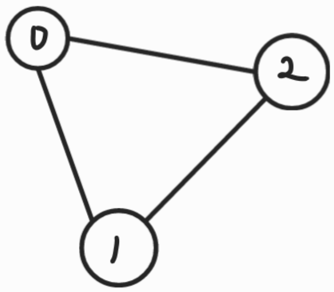
Graph (G)



$$A \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \qquad D \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

$$\therefore L = D - A \quad \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

$$Lx = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$= \begin{bmatrix} 2x_1 - x_2 - x_3 \\ -x_1 + 2x_2 - x_3 \\ -x_1 - x_2 + 2x_3 \end{bmatrix} = \begin{bmatrix} (x_1 - x_2) + (x_1 - x_3) \\ (x_2 - x_1) + (x_2 - x_3) \\ (x_3 - x_1) + (x_3 - x_2) \end{bmatrix} = \begin{bmatrix} \sum (x_1 - x_i) \\ \sum (x_2 - x_i) \\ \sum (x_3 - x_i) \end{bmatrix}$$

Convolution Theorem

$$H(z) = \int_R f(x)\, g(z-x)\, dx$$

$$\mathcal{F}\{f * g\}(v) = \mathcal{F}\{h\}(v)$$

$$= \int_R h(z)\, e^{-2\pi z v}\, dz$$

$$= \int_R \int_R f(x)\, g(z-x)\, e^{-2j\pi v z}\, dx\, dz$$

$$= \int_R f(x) \left(\int_R g(z-x)\, e^{-2j\pi v z}\, dz \right) dx \quad u = z - x$$

$$= \int_R f(x) \left(\int_R g(u)\, e^{-2\pi v (u+x)}\, du \right) dx$$

$$= \int_R f(x)\, e^{-2j\pi v x}\, dx \int_R g(u)\, e^{-2k\pi v u}\, du$$

⇒ convolution 연산은 Fourier Transform 의 multiplication 으로 표현 가능.

Graph domain 에서 Filter와 signal의 convolution의 의미는

node feature

Laplacian Matrix의 eigen decomposition 한 결과와 signal의 곱과 같다.

$$f * g = \mathcal{F}^{-1}\{\mathcal{F}\{f\} \cdot \mathcal{F}\{g\}\}$$

$$g_\theta * x = \mathcal{F}^{-1}(\mathcal{F}(x) \odot \mathcal{F}(g)) = U(U^T x \odot U^T g)$$

Laplacian.

$$x \longrightarrow U^T x \longrightarrow (g_\theta^*) U^T x \longrightarrow \boxed{(U) g_\theta^* U^T x}$$

Signal Graph Fourier Transform Filtering Inverse Graph Fourier Transform

$$\underbrace{g_\theta}_{\text{Filter}} * \underbrace{x}_{\text{signal}} = U g_\theta^* U^T x$$

$$g_\theta = \begin{bmatrix} \hat{g}(\lambda_1) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \hat{g}(\lambda_N) \end{bmatrix}$$

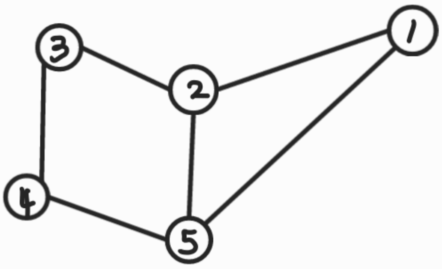
$$g_\theta(\Lambda) = \sum_{k=0}^{K} \theta_k \Lambda^k = \theta_0 \Lambda^0 + \theta_1 \Lambda^1 + \cdots + \theta_K \Lambda^K$$

$$x * g_\theta = U \sum_{k=0}^{K} \theta_k \Lambda^k U^T x$$

$$= U(\theta_0 \Lambda^0 + \theta_1 \Lambda^1 + \cdots + \theta_k \Lambda^K) U^T x$$

$$= \sum_{k=0}^{K} \theta_k L^k x$$

Localized Feature (k-hop)



