# A Restricted Black-box Adversarial Framework Towards Attacking Graph Embedding Models

Members：朱偉綸、黃秉茂、許齊崴

# Abstract

- Graph Embedding Model can effectively transform Node information, but is vulnerable to specific adversarial attacks
- Many researches now focus on White-box Attack, which can learn information such as model, model parameters, gradient direction, etc.
- This research proposes GF-Attack to attack Graph Embedding Model from the perspective of Graph Signal Processing
- In order to examine the generalization of the attack, this article will focus on model verification of GCN, SGN, DeepWalk, LINE, etc.

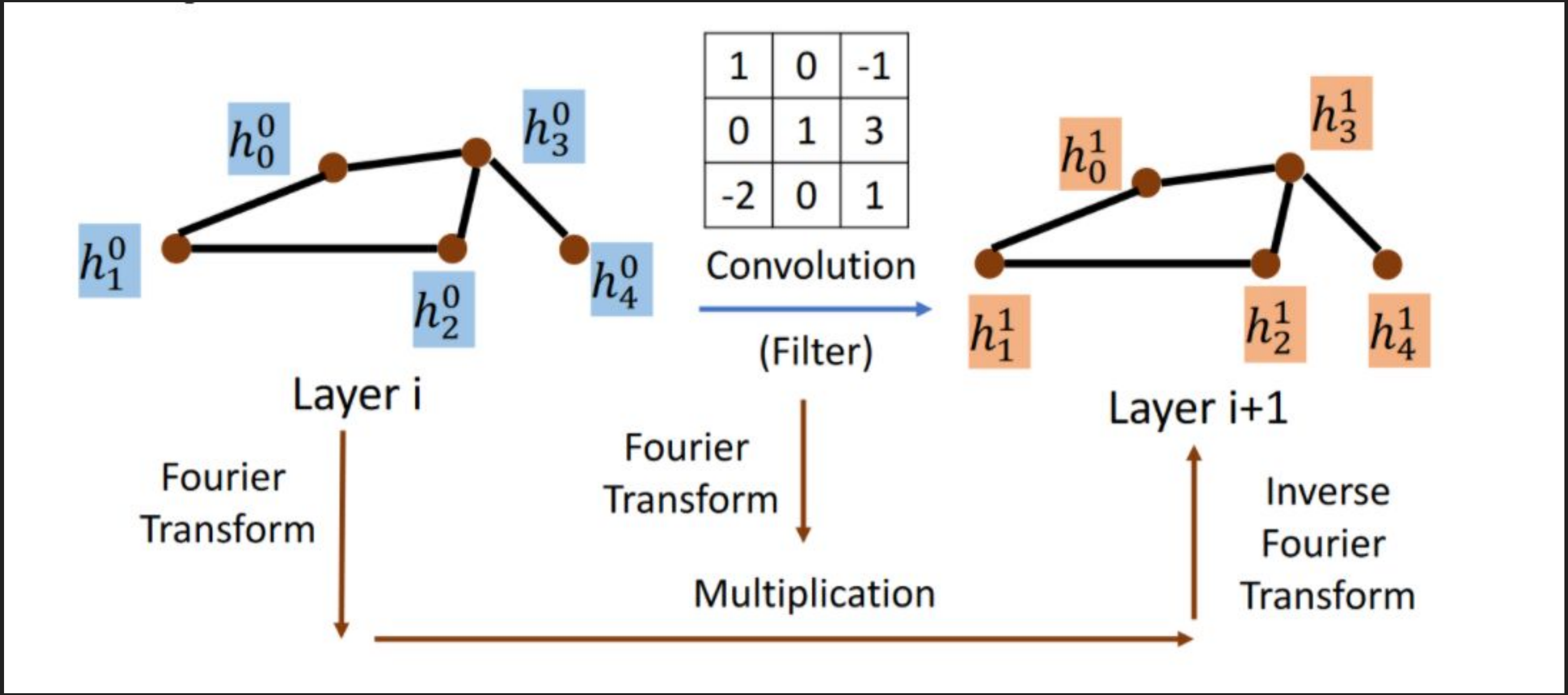
# Introduction

- We try to understand the graph embedding model from a new perspective and propose an attack framework: GF-Attack, which can perform adversarial attack on various kinds of graph embedding models.
- We formulate the graph embedding model as a general graph signal processing with corresponding graph filter which can be computed by the input adjacency matrix.
- GF-Attack is capable to perform the adversarial attack on any graph embedding models which can be formulate to a general graph signal processing.

