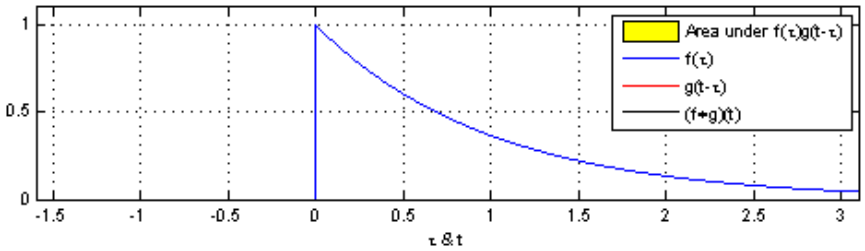
Irregular networks

Convolution

- **Convolution is a mathematical operation on two functions, f and g, to produce a third function h.**
  - **Defined as the integral, in continuous case, or sum, in discrete case, of the product of the two functions after one is reversed and shifted.**

Continuous case

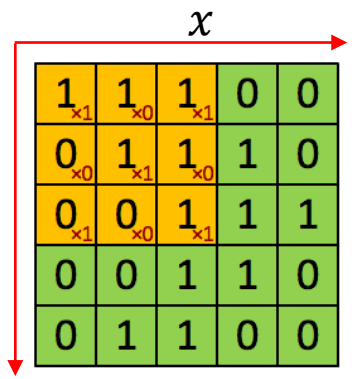
$$h(t) = (f * g)(t) \stackrel{\text{def}}{=} \int f(t)g(t-\tau)\,d\tau$$

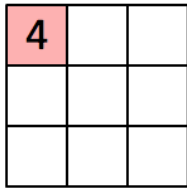


连续数据的卷积

Discrete case

$$h(x,y)$$
$$= (f * g)(x,y)$$
$$\stackrel{\text{def}}{=} \sum_{m,n} f(x-m, y-n)g(m,n)$$





$g=$

$g(1,1)$ 1	$g(0,1)$ 0	$g(-1,1)$ 1
$g(1,0)$ 0	$g(0,0)$ 1	$g(-1,0)$ 0
$g(1,-1)$ 1	$g(0,-1)$ 0	$g(-1,-1)$ 1

离散数据的卷积

Existing methods to define convolution

- **Spectral methods: define convolution in spectral domain**
  - **Convolution is defined via graph Fourier transform and convolution theorem.**
  - **The main challenge is that convolution filter defined in spectral domain is not localized in vertex domain.**

- **Spatial methods: define convolution in the vertex domain**
  - **Convolution is defined as a weighted average function over all vertices located in the neighborhood of target vertex.**
  - **The main challenge is that the size of neighborhood varies remarkably across nodes, e.g., power-law degree distribution.**

每个节点的邻域大小都不同

Spectral methods for graph convolutional neural networks

Spectral methods

- Given a graph $G = (V, E, W)$
  - V is node set with $n = |V|$, E is edge set, and $W \in R^{n \times n}$ is the weighted adjacency matrix
    结点集合 边集合 邻接矩阵
  - Each node is associated with d features, and $X \in R^{n \times d}$ is the feature matrix of nodes, each column of X is a signal defined over nodes
    每个结点都有d个特征，每个特征值不同，相当于是对Graph上的信号处理

- Graph Laplacian
  拉普拉斯矩阵 度矩阵
  - $L = D - W$, where is a diagonal matrix with $D_{ii} = \sum_j W_{ij}$
  - Normalized graph Laplacian
    求信号关于图的平滑程度

normalized版本的拉普拉斯矩阵有更好的数学性质

$$L = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

D1/2W这个部分是为了转换到谱域中

where I is the identity matrix. 单位矩阵

Graph Fourier Transform

现在已经将空间域信号转换到谱域中，需要找到谱域中的一组基（正弦波），也就是：

- **Fourier basis of graph G**

求出空间域信号在不同频率的正弦波（拉普拉斯矩阵的正交特征向量）上的权重（在频域中的表达）

  - **The complete set of orthonormal eigenvectors $\{u_l\}_{l=1}^{n}$ of L, ordered by its non-negative eigenvalues $\{\lambda_l\}_{l=1}^{n}$**

正交特征向量

非负特征值

  - **Graph Laplacian could be diagonalized as**

$$L = U\Lambda U^T$$

where $U = [u_1, \cdots, u_n]$, and $\Lambda = \mathbf{diag}([\lambda_1, \cdots, \lambda_n])$

对角矩阵

- **Graph Fourier transform**
  - **Graph Fourier transform of a signal $x \in R^n$ is defined as**

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

得到x在谱域中的表达 也就是原始信号在每个基（拉普拉斯矩阵正交特征向量）上的权重大小

  - **Graph Fourier inverse transform is**

$$x = U\hat{x}$$

傅里叶逆变换

Define convolution in spectral domain

- **Convolution theorem** 卷积定理
  - **The Fourier transform of a convolution of two signals is the point-wise product of their Fourier transforms**

    两个信号卷积的傅里叶变换是它们的傅里叶变换的点积

- **According to convolution theorem, given a signal x as input and the other signal y as filter, graph convolution $*_G$ could be written as**

$$x *_G y = U\left(\left(\boldsymbol{U^T x}\right)\odot\left(\boldsymbol{U^T} y\right)\right)$$

Here, the convolution filter in spectral domain is $\boldsymbol{U^T} y$.

卷积滤波器

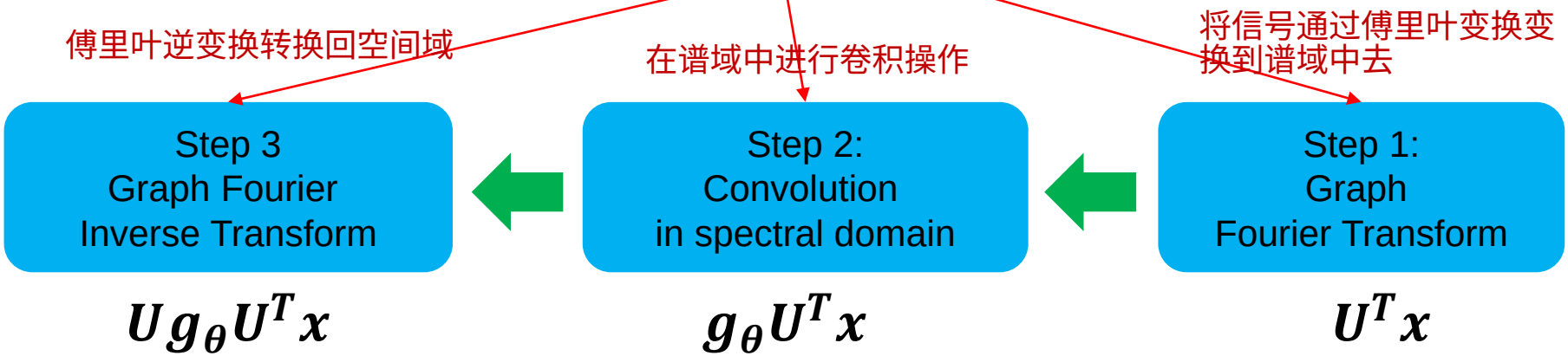
卷积过程：
1、先求傅里叶变换后的结果（谱域中的结果）
2、再求两者的点积
3、再求傅里叶逆变换