$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

$$L = D - A \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 3 & -1 & 0 & 1 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix}$$

$$Lx = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 3 & -1 & 0 & 1 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{bmatrix} = \begin{bmatrix} \sum_{i \in [2.3]} (X_1 - X_i) \\ \sum_{i \in [1.3.5]} (X_2 - X_i) \\ \sum_{i \in [2.4]} (X_3 - X_i) \\ \sum_{i \in [3.5]} (X_4 - X_i) \\ \sum_{i \in [1.2.4]} (X_5 - X_i) \end{bmatrix}$$

1 hop

$$L(Lx) = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 3 & -1 & 0 & 1 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix} \begin{bmatrix} \sum_{i\in[2,3]}(x_1-x_i) \\ \sum_{i\in[1,3,5]}(x_2-x_i) \\ \sum_{i\in[2,4]}(x_3-x_i) \\ \sum_{i\in[3,5]}(x_4-x_i) \\ \sum_{i\in[1,2,4]}(x_5-x_i) \end{bmatrix}$$

2 hop

$$= \begin{bmatrix} 2\sum_{i\in[2,3]}(x_1-x_i) - \sum_{i\in[1,3,5]}(x_2-x_i) - \sum_{i\in[1,2,4]}(x_5-x_i) \\ -\sum_{i\in[2,3]}(x_1-x_i) + 3\sum_{i\in[2,3]}(x_1-x_i) - \sum_{i\in[2,4]}(x_3-x_i) + 3\sum_{i\in[1,2,4]}(x_5-x_i) \\ -\sum_{i\in[1,3,5]}(x_2-x_i) + 2\sum_{i\in[1,3,5]}(x_2-x_i) - \sum_{i\in[3,5]}(x_4-x_i) \\ -\sum_{i\in[2,4]}(x_3-x_i) + 2\sum_{i\in[3,5]}(x_4-x_i) - \sum_{i\in[1,2,4]}(x_5-x_i) \\ -\sum_{i\in[2,3]}(x_1-x_i) - \sum_{i\in[1,3,5]}(x_2-x_i) - \sum_{i\in[3,5]}(x_4-x_i) + 3\sum_{i\in[1,2,4]}(x_5-x_i) \end{bmatrix}$$

첫번째 행만 보면

$$\underbrace{2\sum_{i\in[2,3]}(x_1-x_i)}_{\text{node 1과 1 이웃의 정보.}} - \underbrace{\sum_{i\in[1,3,5]}(x_2-x_i)}_{\text{node 2와 1 이웃의 정보}} - \underbrace{\sum_{i\in[1,2,4]}(x_5-x_i)}_{\text{node 5와 1 이웃의 정보}}$$

→ central node 의 이웃의 이웃까지 feature 활용

$L^k x \Rightarrow$ k-hop 의 이웃정보 포함.

ChebNet

parameterization을 통해 Localized feature 추출 가능.

But computation complexity가 높음.

⇒ Chebyshev polynomial of first kind로 근사

$$g_\theta = \sum_{i=0}^{K} \theta_i T_i(\tilde{\Lambda}), \quad \tilde{\Lambda} = \frac{2\Lambda}{\lambda_{max}} - I_n \quad \underbrace{(-1 < \tilde{\Lambda} < 1)}_{\text{범위 scaling}}$$

$$\underbrace{T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)}_{\text{Chebyshev polynomial of first kind}} \quad (T_0(x) = 1,\ T_1(x) = x)$$

$$x * g_\theta = U\left(\sum_{i=0}^{K} \theta_i T_i(\tilde{\Lambda})\right) U^T x \qquad \tilde{L} = \frac{2L}{\lambda_{max}} - I_n$$

$$x * g_\theta = \sum_{i=0}^{K} \theta_i T_i(\tilde{L}) x$$

GCN

ChebNet에서 크게 3단계의 변화를 줌.