# Preliminary

# Spectral-Based Convolution

將Graph Data與Graph Filter都做Fourier Transform，也就是將Graph Data與Graph Filter從Spatial Domain轉換至Frequency Domain，如此一來對要Graph Data做Convolution直接相乘就好，比較簡單，做完後再將Graph Data做Inverse Fourior Transform，使其從Frequency Domain變回Spatial Domain。



$h_0^0$
$h_3^0$
$h_1^0$
$h_2^0$
$h_4^0$
Layer i
1 0 -1
0 1 3
-2 0 1
Convolution
(Filter)
$h_0^1$
$h_3^1$
$h_1^1$
$h_2^1$
$h_4^1$
Layer i+1
Fourier Transform
Fourier Transform
Multiplication
Inverse Fourier Transform

# Spectral Graph Theory

- 考慮一無向圖，Graph Signal通常反應在Graph Node上(ie. Node Feature)

Graph: $G = (V, E), N = |V|$

$A \in \mathbb{R}^{N \times N}$, adjacency matrix (weight matrix).

$$A_{i,j} = 0 \; if \; e_{i,j} \notin E, else \; A_{i,j} = w(i,j)$$

We only consider undirected graph

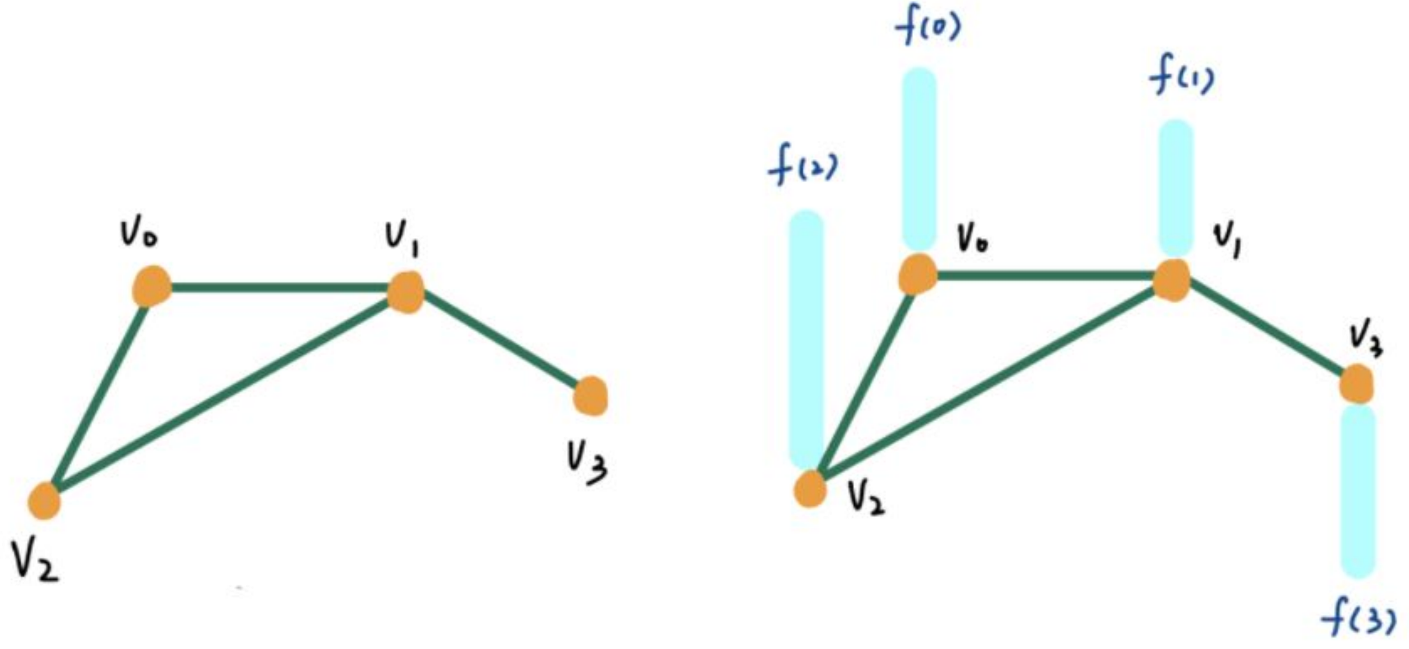
$D \in \mathbb{R}^{N \times N}$ , degree matrix