Define convolution in spectral domain

- **Graph convolution in spectral domain**
  - **Let $U^T y = [\theta_0, \cdots, \theta_{n-1}]^T$ and $g_\theta = \mathbf{diag}([\theta_0, \cdots, \theta_{n-1}])$, we have**

$$x *_G y = U\left((U^T x) \odot (U^T y)\right)$$

$$x *_G y = U g_\theta U^T x$$



Spectral Graph CNN

- **Spectral Graph CNN**

$$x_{k+1,j} = h\left(\sum_{i=1}^{f_k} U F_{k,i,j} U^T x_{k,i}\right)$$

$$j = 1, \cdots, f_{k+1}$$

Signals in the k-th layer

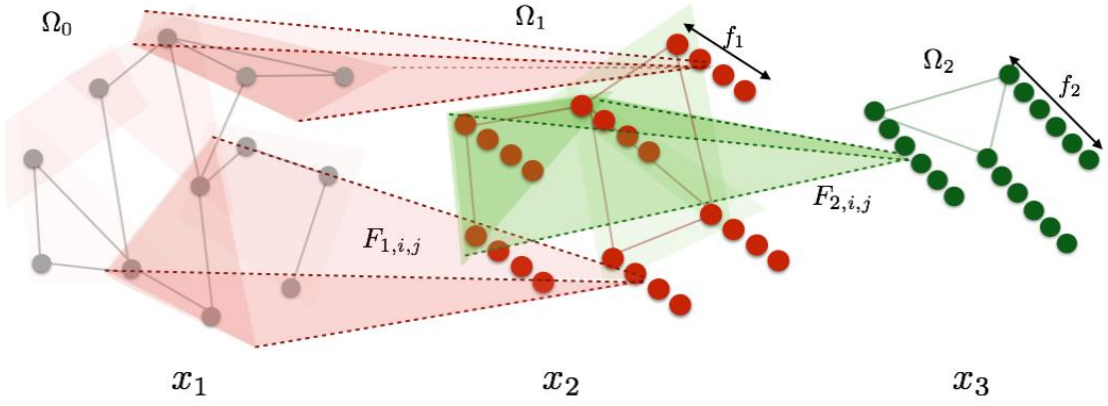
Filter in the k-th layer

Graph Fourier Transform

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

Graph Fourier Inverse Transform

$$x = U\hat{x}$$



J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. ICLR, 2014.

Shortcomings of Spectral graph CNN

- **Requiring eigen-decomposition of Laplacian matrix**
  - **Eigenvectors are explicitly used in convolution**

拉普拉斯矩阵分解

- **High Computational cost**
  - **Multiplication with graph Fourier basis U is $O(n^2)$**

计算复杂度较高

- **Not localized in vertex domain**

在结点域上不是局部化的

ChebyNet: parameterizing filter

- **Parameterizing convolution filter via polynomial approximation**

将自由的卷积核通过多项式函数进行进一步的参数化，从而实现了卷积核的空间约束，并且不再依赖于U

$$g_\theta = \mathbf{diag}([\theta_0, \cdots, \theta_{n-1}])$$

核心思想：进一步参数化卷积核

$$g_\beta(\Lambda) = \sum_{k=0}^{K-1} \beta_k \Lambda^k \qquad \Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$$

- **ChebyNet**

$$x *_G y = U g_\beta(\Lambda) U^T x = \sum_{k=0}^{K-1} \beta_k L^k x$$

U消失了 计算复杂度大大降低

参数减少

The number of free parameters reduces from n to K

M. Defferrard, X. Bresson, P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. NeuralIPS, 2016.

ChebyNet vs. Spectral Graph CNN

优点：

- **Eigen-decomposition is not required**

不需要矩阵分解

- **Computational cost is reduced from $O(n^2)$ to $O(|E|)$**

$$x *_G y = U g_\beta(\Lambda) U^T x = \sum_{k=0}^{K-1} \beta_k L^k x$$

计算复杂度降低
拉普拉斯矩阵中非零元素的个数和边成正比

- **Convolution is localized in vertex domain**
  - **Convolution is strictly localized in a ball of radius K, i.e., K hops from the central vertex**

K个结点的半径 即K跳
通常取3或者4

Is this method good enough? What could we do more?

M. Defferrard, X. Bresson, P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. NeuraIPS, 2016.

Our method:
Graph Wavelet Neural Network
(ICLR 2019)

图小波神经网络

Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, Xueqi Cheng. Graph wavelet neural network. ICLR 2019.

Graph wavelet neural network

- **ChebyNet achieves localized convolutional via restricting the space of graph filters as a polynomial function of eigenvalue matrix Λ**