Step 1 : $K=1$, $\lambda_{max}=2$

Step 2 : $\theta = \theta_0 = -\theta_1$

Step 3 : $I_n + D^{-1/2} A D^{-1/2} \rightarrow \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2}$

Step 1

$K=1$ 로 정하고 activation function을 통해 non-linearity를 추가해 layer를 깊게 쌓는 방식으로 만듦. $\lambda_{max}=2$ 로 가정.

$$x * g_\theta = \sum_{i=0}^{K} \theta_i T_i(\mathcal{L}) x$$

$$\Rightarrow \boxed{x * g_\theta = \theta_0 x - \theta_1 (L - I_n) x}$$

Step 2

$\theta = \theta_0 = -\theta_1$ 와 같이 parameter를 줄여서 overfitting을 방지

$$\boxed{x * g_\theta = \theta \left(I_n + D^{-1/2} A D^{-1/2} \right) x}$$

$\boxed{\text{Step3}}$

$I_n + D^{-1/2} A D^{-1/2}$ ⇝ eigenvalue $[0, 2]$ 범위를 가짐

Renormalization trick을 통해 eigenvalue의 최댓값의 크기를 줄여줌.

$$I_n + D^{-1/2} A D^{-1/2} \longrightarrow \tilde{D}^{-1/2} \tilde{A}^{-1/2} \tilde{D}^{-1/2}$$

$$\left(\tilde{A} = A + I_n \ , \ \tilde{D}_{ii} = \sum_j \hat{A}_{ij} \right)$$

Self-loop를 추가하여 oversmoothing 방지.

Eigenvalue의 범위를 줄여서 연산과정에서의 stability 유임.

$$x * g_\theta = \theta \tilde{D}^{-1/2} \tilde{A}^{-1/2} \tilde{D}^{-1/2}$$

↓ 일반화

$$H^k = \sigma (\hat{A} H^{k-1} \theta^k)$$

$\hat{A} = \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2}$, $(R^{n \times n})$

$H^0 = X$, $(X \in R^{n \times C})$ — channel 수

$H^k \in R^{n \times F}$ — output channel 수 ⇒ k번째 layer output

$\theta^k \in R^{C \times F}$ ⇒ k번째 layer의 가중치 행렬.

σ ⇒ activation function.

Example

Class 의 수가 5개인 node classification.

$K=3$, $F_1 = 30$, $F_2 = 20$, $\sigma = ReLU$

입력값 : 총 30개의 노드, 40개의 feature $X \in R^{30 \times 40}$

$$H^{(0)} = X \in R^{30 \times 5}$$

$$H^{(1)} = ReLU\ (\hat{A} H^{(0)} \Theta^{(1)}) \in R^{30 \times 30}$$

$$H^{(2)} = ReLU\ (\hat{A} H^{(1)} \Theta^{(2)}) \in R^{30 \times 20}$$

$$H^{(3)} = softmax(\hat{A} H^{(2)} \Theta^{(3)}) \in R^{30 \times 5}$$

mini batch 방식으로 학습 불가. ∴ data가 큰 경우 학습이 쉽지 않음.

1. Normalized Adjacency Matrix 의 eigenvalue는 [0, 2] 이다.

(1) A의 가장 큰 eigenvalue 범위.

A가 undirected graph(G) 의 adjacency matrix 이고 λ_1이 A의 가장 큰 eigenvalue라고 하자. 이때, 아래 식이 성립한다.

$$d_{avg} \le \lambda_1 \le d_{max}$$

d_{avg} : G의 평균 degree.

d_{max} : G의 최대 degree

[증명 1]

V_1 (eigen vector) $\longleftrightarrow$ λ_1 (eigenvalue)

j번째 노드는 모든 i에 대해 $V_1(j) \ge V_1(i)$가 성립한다고 하자.

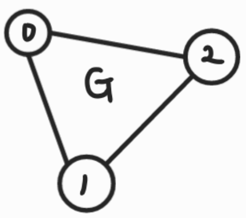
$$\lambda_1 V_1(j) = (AV_1)(j)$$

$$= \sum_{(i,j)\in E(G)} V_1(i) \le \sum_{(i,j)\in E(G)} V_1(j)$$

$$= deg(j)\, V_1(j) \le d_{max} V_1(j)$$

About $\sum_{(i,j)\in E(G)} V_1(i) \le \sum_{(i,j)\in E(G)} V_1(j)$

G로부터 계산한 A의 첫번째 eigenvector.