$$D_{i,j} = \begin{cases} d(i) & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad \text{(Sum of row } i \text{ in } A\text{)}$$

$f: V \to \mathbb{R}^N$, signal on graph (vertices). $f(i)$ denotes the signal on vertex $i$

# Spectral Graph Theory

- 假設一簡單圖G有V0、V1、V2、V3等節點，f(0)、f(1)、f(2)、f(3)指節點訊號



# Spectral Graph Theory

- 定義Laplacian Matrix，屬於半正定矩陣，對L特徵分解則可以得到Eigen-Vector U和Eigen-Value λ。下圖將Eigenvalue定義為Frequency，Eigenvector定義為Basis，也就是說L可以作為一運算子，對Graph Signal進行處理

Graph Laplacian $L = D - A,\ L \succcurlyeq 0$ (Positive semidefinite)

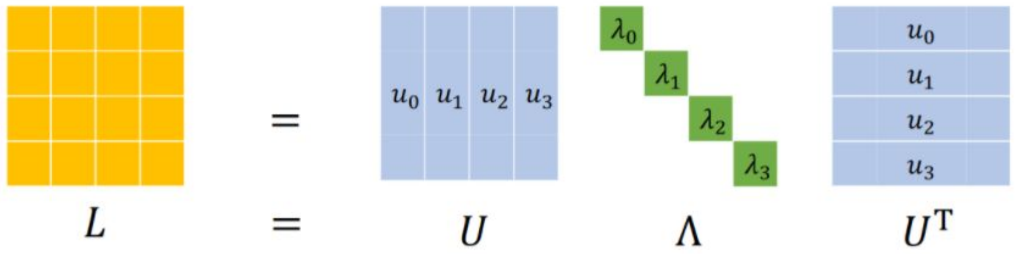
$L$ is symmetric (for undirected graph)

$L = U \Lambda U^{\mathrm{T}}$ (spectral decomposition)

$\Lambda = \mathrm{diag}(\lambda_0, \dots, \lambda_{N-1}) \in \mathbb{R}^{N \times N}$

$U = [u_0, \dots, u_{N-1}] \in \mathbb{R}^{N \times N}$, orthonormal

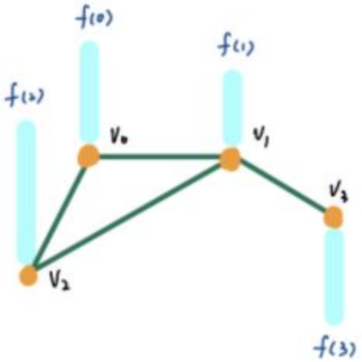
$\lambda_l$ is the frequency, $u_l$ is the basis corresponding to $\lambda_l$



# Spectral Graph Theory

- 根據G，可以得到下圖矩陣，f為事先定義之Signal，對L特徵分解可以得到對應的Eigenvalue和Eigenvector

✓ Vertex domain signal




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

$$L = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \quad \Lambda = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