$$g_\theta(\Lambda) = \sum_{k=0}^{K-1} \theta_k \Lambda^k$$

- **We focus on the Fourier basis to achieve localized graph convolution**

$$x *_G y = U g_\theta U^T x$$

- **We propose to replace Fourier basis with wavelet basis.**

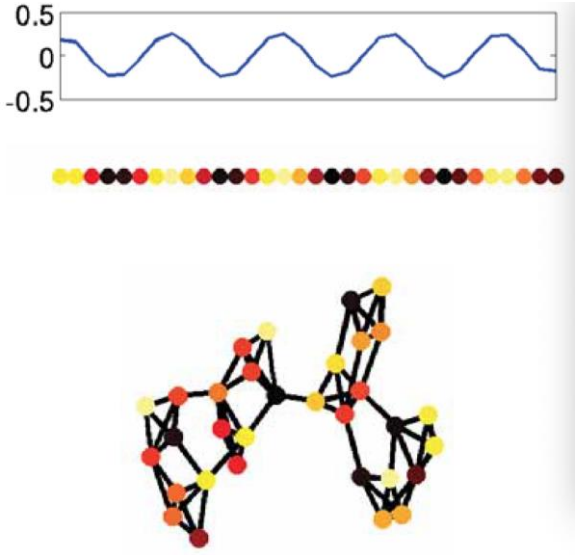
Fourier vs. Wavelet

Fourier Basis

拉普拉斯基
（傅里叶基）

- Dense
- Not localized
- High Computational cost

正弦波

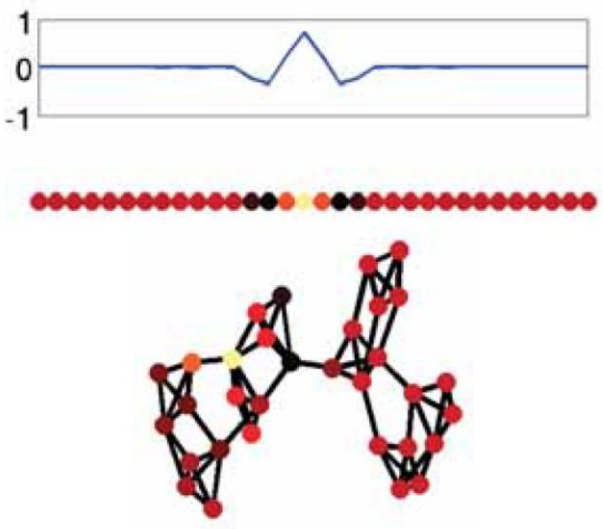


Fourier basis：U

Wavelet Basis

小波基

- Sparse
- Localized
- Low Computational cost



Wavelet basis：$\psi_s = Ue^{\lambda s}U^T$

B. Xu, H. Shen, Q. Cao, Y. Qiu, X. Cheng. Graph wavelet neural network. ICLR 2019.

Graph wavelet neural network

- **Graph Wavelet Neural Network**
  - **Replace graph Fourier transform with graph wavelet transform**

Graph Fourier transform

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

Inverse Fourier transform

$$x = U\hat{x}$$

Graph Wavelet transform

$$x^* = {\psi_s}^{-1} x$$

Inverse Wavelet transform

$$x = \psi_s x^*$$

Graph wavelet neural network (GWNN)

- **Graph convolution via wavelet transform**

$$\boldsymbol{x} *_{\mathcal{G}} \boldsymbol{y} = \boldsymbol{U}\left((\boldsymbol{U}^{\top}\boldsymbol{y}) \odot (\boldsymbol{U}^{\top}\boldsymbol{x})\right),$$

$$\boldsymbol{x} *_{\mathcal{G}} \boldsymbol{y} = \psi_s\left((\psi_s^{-1}\boldsymbol{y}) \odot (\psi_s^{-1}\boldsymbol{x})\right)$$

Replacing basis

核心思想：基的替换

- **Graph wavelet neural network**

$$x_{k+1,j} = h\left(\sum_{i=1}^{p} U F_{k,i,j} U^{T} x_{k,i}\right) \Rightarrow x_{k+1,j} = h\left(\sum_{i=1}^{p} \psi_s F_{k,i,j} \psi_s^{-1} x_{k,i}\right)$$

$j = 1, \cdots, q$

Parameter complexity: $O(n * p * q)$

Reducing parameter complexity

- **Key idea:** 从特征变换中分离图卷积
  - **Detaching graph convolution from feature transformation**

$$x_{k+1,j} = h\left(\sum_{i=1}^{p} \psi_s F_{k,i,j} \psi_s^{-1} x_{k,i}\right) \qquad j = 1, \cdots, q$$

$\boldsymbol{T} \in \boldsymbol{R}^{q \times p}$ with $\boldsymbol{p} * \boldsymbol{q}$ parameters

F_k is a diagonal matrix with $\boldsymbol{n}$ parameters

$$y_{k,j} = \sum_{i=1}^{p} T_{ji} x_{k,i} \quad + \quad x_{k+1,j} = h\left(\psi_s F_k \psi_s^{-1} y_{k,j}\right)$$

Feature transformation　　Graph convolution

The number of parameters reduces from $\boldsymbol{O(n * p * q)}$ to $\boldsymbol{O(n + p * q)}$

GWNN vs. ChebyNet

- **Benchmark datasets**

基准数据集

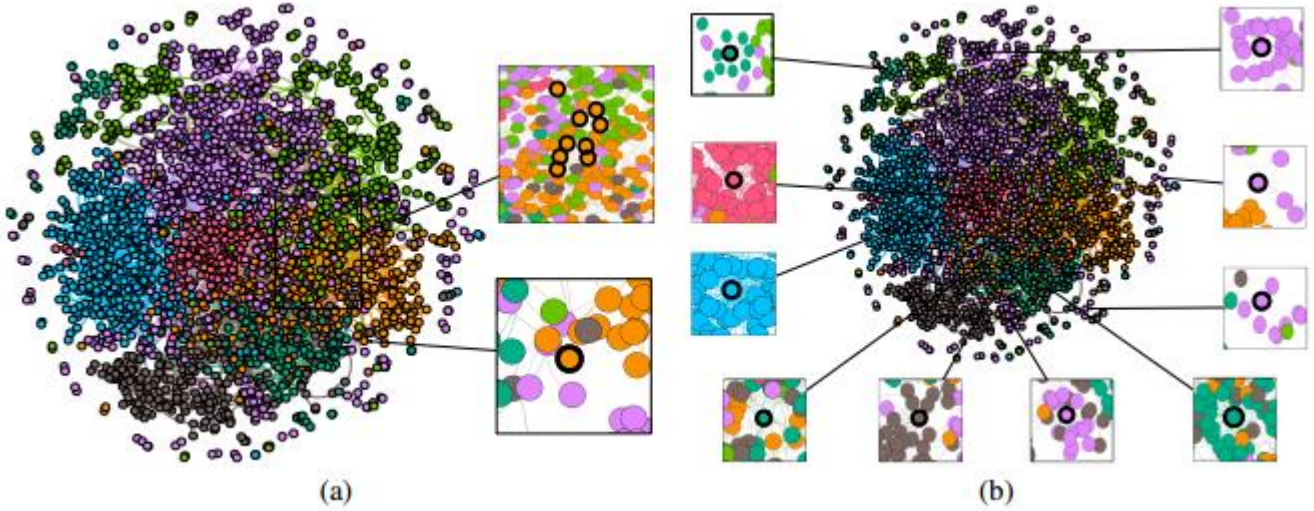
Dataset	Nodes	Edges	Classes	Features	Label Rate
Citeseer	3,327	4,732	6	3,703	0.036
Cora	2,708	5,429	7	1,433	0.052
Pubmed	19,717	44,338	3	500	0.003

- **Results at the task of node classification**

Method	Cora	Citeseer	Pubmed
MLP	55.1%	46.5%	71.4%
ManiReg	59.5%	60.1%	70.7%
SemiEmb	59.0%	59.6%	71.7%
LP	68.0%	45.3%	63.0%
DeepWalk	67.2%	43.2%	65.3%
ICA	75.1%	69.1%	73.9%
Planetoid	75.7%	64.7%	77.2%
Spectral CNN	73.3%	58.9%	73.9%
ChebyNet	81.2%	69.8%	74.4%
GWNN	**82.8%**	**71.7%**	**79.1%**

Graph wavelet neural network

- Each Graph wavelet offers us a local view, i.e., from a center node, about the proximity for each pair of nodes