$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \quad V_1 = \begin{bmatrix} V_1(1) \\ V_2(2) \\ V_3(3) \end{bmatrix}$$

if $j==2$: $V_1(2) \ge V_1(i)$, $i \in [1,3]$

$$\sum_{(i,j)\in G} V_1(i) = 1\times V_1(1) + 1 \times V_1(3)$$

$$\sum_{(i,j)\in G} V_1(i) \le \sum_{(i,j)\in E(G)} V_1(2)$$

[증명 2]

모든 벡터의 원소가 1인 $\vec{x}$에 대해 A에 대한 레일리 몫(Rayleigh quotient)를 적용하면 다음이 성립한다.

$$\lambda_1 = \max_x \frac{x^T A x}{x^T x} \geq \frac{1^T A 1}{1^T 1}$$

$$= \frac{\sum_{(i,j)\in E(G)} A(i,j)}{n}$$

$$= \frac{\sum_i d(i)}{n}$$

$$= d_{avg}$$

2. Normalized A의 eigenvalue 범위는 [0,2] 이다.

normalized A : $\tilde{A} = D^{-1/2} A D^{-1/2}$

$\tilde{L} = I - \tilde{A}$

$$\tilde{L} = I - \tilde{A}$$
$$= D^{-1/2}(D-A)D^{-1/2}$$
$$= D^{-1/2} L D^{-1/2}$$

A의 eigenvalue 범위 : [0,2]

$\tilde{A}$ → normalized는 가장 큰 eigenvalue의 값을 1로 만들겠다는 뜻.

$\tilde{A}$의 eigenvalue : $\alpha_1 \geq \cdots \alpha_n$	$1 = \alpha_1 \geq \cdots \geq \alpha_n \geq -1$
$\tilde{L}$의 〃 : $\lambda_1 \leq \cdots \leq \lambda_n$	$0 = \lambda_1 \leq \cdots \lambda_n \leq 2$

$x = D^{-1/2}v$ $\rightsquigarrow$ $\tilde{L}$의 eigenvalue가 0이 될 수 있음.

($\vec{v}$는 L의 eigenvector)

$$\tilde{L}(D^{1/2}v) = D^{-1/2} L \cancel{D^{-1/2}} \cdot \cancel{D^{1/2}} v$$

$$= D^{-1/2} v$$

$$= 0$$

v는 0의 값을 갖는 eigen value 에 대응하는 L의 eigenvector에서 $D^{1/2}v$ 에 해당하는 $\tilde{L}$의 eigenvalue 는 0이다. 이 값이 가장 작은 값임을 보이기 위해 $\tilde{L}$가 positive semi-definite 임을 보여야 한다. positive semi definite는 모든 $x \in R^n$에 대해 $x^T \tilde{L} x$ 의 결과가 항상 0보다 크거나 같은 값을 내는것을 말한다.

$$x^T \tilde{L} x = x^T (I - \tilde{A}) x$$

$$= \sum_{i \in V} x(i)^2 - \sum_{(i,j)\in E} \frac{2x(i)x(j)}{\sqrt{d(i)d(j)}}$$

$$= \sum_{(i,j)\in E} \left(\frac{x(i)}{\sqrt{d(i)}} - \frac{x(j)}{\sqrt{d(j)}} \right)^2 \geq 0$$

$\Rightarrow$ $\tilde{L}$가 positive semi-definite 이고 $\lambda_1 = 0$ 임을 보였다.

$\alpha_1 \le 1$ 임을 보이기 위해 positive semi-definite 인 $\tilde{L}$ 을 활용하면 모든

$x \in \mathbb{R}^n$ 에 대해 다음이 성립한다.

$$x^T(1-\tilde{A})x \geq 0 \rightarrow x^Tx - x^T\tilde{A}x \geq 0 \rightarrow 1 \geq \frac{x^T\tilde{A}x}{x^Tx}$$

$$\therefore 1 \geq \alpha_1$$

레일리 몫을 통해 α_1의 상한선을 나타낼 수 있다. 위 식은 $x = D^{1/2}v$을 다시

대입해보면 앞서 언급한 내용을 활용해서 $\tilde{A}$의 max eigenvalue가 1임을 알 수 있다.

$$x^T\tilde{L}x = 0 \rightarrow x^T(1-\tilde{A})x = 0 \rightarrow x^Tx = x^T\tilde{A}x$$

