CAPSULE GRAPH NEURAL NETWORK

ICLR 2019

Zhang Xinyi, Lihui Chen
School of Electrical and Electronic Engineering
Nanyang Technological University, Singapore
xinyi001@e.ntu.edu.sg, elhchen@ntu.edu.sg

https://github.com/benedekrozemberczki/CapsGNN

introduction

- GCN
  - CNN输入模型的局限
  - GCN中卷积的计算方式
  - GCN如何传导
- Capsule net
  - CNN的缺陷
  - 输入与CNN有何不同
  - cell之间如何传导
  - CapsNet architecture

introduction

Dynamic Routing Between Capsules

Sara Sabour **Nicholas Frosst**

Geoffrey E. Hinton
Google Brain
Toronto
{sasabour, frosst, geoffhinton}@google.com

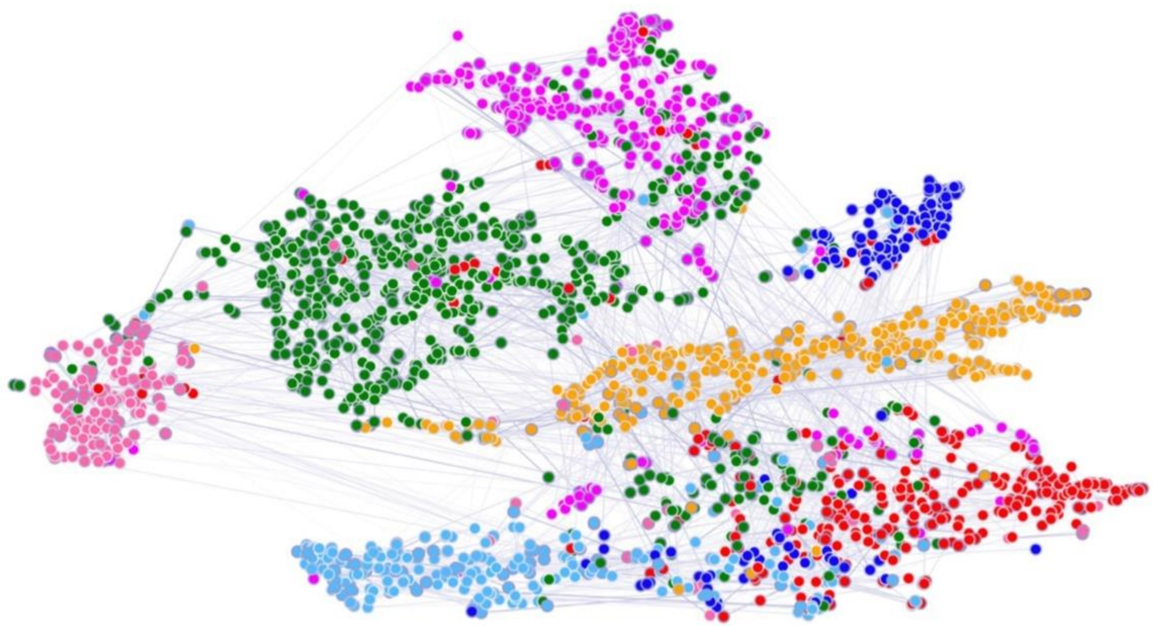
SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

Thomas N. Kipf
University of Amsterdam
T.N.Kipf@uva.nl

Max Welling
University of Amsterdam
Canadian Institute for Advanced Research (CIFAR)
M.Welling@uva.nl

introduction

- CNN输入模型的局限



Cora dataset

introduction

- GCN中卷积的计算方式
  - 欧拉公式 对于$\theta \in \mathbb{R}$，有$e^{i\theta} = cos\theta + isin\theta$。
  - 傅里叶级数 $f(x) = C + \sum_{n=1}^{\infty}\left(a_n cos(\frac{2\pi n}{T}x) + b_n sin(\frac{2\pi n}{T}x)\right), C \in \mathbb{R}$
  - 傅里叶变换 $\mathcal{F}\{f\}(v) = \int_{\mathbb{R}} f(x)e^{-2\pi i x \cdot v}dx$
  - 傅里叶逆变换 $\mathcal{F}^{-1}\{f\}(x) = \int_{\mathbb{R}} f(v)e^{2\pi i x \cdot v}dv$
  - 卷积公式 $(f * g)(t) = \int_{\mathbb{R}} f(x)g(t-x)dx$
  - 变换后的卷积 $f * g = \mathcal{F}^{-1}\{\mathcal{F}\{f\} \cdot \mathcal{F}\{g\}\}$

introduction

- GCN中卷积的计算方式
  - Laplacian算子

$$\Delta f(x) = \lim_{h \to 0} \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

  - 图中Laplacian算子

$$L = D - A$$

  - 标准化

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$$

  - 图中Laplacian算子分解

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

其中 Λ 是特征值组成的对角矩阵

$$U = [u_1 \ldots u_n]$$

  - 图中傅里叶变换

$$\mathcal{GF}\{f\}(\lambda_l) = \sum_{i=1}^{n} f(i) u_l^*(i) \qquad \mathcal{GF}\{x\} = U^T x$$

$$x = (f(1) \ldots f(n)) \in \mathbb{R}^n$$

  - 图中傅里叶逆变换

$$\mathcal{IGF}\{\hat{f}\}(i) = \sum_{l=0}^{n-1} \hat{f}(\lambda_l) u_l(i) \qquad \mathcal{IGF}\{x\} = Ux$$