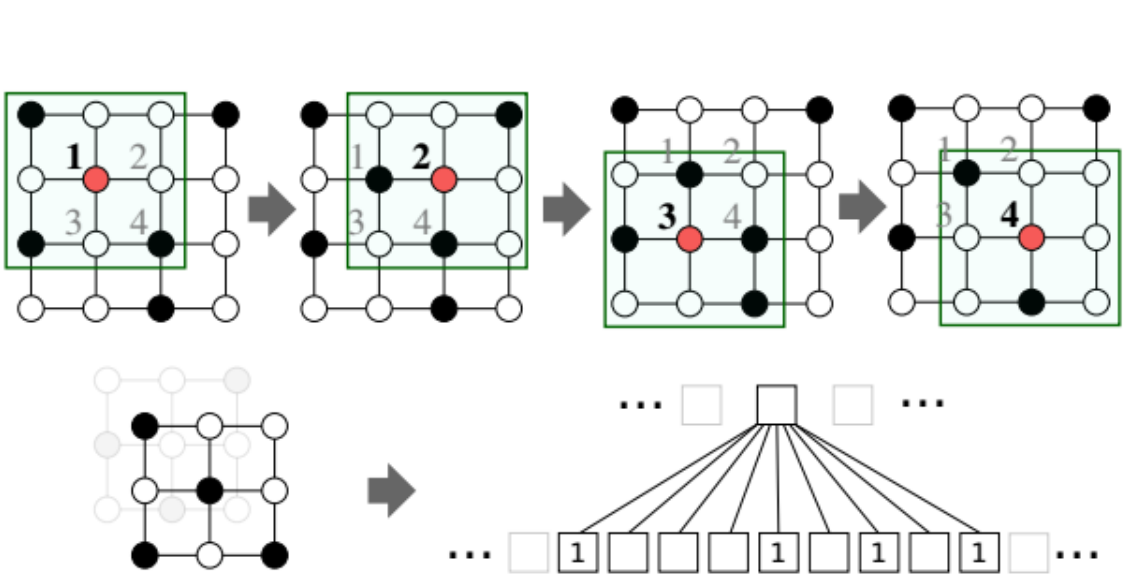
Wavelet offers us a better basis for defining graph convolutional networks in spectral domain

空间方法

Spatial methods for graph convolutional neural networks

Spatial Methods for Graph CNN

- **By analogy**
  - **What can we learn from the architecture of standard convolutional neural network?**



1. Determine Neighborhood

2. Impose an order in neighborhood

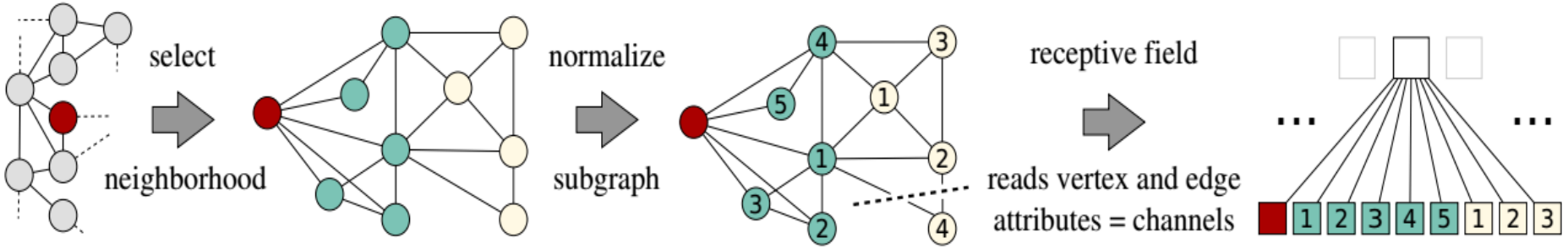
3. Parameter sharing

M. Niepert, M. Ahmed, K. Kutzkov. Learning Convolutional Neural Networks for Graphs. ICML, 2016.

Spatial Methods for Graph CNN

- **By analogy**
  - **For each node, select the fixed number of nodes as its neighboring nodes, according to certain proximity metric**
  - **Impose an order according to the proximity metric**
  - **Parameter sharing**

相似度度量



1. Determine Neighborhood

2. Impose an order in neighborhood

3. Parameter sharing

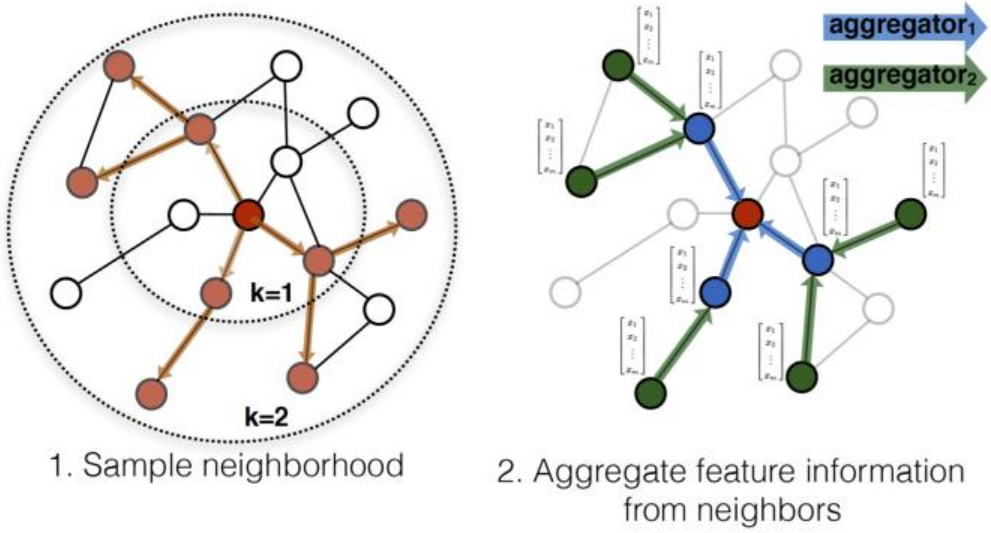
M. Niepert, M. Ahmed, K. Kutzkov. Learning Convolutional Neural Networks for Graphs. ICML, 2016.