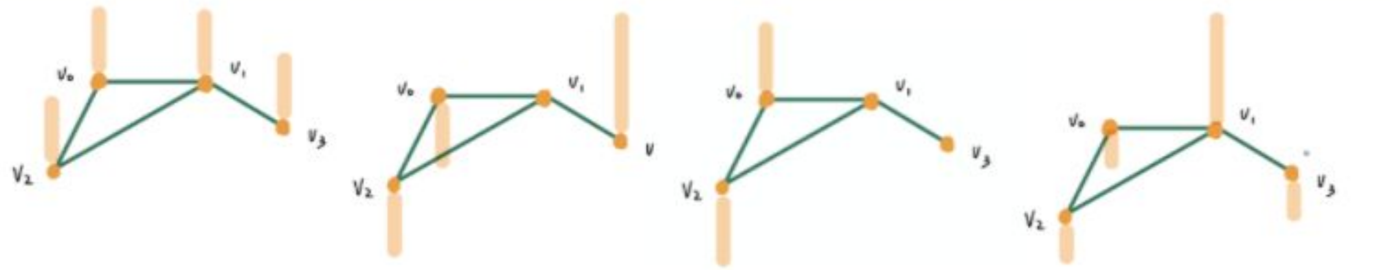
$$f = \begin{bmatrix} 4 \\ 2 \\ 4 \\ -3 \end{bmatrix} \quad U = \begin{bmatrix} 0.5 & -0.41 & 0.71 & -0.29 \\ 0.5 & 0 & 0 & 0.87 \\ 0.5 & -0.41 & -0.71 & -0.29 \\ 0.5 & 0.82 & 0 & -0.29 \end{bmatrix}$$

# Spectral Graph Theory

- 若將Eigenvalue視為Frequency，則不同Eigenvalue會對應不同Eigenvector，而它能被視為影響Signal大小的Transform

Low frequency → High frequency

| $\lambda$ | 0 | 1 | 3 | 4 |
|---|---|---|---|---|
| $u$ | $[0.5 \quad 0.5 \quad 0.5 \quad 0.5]^T$ | $[-0.41 \quad 0 \quad -0.41 \quad 0.82]^T$ | $[0.71 \quad 0 \quad -0.71 \quad 0]^T$ | $[-0.29 \quad 0.87 \quad -0.29 \quad -0.29]^T$ |



(DC component )

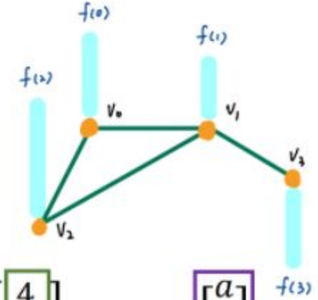
# Spectral Graph Theory

- L的物理意義在於作為一運算子，能考量不同鄰居間的Node Signal進行運算。以下圖V0節點的Signal為例，它會需要考慮V1和V2的Signal影響

## Spectral Graph Theory

✓ Interpreting vertex frequency

- $L$ as an operator on graph
- Given a graph signal $f$, what does $Lf$ mean?
- $Lf = (D - A)f = Df - Af$




$$D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad f = \begin{bmatrix} 4 \\ 2 \\ 4 \\ -3 \end{bmatrix} \quad Lf = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

- Lets focus on the first row of $Lf$
- $a = [2\ 0\ 0\ 0] \cdot [4\ 2\ 4\ -3] - [0\ 1\ 1\ 0] \cdot [4\ 2\ 4\ -3]$

\# of neighbors of $v_0$　　Signal on $v_0$'s neighbors

$$= 2 \times 4 - 2 - 4 = (4 - 2) + (4 - 4) = 2$$

Signal on $v_0$　　Sum of difference between $v_0$ and its neighbors

# Spectral Graph Theory

- 受到其他Signal影響可以定義為下圖公式，也就是計算Signal之間的平滑程度

✓ $(Lf)(v_i) = \sum_{v_j \in V} w_{i,j}(f(v_i) - f(v_j))$ , where $w_{i,j}$ is the $(i,j)^{th}$ entry of $A$

$$f^T L f = \sum_{v_i \in V} f(v_i) \sum_{v_j \in V} w_{i,j}(f(v_i) - f(v_j))$$

$$= \sum_{v_i \in V} \sum_{v_j \in V} w_{i,j}(f^2(v_i) - f(v_i)f(v_j))$$

$$= \frac{1}{2} \sum_{v_i \in V} \sum_{v_j \in V} w_{i,j}(f^2(v_i) - f(v_i)f(v_j) + f^2(j) - f(v_j)f(v_i))$$

$$= \frac{1}{2} \sum_{v_i \in V} \sum_{v_j \in V} w_{i,j}\left(f(v_i) - f(v_j)\right)^2$$