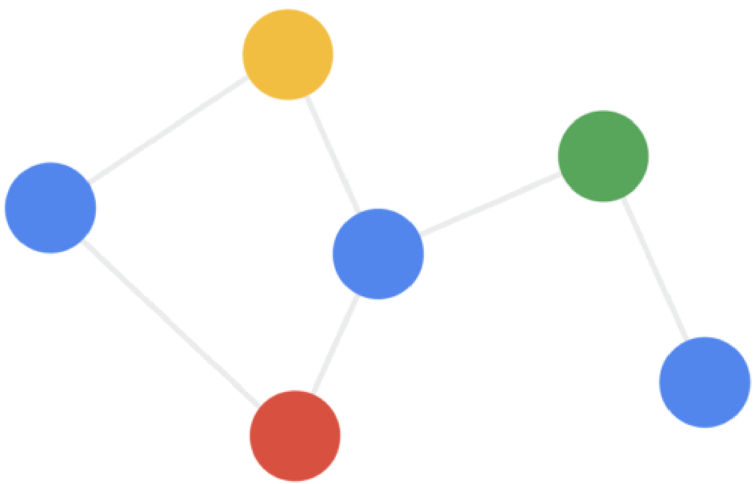
introduction

- GCN中卷积的计算方式
  - 图卷积公式 $g * x = U(U^T g \cdot U^T x)$
  - 利用Laplacian矩阵实现类似CNN的局部性
    定义g为Laplacian矩阵的函数g(L)
  - 改写后 $g_\theta * x = U g_\theta U^T x = U g_{\theta'}(\Lambda) U^T x$
  - 化简
    $$\begin{aligned} g_{\theta'} * x &\approx \theta(I_N + L)x \\ &= \theta(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}})x \end{aligned}$$
  - 加上激活函数 $H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)})$