Spatial Methods for Graph CNN

- **GraphSAGE**
  - **Sampling neighbors** 随机采样邻域 随机行走 直到找到k个邻居
  - **Aggregating neighbors** 聚合邻域

$$a_v^{(k)} = \text{AGGREGATE}^{(k)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

$$h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right)$$



1. Sample neighborhood

2. Aggregate feature information from neighbors

GraphSAGE: Inductive Learning

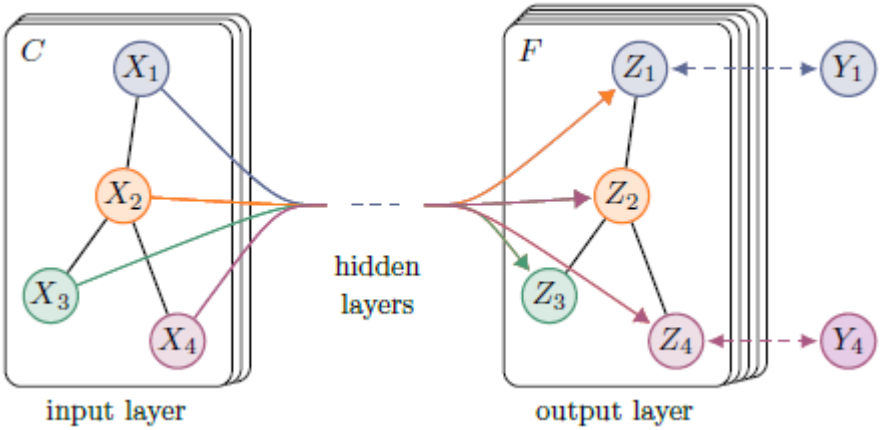
General framework of graph neural networks:
Aggregate the information of neighboring nodes to update the representation of center node

W. L. Hamilton, R. Ying, J. Leskovec. Inductive Representation Learning on Large Graphs. NeuraIPS 2017

Spatial Methods for Graph CNN

- **GCN: Graph Convolution Network**
  - **Aggregating information from neighborhood via a normalized Laplacian matrix** 通过归一化拉普拉斯矩阵从邻域聚合信息
  - **Shared parameters are from feature transformation**
  - **A reduced version of ChebNet**

$$Z = f(X, A) = \mathrm{softmax}\Big(\hat{A}\ \mathrm{ReLU}\Big(\hat{A}X\boxed{W^{(0)}}\Big)\boxed{W^{(1)}}\Big)$$



Parameter for feature Transformation

W——特征变换

T. N. Kipf, and M. Welling. Semi-supervised classification with graph convolutional networks. ICLR 2017

Spatial Methods for Graph CNN

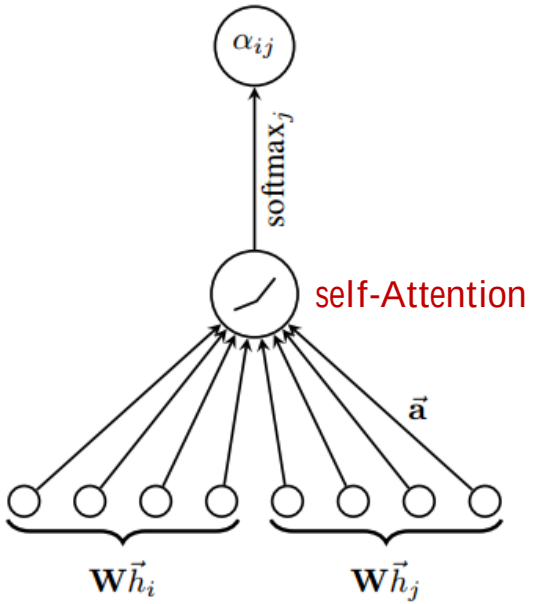
- **GAT: Graph Attention Network** GCN中参数的表达太弱
  - **Learning the aggregation matrix, i.e., Laplacian matrix in GCN, via attention mechanism** 用Attention学习聚合的矩阵权重
  - **Shared parameters contain two parts**
    - **Parameters for feature transformation**
    - **Parameters for attention**

Parameter for feature Transformation

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$

Parameter of Attention mechanism

类比卷积核



Attention Mechanism in GAT

P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio. Graph Attention Networks. NeuralPS 2018.

Spatial Methods for Graph CNN

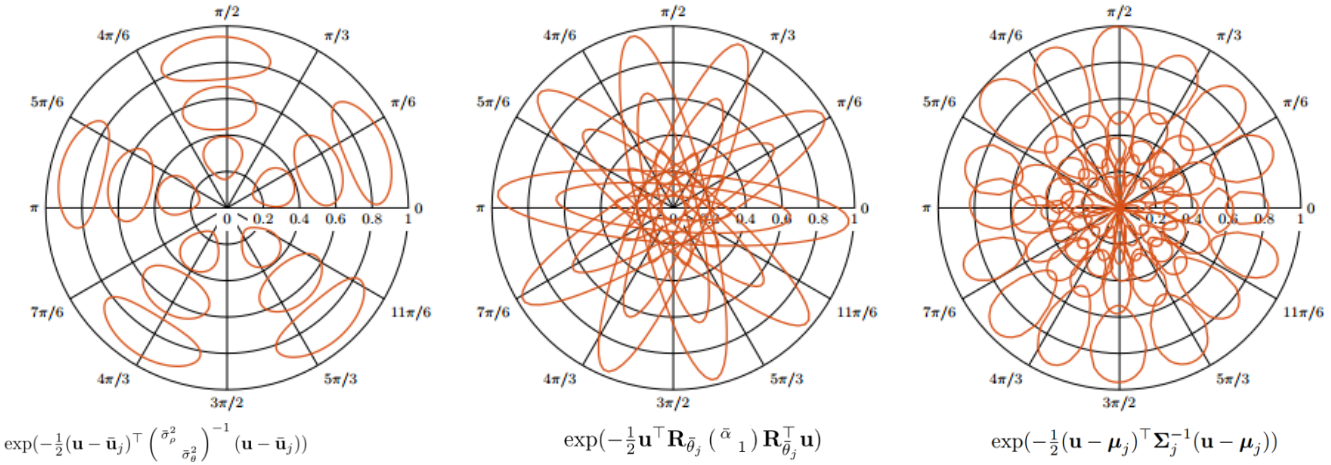
- **MoNet: A general framework for spatial methods**
  - **Define multiple kernel functions, parameterized or not, to measure the similarity between target node and other nodes**
  - **Convolution kernels are the weights of these kernel functions**

计算结点相似度的方式

$$(f \star g)(x) = \sum_{j=1}^{J} g_j D_j(x) f$$

Convolution kernel

多种核函数计算结果加权



F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, M. M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model CNNs. CVPR 2017.

Our method:

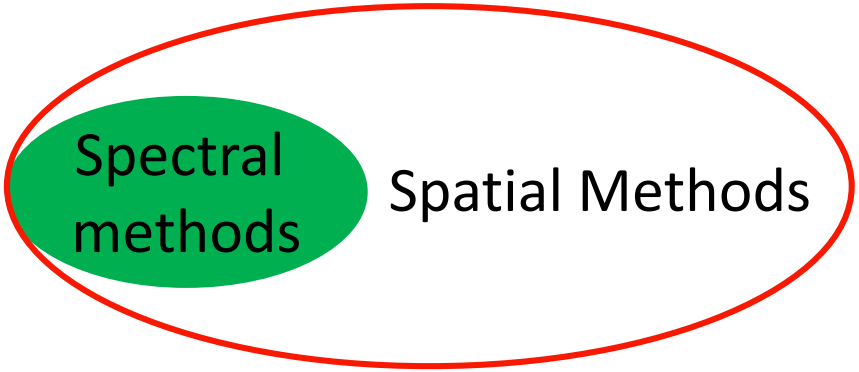
Graph Convolutional Networks using Heat Kernel for Semi-supervised Learning

(IJCAI 2019)

Bingbing Xu, Huawei Shen, Qi Cao, Keting Cen, Xueqi Cheng. Graph Convolutional Networks using Heat Kernel for Semi-supervised Learning, IJCAI 2019.