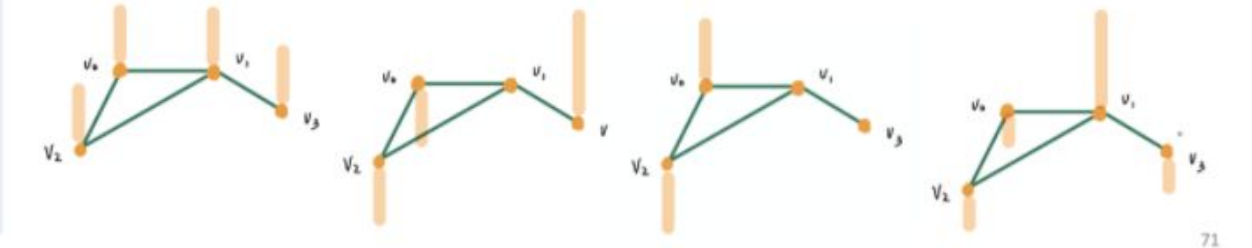
"Power" of signal variation between nodes, i.e., smoothness of graph signal

# Spectral Graph Theory

- 根據上述公式，可以將Eigenvalue視為Frequency
  - ✓ $u_i^T L u_i = u_i^T \lambda_i u_i = \lambda u_i^T u_i = \lambda_i$
  - ✓ The eigenvectors corresponding to small $\lambda$ belong to the low-pass part of a graph signal

Low frequency → High frequency

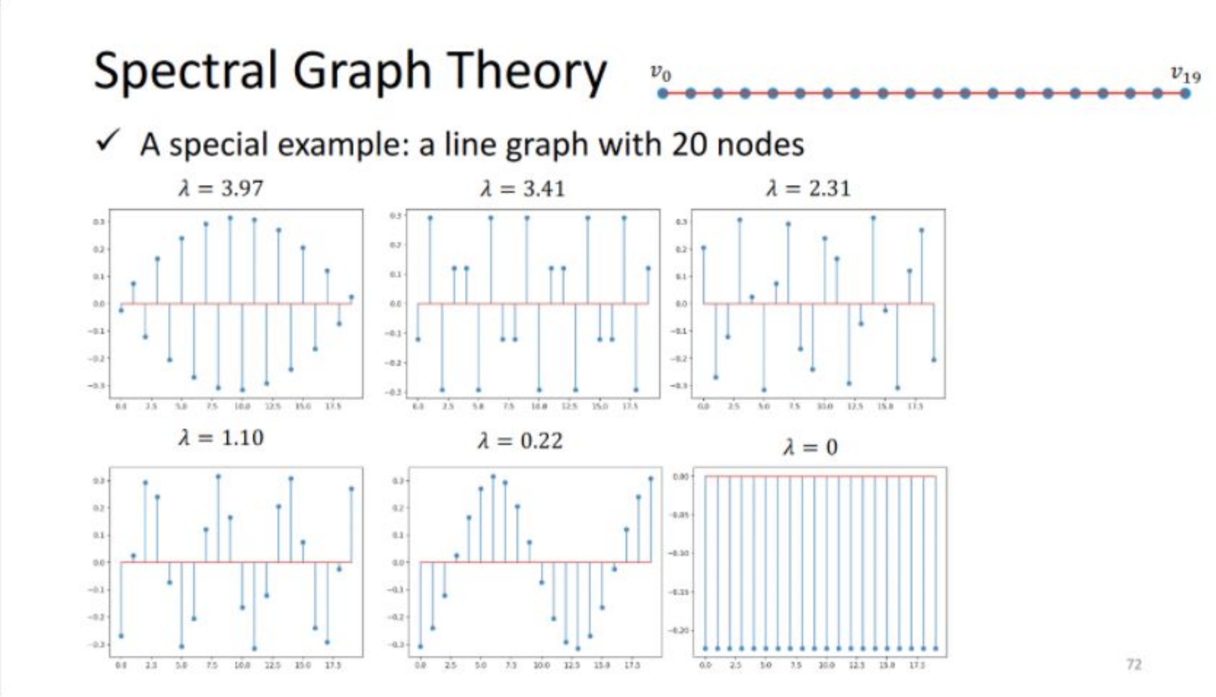
| $\lambda$ | 0 | 1 | 3 | 4 |
|---|---|---|---|---|
| $u$ | $[0.5 \quad 0.5 \quad 0.5 \quad 0.5]^T$ | $[-0.41 \quad 0 \quad -0.41 \quad 0.82]^T$ | $[0.71 \quad 0 \quad -0.71 \quad 0]^T$ | $[-0.29 \quad 0.87 \quad -0.29 \quad -0.29]^T$ |