introduction

- GCN如何传导

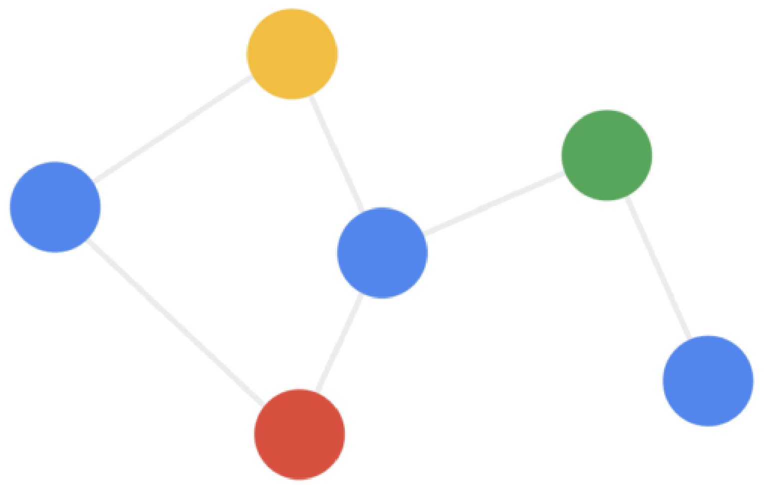
Step1 send



introduction

- GCN如何传导

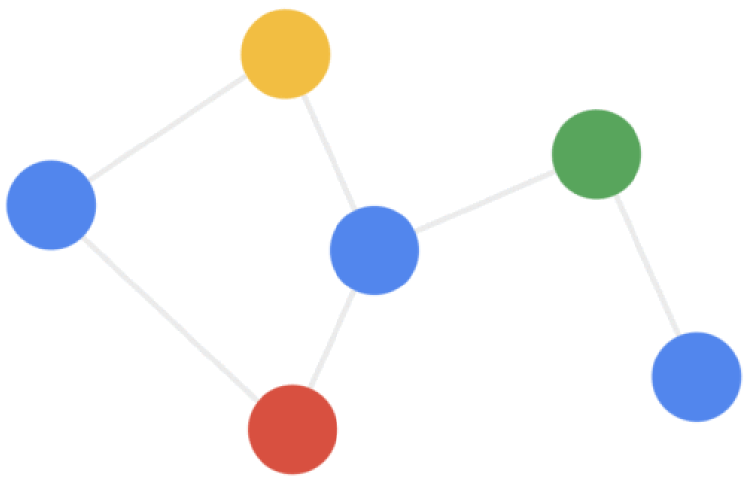
Step2 receive



introduction

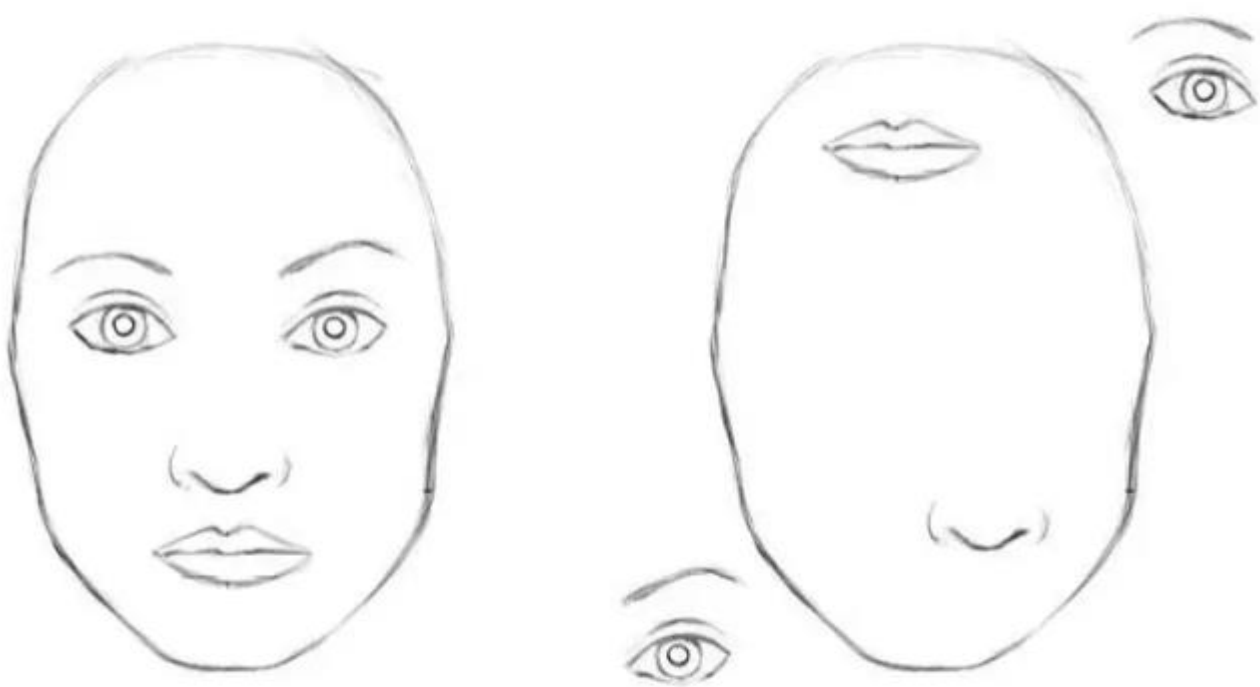
- GCN如何传导

Step3 transform



introduction

- CNN的缺陷



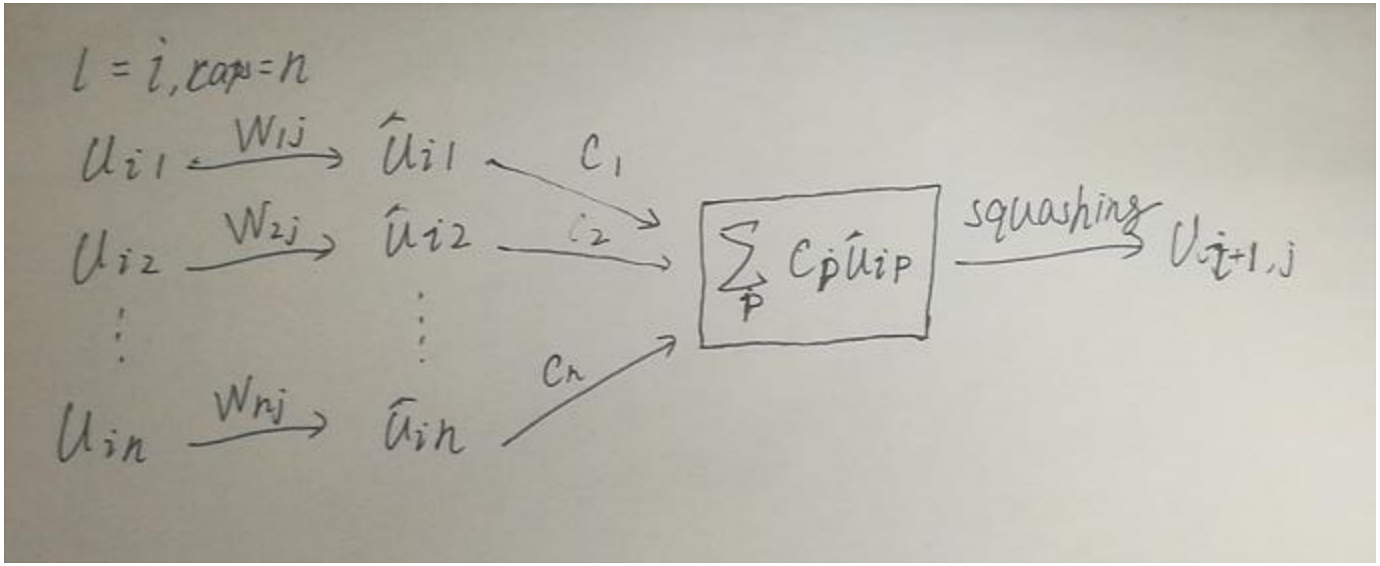
introduction

- 输入与CNN有何不同

Capsule vs. Traditional Neuron			
Input from low-level capsule/neuron		vector($\mathbf{u}_i$)	scalar(x_i)
Operation	Affine Transform	$\hat{\mathbf{u}}_{j\|i} = \mathbf{W}_{ij}\mathbf{u}_i$	–
	Weighting	$\mathbf{s}_j = \sum_i c_{ij}\hat{\mathbf{u}}_{j\|i}$	$a_j = \sum_i w_i x_i + b$
	Sum		
	Nonlinear Activation	$\mathbf{v}_j = \frac{\\|\mathbf{s}_j\\|^2}{1+\\|\mathbf{s}_j\\|^2}\frac{\mathbf{s}_j}{\\|\mathbf{s}_j\\|}$	$h_j = f(a_j)$
Output		vector($\mathbf{v}_j$)	scalar(h_j)

introduction

- cell之间如何传导



$$\mathbf{s}_j = \sum_i c_{ij} \hat{\mathbf{u}}_{j|i} \ , \qquad \hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij} \mathbf{u}_i$$

introduction

- 动态路由

Procedure 1 Routing algorithm.