Spectral methods vs. Spatial methods

- **Connections**
  - **Spectral methods are special cases of spatial methods**



$$(f \star g)(x) = \sum_{j=1}^{J} g_j D_j(x) f$$

Kernel function: Characterizing the similarity or distance among nodes

- **Difference**
  - **Spectral methods define kernel functions via an explicit space transformation, i.e., projecting into spectral space**
  - **Spatial methods directly define kernel functions**

Spectral methods: Recap

- **Spectral CNN**

$$y = U g_\theta U^T x = (\theta_1 u_1 u_1^T + \theta_2 u_2 u_2^T + \cdots + \theta_n u_n u_n^T) x$$

- **ChebNet**

$$y = (\theta_0 I + \theta_1 L + \theta_2 L^2 + \cdots + \theta_{K-1} L^{K-1}) x$$

- **GCN**

$$y = \theta(I - L)x$$

Question:

Why GCN with less parameters performs better than ChebyNet?

Graph Signal Processing: filter

- **Smoothness of a signal x over graph is measured by**

$$x^T Lx = \sum_{(u,v)\in E} A_{uv}\left(\frac{x_u}{\sqrt{d_u}} - \frac{x_v}{\sqrt{d_v}}\right)^2$$

$\lambda_i = u_i^T L u_i$ can be viewed as the frequency of u_i

- **Basic filters**
  - $u_i u_i^T (1 \le i \le n)$ **are a set of basic filters**
  - **For a graph signal x, the basic filter $u_i u_i^T$ only allows the component with frequency λ_i passes**

$$x = \alpha_1 u_1 + \alpha_2 u_2 + \cdots + \alpha_n u_n,$$

$$u_i u_i^T x = \alpha_i u_i$$

Combined filters: High-pass vs. Low-pass

- **Combined filters**
  - **A linear combination of basic filters**

$$\theta_1 u_1 u_1^T + \theta_2 u_2 u_2^T + \cdots + \theta_n u_n u_n^T$$

  - **L^k is a combined filter with the coefficients $\{\lambda_i^k\}_{i=1}^n$**
  - **L^k assign high coefficients to basic filters with high-frequency, i.e., L^k is a high-pass filter**

- **GCN only consider $k = 0$ and $k = 1$, avoiding the boosting effect to basic filters with high-frequency**
  - **Behaving as a low-pass combined filter**
  - **Explaining why GCN performs better than ChebyNet**

Our method: GraphHeat

- **Low-pass combined filters**
  - $\{e^{-skL}\}$, **where** s **is scaling parameter, and k is order**
  - e^{-sL} **is heat kernel over graph, which defines the similarity among nodes via heat diffusion over graph**

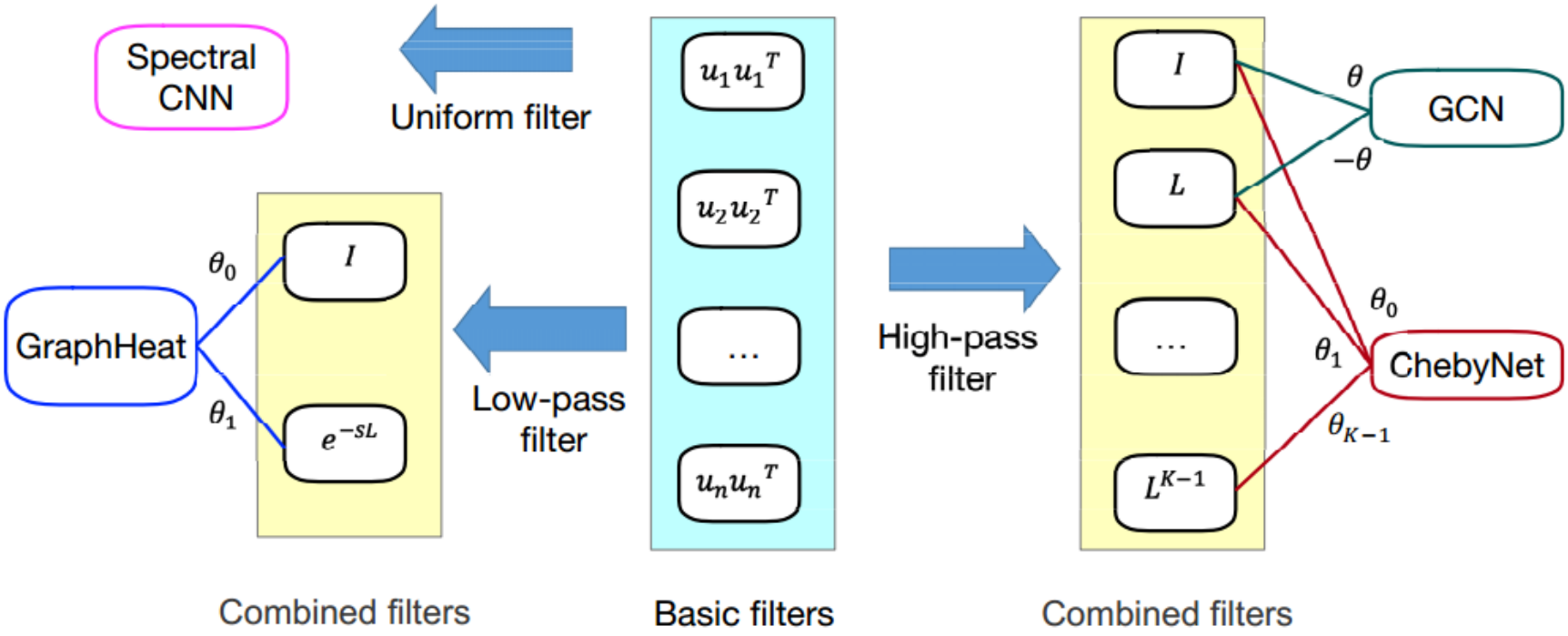
$$e^{-sL} = Ue^{-s\Lambda}U^T, \Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$$

  - **The basic filter** $u_i u_i^T (1 \leq i \leq n)$ **has the coefficient** $e^{-s\lambda_i}$**, suppressing signals with high-frequency**

B. Xu, H. Shen, Q. Cao, K. Cen, X. Cheng. Graph Convolutional Networks using Heat Kernel for Semi-supervised Learning, IJCAI 2019.