(DC component )

# Spectral Graph Theory

- 以下圖證明Eigenvalue擁有Frequency特性



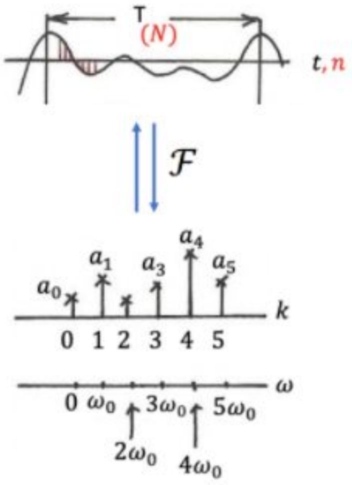
# Graph Fourier Transform

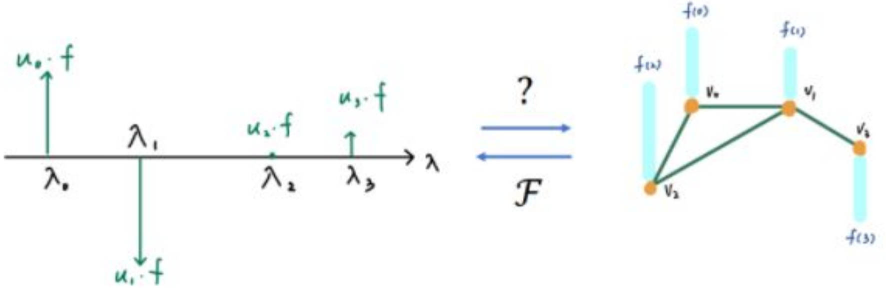
- 應用Fourier Transform的概念，將Vertex Domain(Spatial Domain)的訊號，透過Eigenvector Transpose投射到Spectral Domain上
  - ✓ Graph Fourier Transform of signal $x$: $\hat{x} = U^T x$

$$\begin{array}{c} 分析 \\ 分析 \\ 分析 \\ 分析 \end{array} \begin{bmatrix} u_0 \cdot x \\ u_1 \cdot x \\ u_2 \cdot x \\ u_3 \cdot x \end{bmatrix} = \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$