procedure ROUTING($\hat{\boldsymbol{u}}_{j|i}$, r, l)
    for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow 0$.
    for r iterations **do**
        for all capsule i in layer l: $\mathbf{c}_i \leftarrow \texttt{softmax}(\mathbf{b}_i)$ ▷ `softmax` computes Eq. 3
        for all capsule j in layer $(l+1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$
        for all capsule j in layer $(l+1)$: $\mathbf{v}_j \leftarrow \texttt{squash}(\mathbf{s}_j)$ ▷ `squash` computes Eq. 1
        for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}.\mathbf{v}_j$
    return $\mathbf{v}_j$

softmax $$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

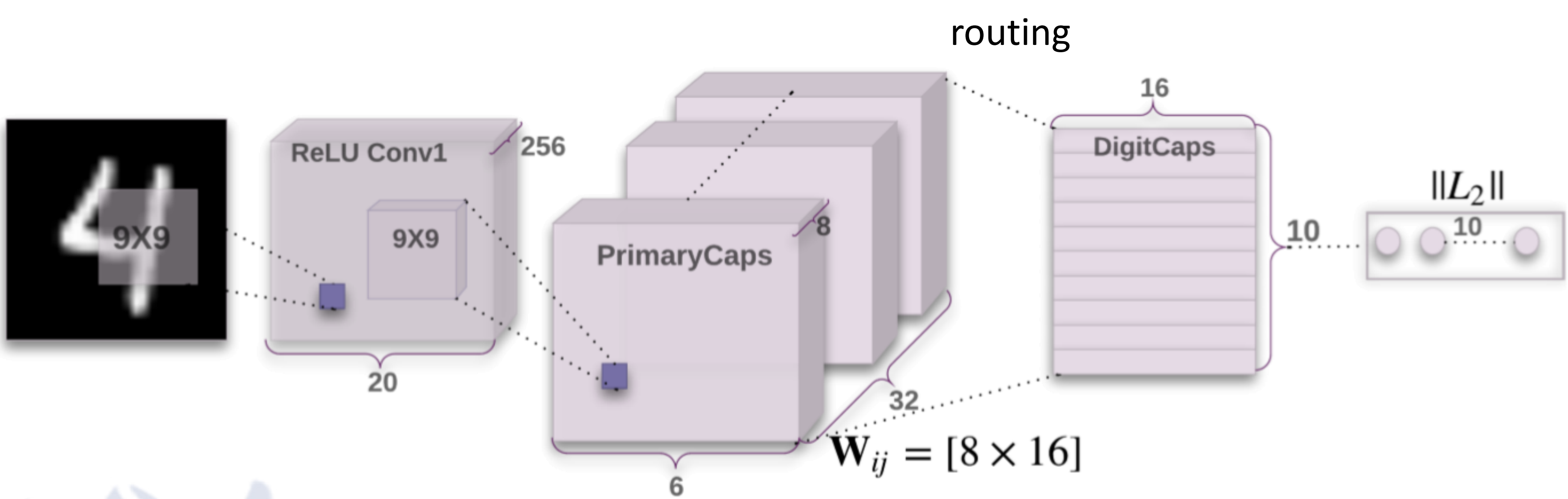
introduction

- Squash激活函数

$$\mathbf{v}_j = \underbrace{\boxed{\frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2}}}_{\text{additional “squashing”}} \underbrace{\boxed{\frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}}}_{\text{unit scaling}}$$

introduction

- CapsNet architecture



Caps num = 32
Kernel = 8×9×9×256 (conv2d)