이와 비슷하게 α_n의 하한선을 구하기 위해 positive semi-definite 인 $I+\tilde{A}$를 적용하면

$$x^T(I+\tilde{A})x \geq 0 \rightarrow x^Tx + x^T\tilde{A}x \geq 0 \rightarrow \frac{x^T\tilde{A}x}{x^Tx} \geq -1$$

$$\therefore \alpha_n \geq -1$$

마지막으로 $x^T(I+\tilde{A})x \geq 0$ 을 $\tilde{L}$ 에 대해 나타내면 레일리 몫을 통해 λ_n의

상한선에 대해 나타낼 수 있다.

$$-x^T\tilde{A}x \leq x^Tx \rightarrow x^TIx - x^TAx \leq 2x^Tx \rightarrow \frac{x^T\tilde{L}x}{x^Tx} \leq 2$$

$$\therefore \lambda_n \leq 2$$

2. Renormalization Trick.

$$I_n + D^{-1/2} A D^{-1/2} \rightsquigarrow S_1\text{-order}.$$

$$S_1\text{-order} = 2I - \tilde{L}$$

S_1^K-order 는 filter coefficient가 $(2-\lambda_i)^K$인 값을 가진다. 여기서 λ_i는 $\tilde{L}$의 eigenvalue 이다.

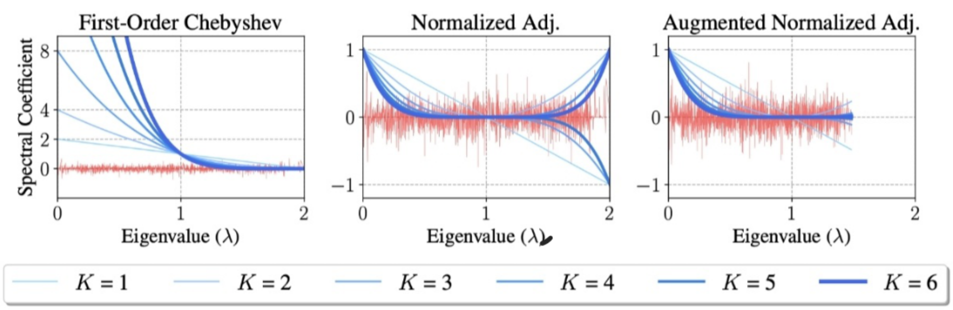


Figure 2. Feature (red) and filters (blue) spectral coefficients for different propagation matrices on Cora dataset (3rd feature).

그림 5. Simplifying GCN에서 실험한 augmented normalized adjacency의 효과

$\lambda_i < 1$ 에서 K가 커짐에 따라 filter coefficient 값이 급격히 커진다. 이를 방지하기 위해 renormalization trick을 써서 eigenvalue의 크기를 줄인다.

$$\tilde{S}_{adj} = \tilde{D}^{-1/2} \hat{A} \tilde{D}^{-1/2} \quad (\hat{A} = A + I,\ \tilde{D} = D + I)$$

$$\tilde{L} = I - \tilde{D}^{-1/2} \hat{A} \tilde{D}^{-1/2}$$

이때 renormalization trick을 적용한 filter coefficient는

$\hat{g}(\tilde{\lambda}_i) = (1-\tilde{\lambda}_i)^K$ 이다. ($\tilde{\lambda}_i$는 $\tilde{L}$의 eigen value)

Self-loop를 추가하게 되면 최대 eigen value 값이 줄어든다.

[Theorem]

A가 undirected graph G에 대한 adjacency matrix이고 이에 대한 degree matrix는 D라고 하자. 이 때 augmented adjacency matrix를 $\tilde{A} = A + \gamma I$ $\gamma > 0$ 으로 나타내고 이에 대한 degree matrix는 $\tilde{D}$ 이다. 또한 $\tilde{L} = I - D^{-1/2} A D^{-1/2}$ 의 가장 작은 eigenvalue와 가장 큰 eigenvalue를 각각 λ_1, λ_n 이라고 쓴다. 이때 다음 식이 성립한다.

$$0 = \lambda_1 = \tilde{\lambda}_1 < \tilde{\lambda}_n < \lambda_n$$