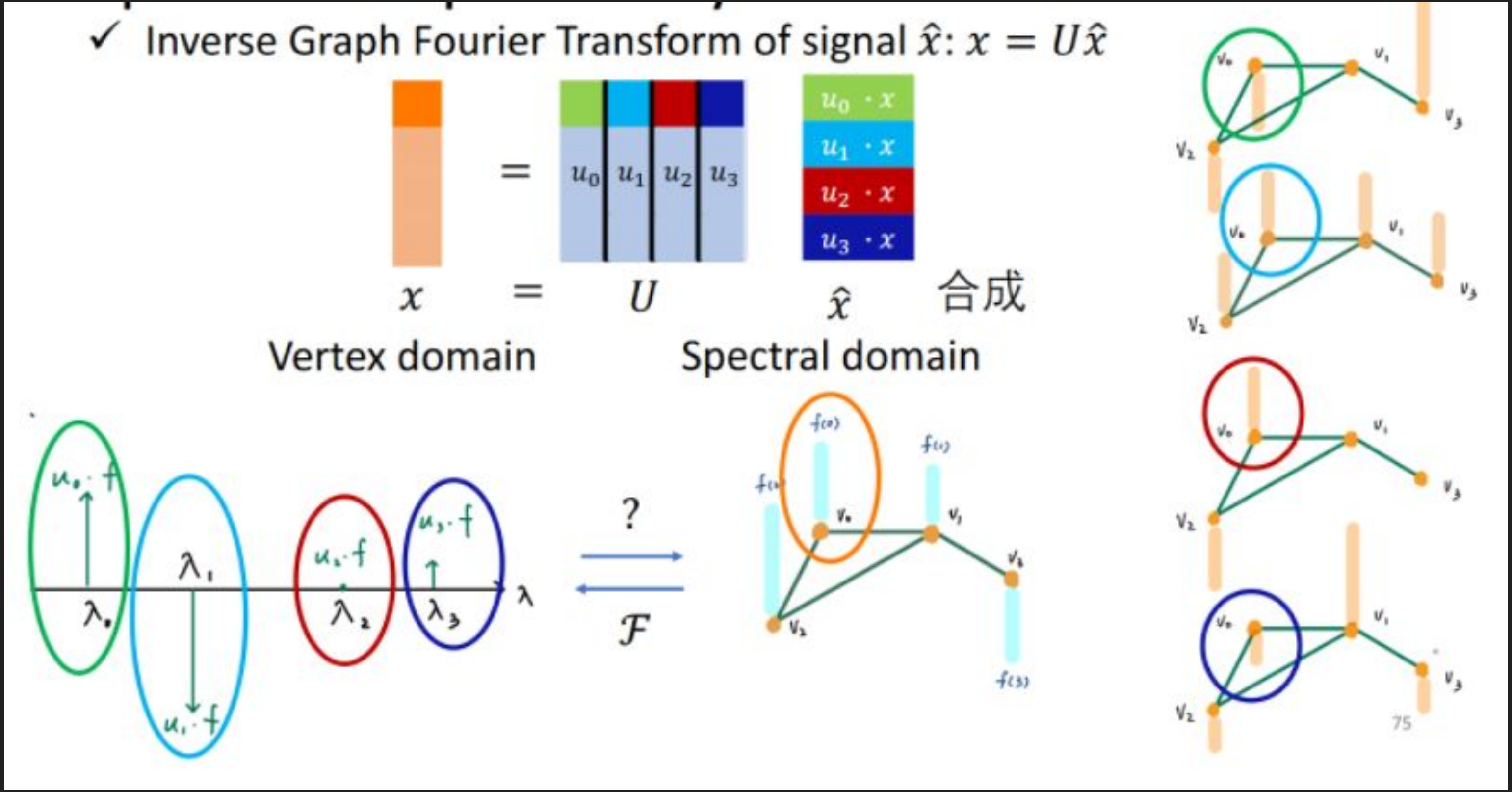
$$\hat{x} = U^{\mathrm{T}} \quad x \qquad 合成$$