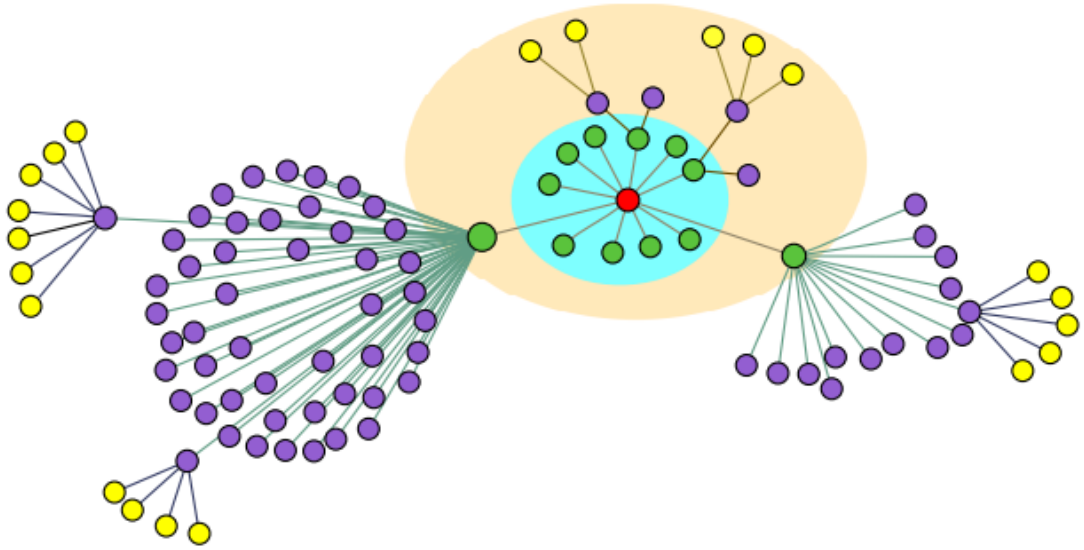
GraphHeat vs. baseline methods

- **Compared with baseline methods**



GraphHeat vs. baseline methods

- **Neighborhood**
  - **GCN and ChebNet determine neighborhood according to the hops away from center node, i.e., in an order-style**
    - **Nodes in different colors**
  - **GraphHeat determines neighborhood according to the similarity function by heat diffusion over graph**
    - **Nodes in different circles**



Experimental results

- **Results at the task of node classification**

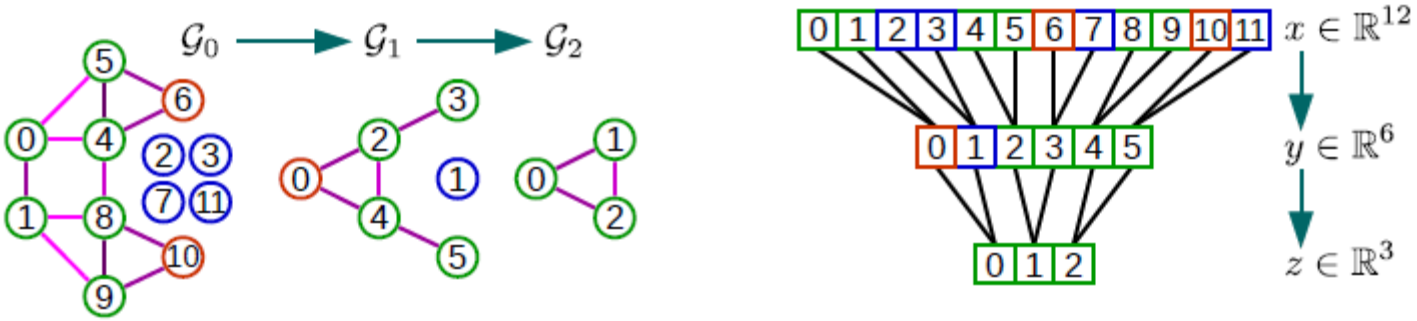
Method	Cora	Citeseer	Pubmed
MLP	55.1%	46.5%	71.4%
ManiReg	59.5%	60.1%	70.7%
SemiEmb	59.0%	59.6%	71.7%
LP	68.0%	45.3%	63.0%
DeepWalk	67.2%	43.2%	65.3%
ICA	75.1%	69.1%	73.9%
Planetoid	75.7%	64.7%	77.2%
ChebyNet	81.2%	69.8%	74.4%
GCN	81.5%	70.3%	79.0%
MoNet	81.7±0.5%	—	78.8±0.3%
GAT	83.0±0.7%	72.5±0.7%	79.0±0.3%
GraphHeat	**83.7%**	**72.5%**	**80.5%**

GraphHeat achieves state-of-the-art performance on the task of node classification on the three benchmark datasets

Graph Pooling

Graph Pooling via graph coarsening

- **Graph coarsening** 图粗化
  - **Merging nodes into clusters and take each cluster as a super node**