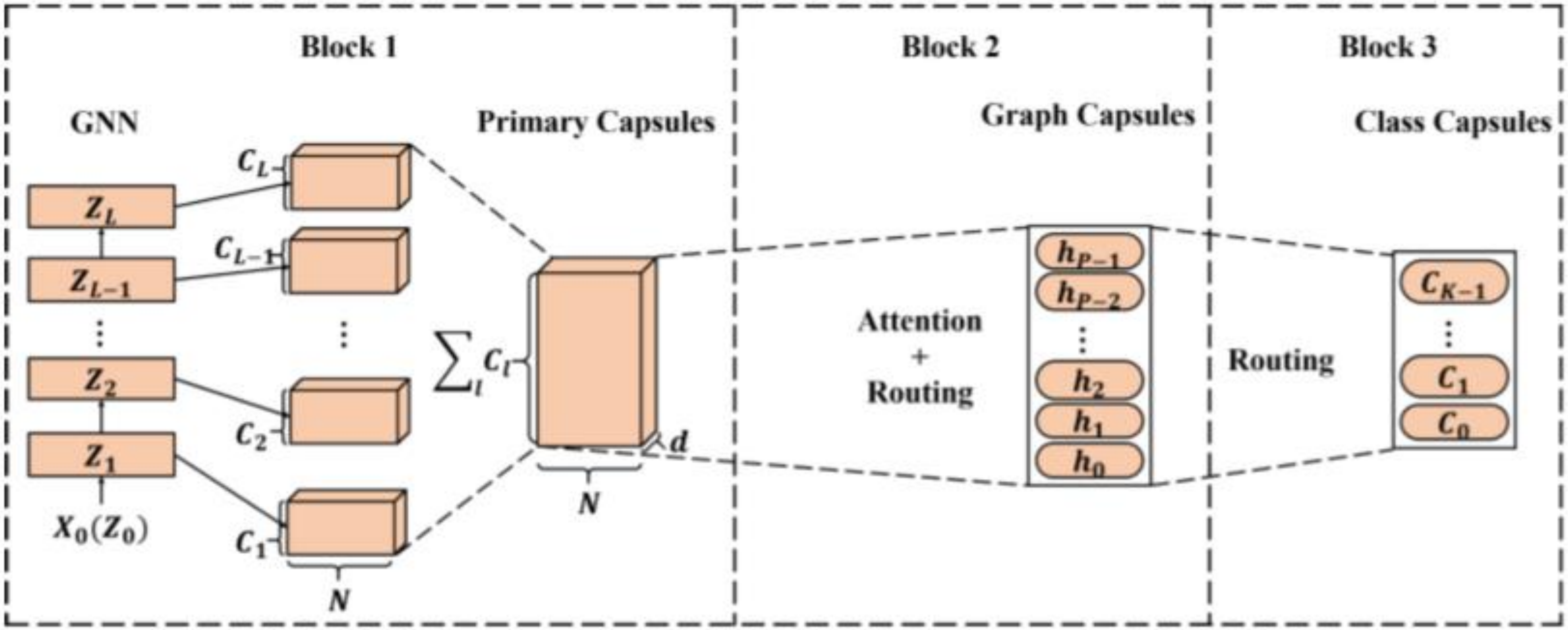
introduction

- CapsNet reconstruction result

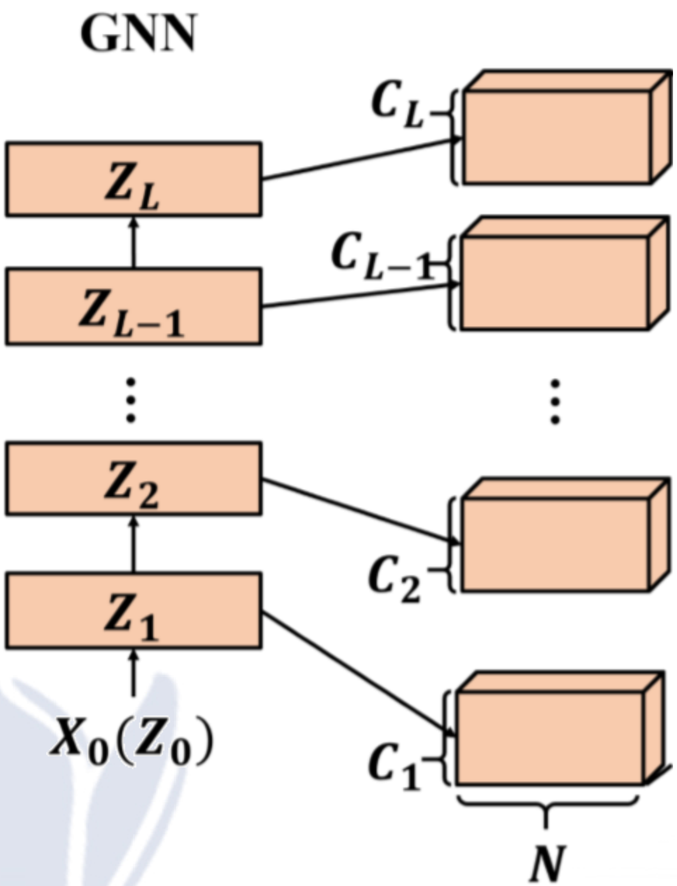
improvement

- Vector较scale能更有效的保存信息
- Routing能够保证没有信息丢失

Network


Block 1
Block 2
Block 3
GNN
Primary Capsules
Graph Capsules
Class Capsules
Z_L
Z_{L-1}
Z_2
Z_1
$X_0(Z_0)$
C_L
C_{L-1}
C_2
C_1
N
$\sum_l C_l$
d
N
Attention
+
Routing
h_{P-1}
h_{P-2}
h_2
h_1
h_0
Routing
C_{K-1}
C_1
C_0

Primary capsules

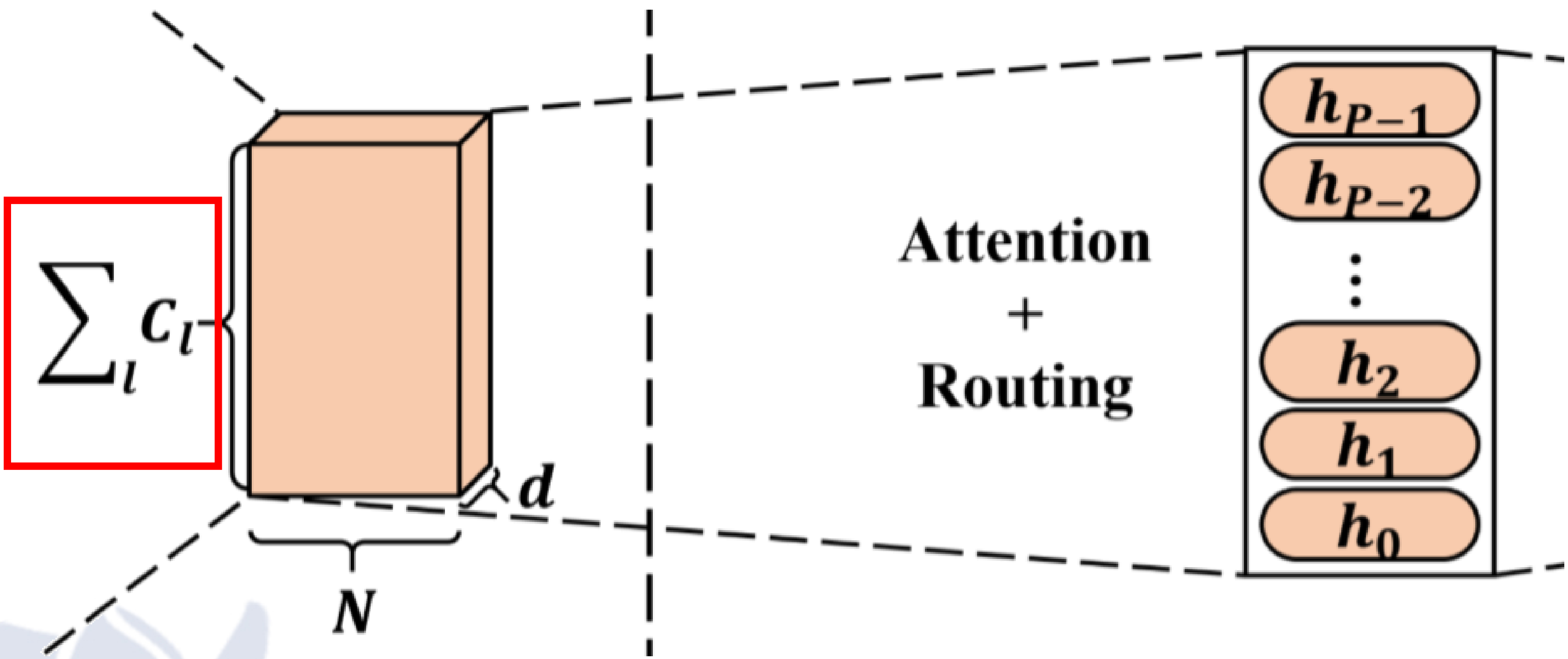


$$Z_j^{l+1} = f\left(\sum_i \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} Z_i^l W_{ij}^l\right)$$

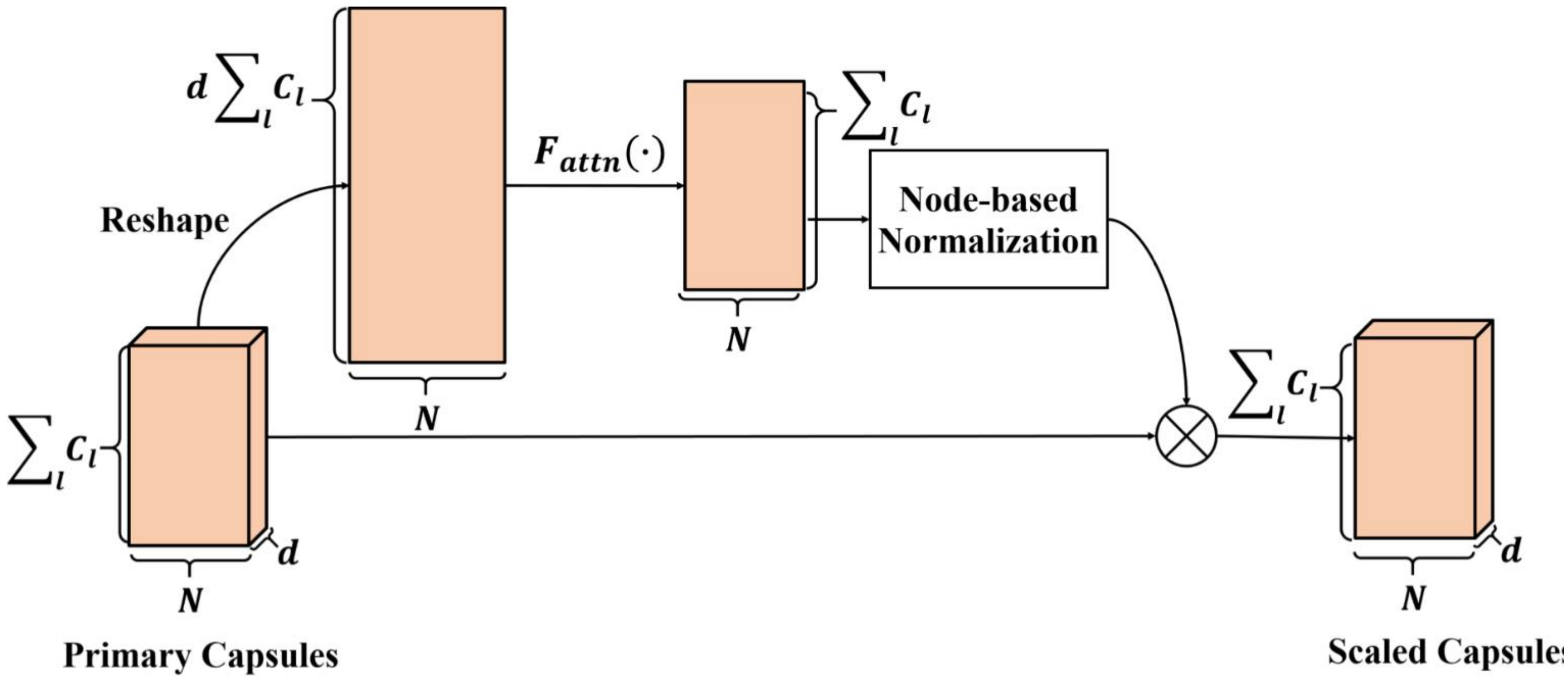
$$f(\cdot) = tanh(\cdot)$$

train $W_{ij}^l \in \mathbb{R}^{d \times d'}$

Graph capsules



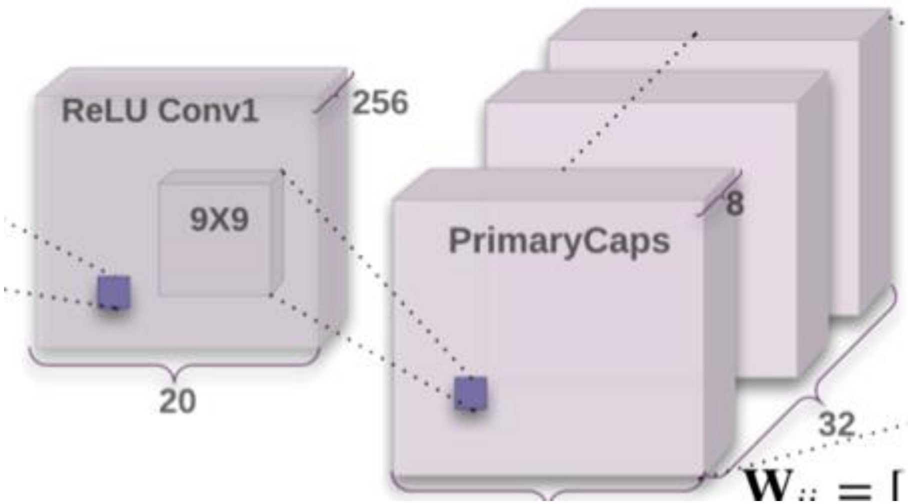
attention module



$$scaled(\boldsymbol{s}_{(n,i)}) = \frac{F_{attn}(\tilde{\boldsymbol{s}_n})_i}{\sum_n F_{attn}(\tilde{\boldsymbol{s}_n})_i}\boldsymbol{s}_{(n,i)}$$