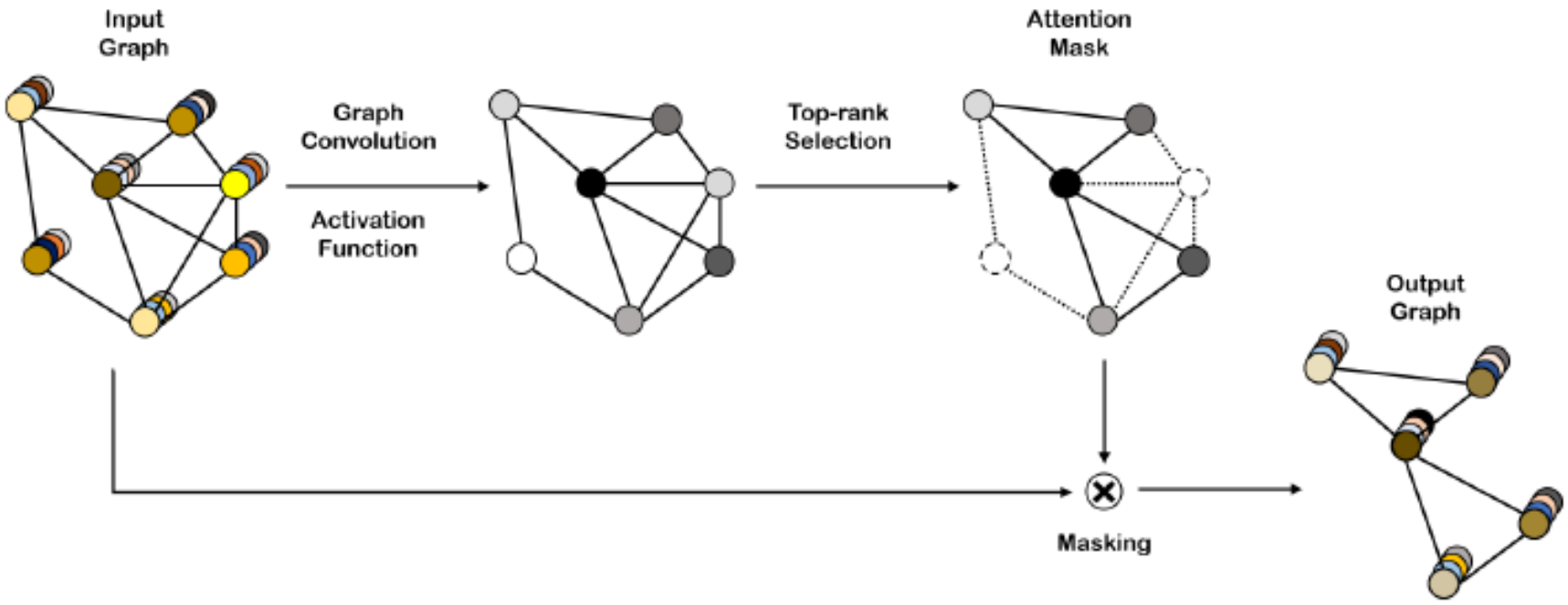


  - **Node merging could be done a priori or during the training process of graph convolutional neural networks, e.g, DiffPooling**

Ying, R., You, J., Morris, C., Ren, X., Hamilton, W. L., and Leskovec, J. Hierarchical graph representation learning with differentiable pooling, NeuraIPS 2018

Graph pooling via node selection

- **Node selection** 选择代表结点
  - **Learn a metric to quantity the importance of nodes and select several nodes according to the learned metric**



J. Lee, I. Lee, J. Kang. Self-attention Graph Pooling, ICML 2019.

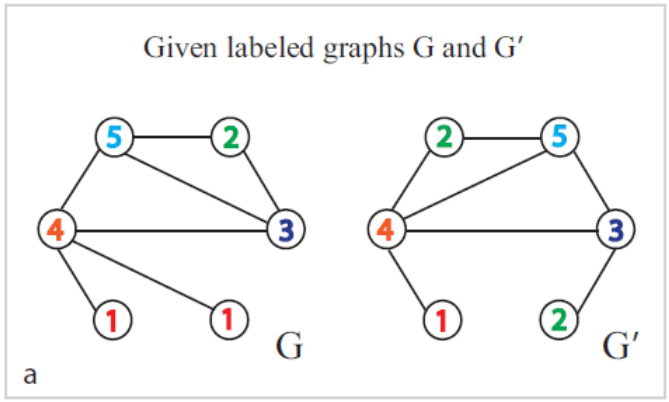
Discussions

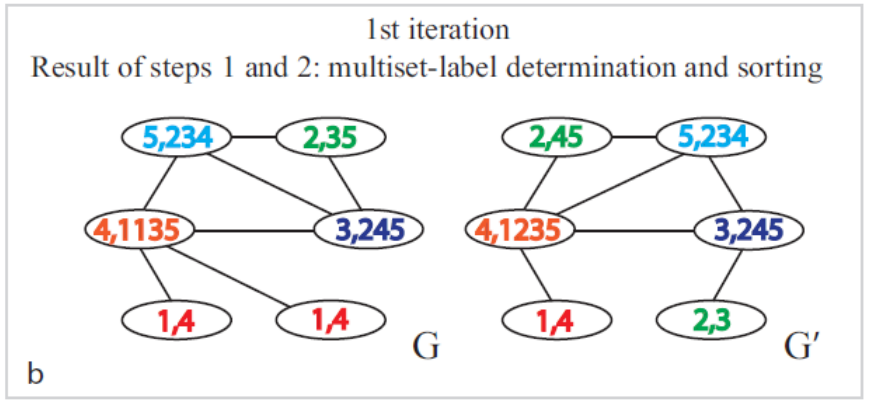
Question 1: Does structure matters?

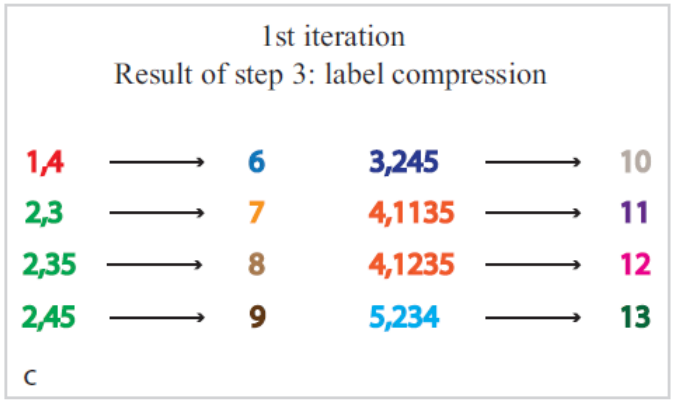
- **CNN learns stationary local patterns. How about graph CNN?** 目前以结构作为约束来做特征变换
  - **Both spectral methods and spatial methods fail to offer explicit clues or possibility to extract structural patterns**
  - **Instead, it seems that graph CNNs aim to learn the way in which features of neighboring nodes diffuse to the center node**
    - **Context representation**
  - **Explicitly correlate graph CNN with structural patterns, e.g., motif-based graph CNN, or graph CNN on heterogeneous networks**

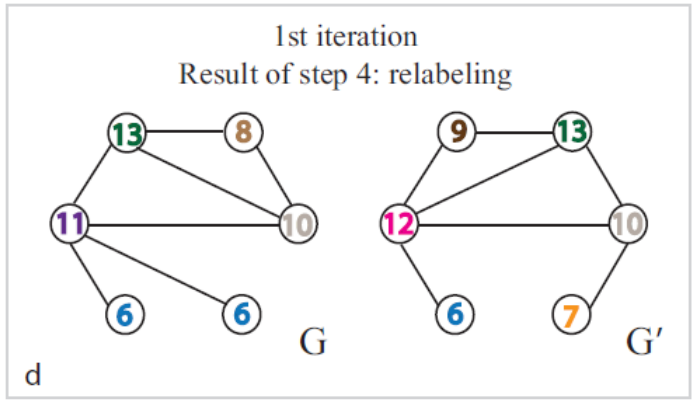
Question 2: Context representation?

- **Weisfeiler-Lehman isomorphism Test: WL Test**









Is graph CNN a soft version of WL test, working on networks with real-value node attributes instead of discrete labels?

K. Xu, W. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks? ICLR 2019.

Question 3: Future applications?

- **Three major scenarios**
  - **Node-level**
    - **Node classification: predict the label of nodes according to several labeled nodes and graph structure**
    - **Link prediction: predict the existence or occurrence of links among pairs of nodes**
  - **Graph-level**
    - **Graph classification: predict the label of graphs via learning a graph representation using graph CNN**
  - **Signal-level**
    - **Signal classification, similar to image classification which is signal-level scenario on a grid-like network**

Acknowledgement

Bingbing Xu

Qi Cao

Keting Cen

Yunqi Qiu

Xueqi Cheng

Thank you for your attentions!