$\tilde{\boldsymbol{s}_n} \in \mathbb{R}^{1\times C_{all}d}$ $\boldsymbol{s}_{(n,i)} \in \mathbb{R}^{1\times d}$ $F_{attn}(\tilde{\boldsymbol{s}_n}) \in \mathbb{R}^{1\times C_{all}}$

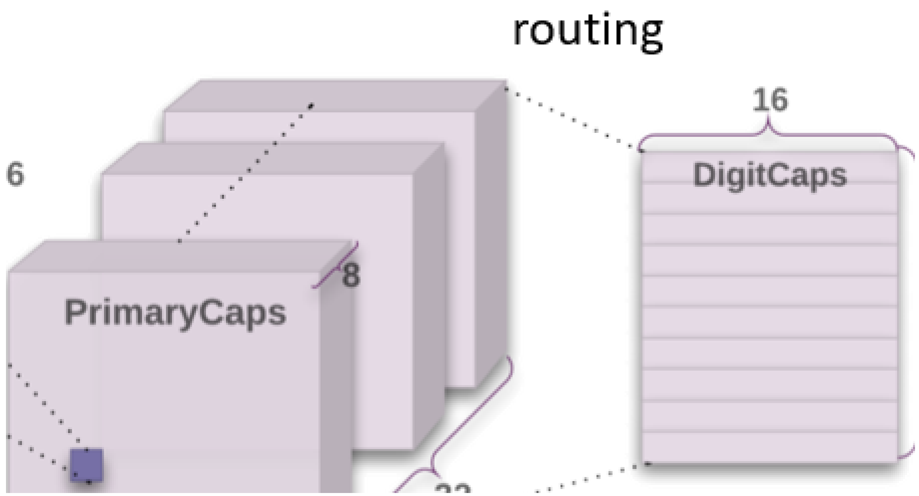
Calculate votes



Cpas num = P

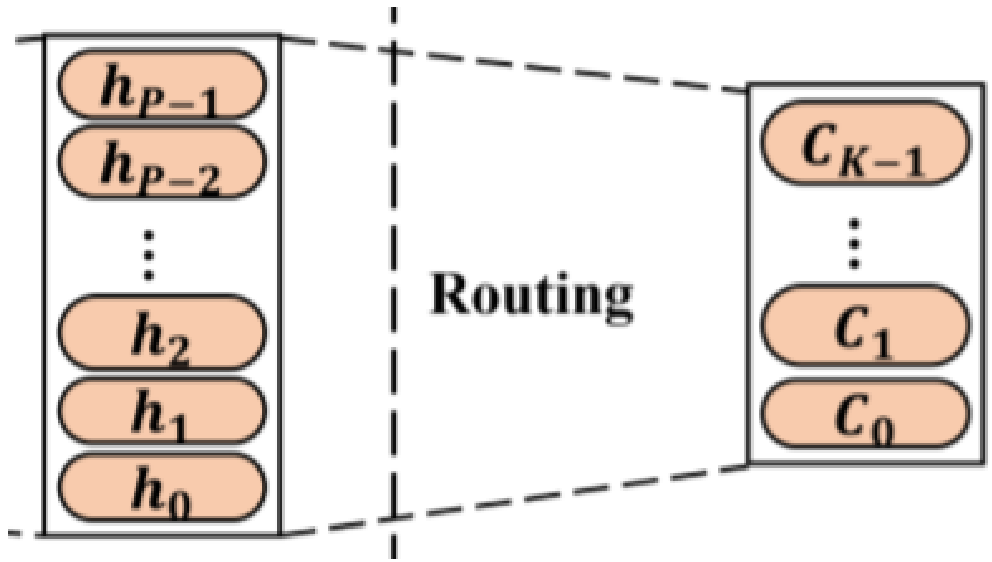
Kernel = d × 1 ×1× d

Dynamic Routing



Input: N × C_{all} × d × P = P × (NC_{all}d) (dim: 4 -> 2)
W : shape (P × P)
Out: shape (P × d')

Class Capsules



Class num = K

Input: P × d
W: P × K
Output： K × d

Classification Loss

$$Loss_c = \sum_k \{T_k \max(0, m^+ - \|\boldsymbol{c}_k\|)^2 + \lambda(1 - T_k) \max(0, \|\boldsymbol{c}_k\| - m^-)^2\}$$

同类时，分类概率必须大于m⁺　　异类时，分类的概率必须小于m⁻

use $\lambda = 0.5$

$m^+ = 0.9$, $m^- = 0.1$ and $T_k = 1$ iff the input graph belongs to class k.

Classification Experiment result

Table 1: Experiment Result of Biological Dataset

Algorithm	MUTAG	NCI1	PROTEINS	D&D	ENZYMES
WL	82.05±0.36	**82.19±0.18**	74.68±0.49	**79.78±0.36**	52.22±1.26
GK	81.58±2.11	62.49±0.27	71.67±0.55	78.45±0.26	32.70±1.20
RW	79.17±2.07	>3days	74.22±0.42	>3days	24.16±1.64
Graph2vec	83.15±9.25	73.22±1.81	73.30±2.05	-	-
AWE	**87.87±9.76**	-	-	71.51±4.02	35.77±5.93
DGK	87.44±2.72	80.31±0.46	75.68±0.54	73.50±1.01	53.43±0.91
PSCN	**88.95±4.37**	76.34±1.68	75.00±2.51	76.27±2.64	-
DGCNN	85.83±1.66	74.44±0.47	75.54±0.94	**79.37±0.94**	51.00±7.29
ECC	76.11	76.82	-	72.54	45.67
GCAPS-CNN	-	**82.72±2.38**	**76.40±4.17**	77.62±4.99	**61.83±5.39**
CapsGNN	86.67±6.88	78.35±1.55	**76.28±3.63**	75.38±4.17	**54.67±5.67**

Classification Experiment result

Table 2: Experiment Result of Social Dataset

Algorithm	COLLAB	IMDB-B	IMDB-M	RE-M5K	RE-M12K
WL	**79.02±1.77**	**73.40±4.63**	49.33±4.75	49.44±2.36	38.18±1.30
GK	72.84±0.28	65.87±0.98	43.89±0.38	41.01±0.17	31.82±0.08
DGK	73.09±0.25	66.96±0.56	44.55±0.52	41.27±0.18	32.22±0.10
AWE	73.93±1.94	**74.45±5.83**	**51.54±3.61**	**50.46±1.91**	39.20±2.09
PSCN	72.60±2.15	71.00±2.29	45.23±2.84	49.10±0.70	**41.32±0.42**
DGCNN	73.76±0.49	70.03±0.86	47.83±0.85	48.70±4.54	-
GCAPS-CNN	77.71±2.51	71.69±3.40	48.50±4.10	50.10±1.72	-
CapsGNN	**79.62±0.91**	73.10±4.83	**50.27±2.65**	**52.88±1.48**	**46.62±1.90**

欧拉公式推导

n阶泰勒公式

$$f(x)=\frac{f(x_0)}{0!}+\frac{f'(x_0)}{1!}(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+\ldots+\frac{f^{(n)}(x_0)}{n!}(x-x_0)^n+R_n(x)$$

$$e^x=1+x+\frac{1}{2!}x^2+\frac{1}{3!}x^3+\cdots$$

$$sin(x)=x-\frac{1}{3!}x^3+\frac{1}{5!}x^5+\cdots$$

$$cos(x)=1-\frac{1}{2!}x^2+\frac{1}{4!}x^4+\cdots$$ 将$x=i\theta$代入e

$$\begin{aligned}
e^{i\theta}&=1+i\theta+\frac{(i\theta)^2}{2!}+\frac{(i\theta)^3}{3!}+\frac{(i\theta)^4}{4!}+\frac{(i\theta)^5}{5!}+\frac{(i\theta)^6}{6!}+\frac{(i\theta)^7}{7!}+\frac{(i\theta)^8}{8!}+\cdots\\
&=1+i\theta-\frac{\theta^2}{2!}-\frac{i\theta^3}{3!}+\frac{\theta^4}{4!}+\frac{i\theta^5}{5!}-\frac{\theta^6}{6!}-\frac{i\theta^7}{7!}+\frac{\theta^8}{8!}+\cdots\\
&=\left(1-\frac{\theta^2}{2!}+\frac{\theta^4}{4!}-\frac{\theta^6}{6!}+\frac{\theta^8}{8!}-\cdots\right)+i\left(\theta-\frac{\theta^3}{3!}+\frac{\theta^5}{5!}-\frac{\theta^7}{7!}+\cdots\right)\\
&=\cos\theta+i\sin\theta
\end{aligned}$$

卷积傅里叶变换推导

$$h(z) = \int_{\mathbb{R}} f(x) g(z-x) dx$$

$$\begin{aligned}
\mathcal{F}\{f * g\}(v) &= \mathcal{F}\{h\}(v) \\
&= \int_{\mathbb{R}} h(z) e^{-2\pi i z \cdot v} dz \\
&= \int_{\mathbb{R}} \int_{\mathbb{R}} f(x) g(z-x) e^{-2\pi i z \cdot v} dx dz \\
&= \int_{\mathbb{R}} f(x) \left(\int_{\mathbb{R}} g(z-x) e^{-2\pi i z \cdot v} dz \right) dx
\end{aligned}$$

带入 $y = z - x$；$dy = dz$

$$\begin{aligned}
\mathcal{F}\{f * g\}(v) &= \int_{\mathbb{R}} f(x) \left(\int_{\mathbb{R}} g(y) e^{-2\pi i (y+x) \cdot v} dy \right) dx \\
&= \int_{\mathbb{R}} f(x) e^{-2\pi i x \cdot v} \left(\int_{\mathbb{R}} g(y) e^{-2\pi i y \cdot v} dy \right) dx \\
&= \int_{\mathbb{R}} f(x) e^{-2\pi i x \cdot v} dx \int_{\mathbb{R}} g(y) e^{-2\pi i y \cdot v} dy \\
&= \mathcal{F}\{f\}(v) \cdot \mathcal{F}\{g\}(v)
\end{aligned}$$

图卷积公式化简

化简前图卷积公式

$$g_\theta * x = U g_\theta U^T x = U g_{\theta'}(\Lambda) U^T x$$

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

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

$\lambda_{\max}$ denotes the largest eigenvalue of L

其中 T_k 是Chebyshev多项式。这里可以把简单 $g_\theta(\Lambda)$ 简单看成是 Λ 的多项式。

因为 $U \Lambda^k U^T = (U \Lambda U^T)^k = L^k$

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

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

设定 $K = 1$ 那卷积公式可以简化为

$$\begin{aligned} g_{\theta'} * x &\approx \theta(I_N + L)x \\ &= \theta(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}})x \end{aligned}$$