Spectral domain　　Vertex domain



# Inverse Graph Fourier Transform



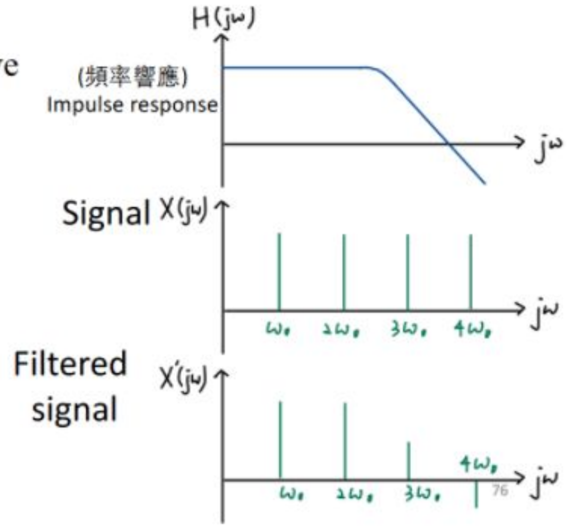
# Filtering

- 針對不同頻率間的訊號進行過濾

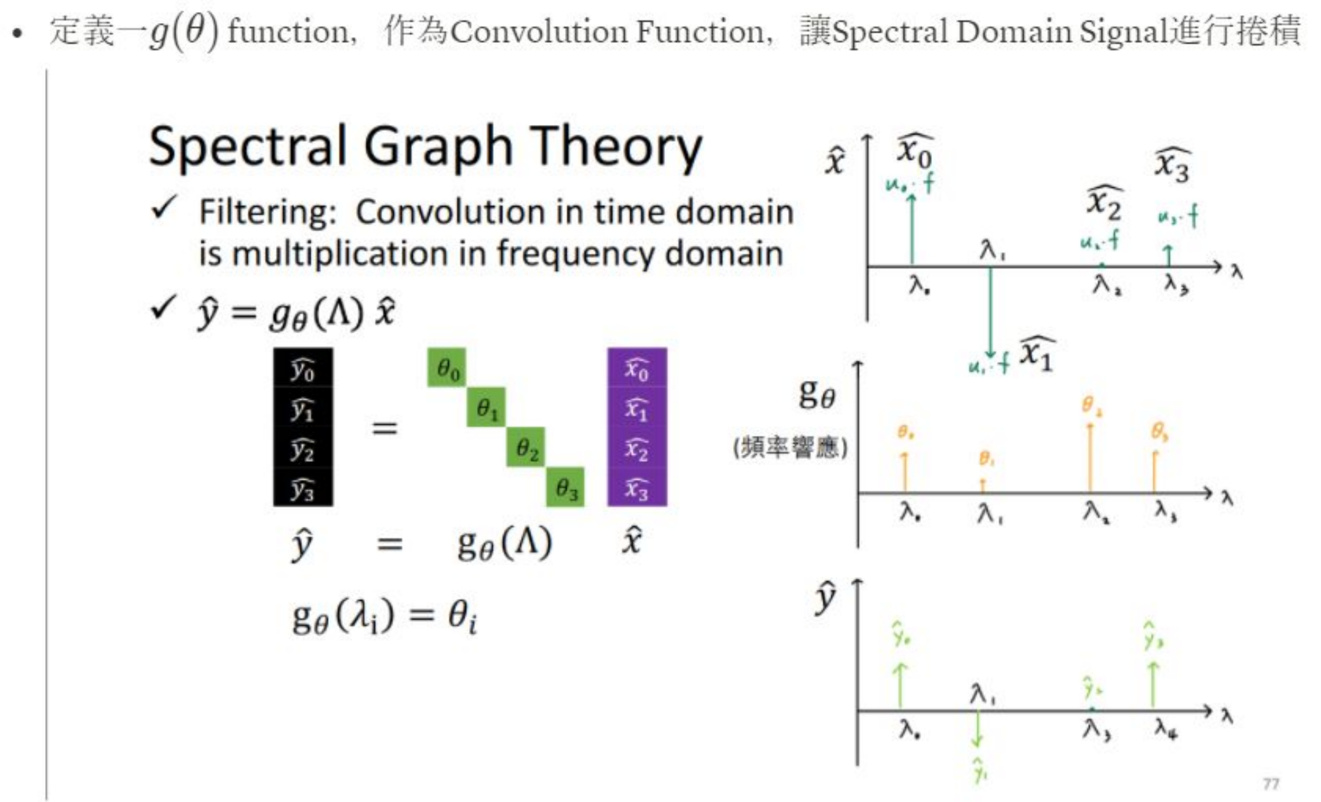
## Filtering

modifying the amplitude/ phase of the different frequency components in a signal, including eliminating some frequency components entirely

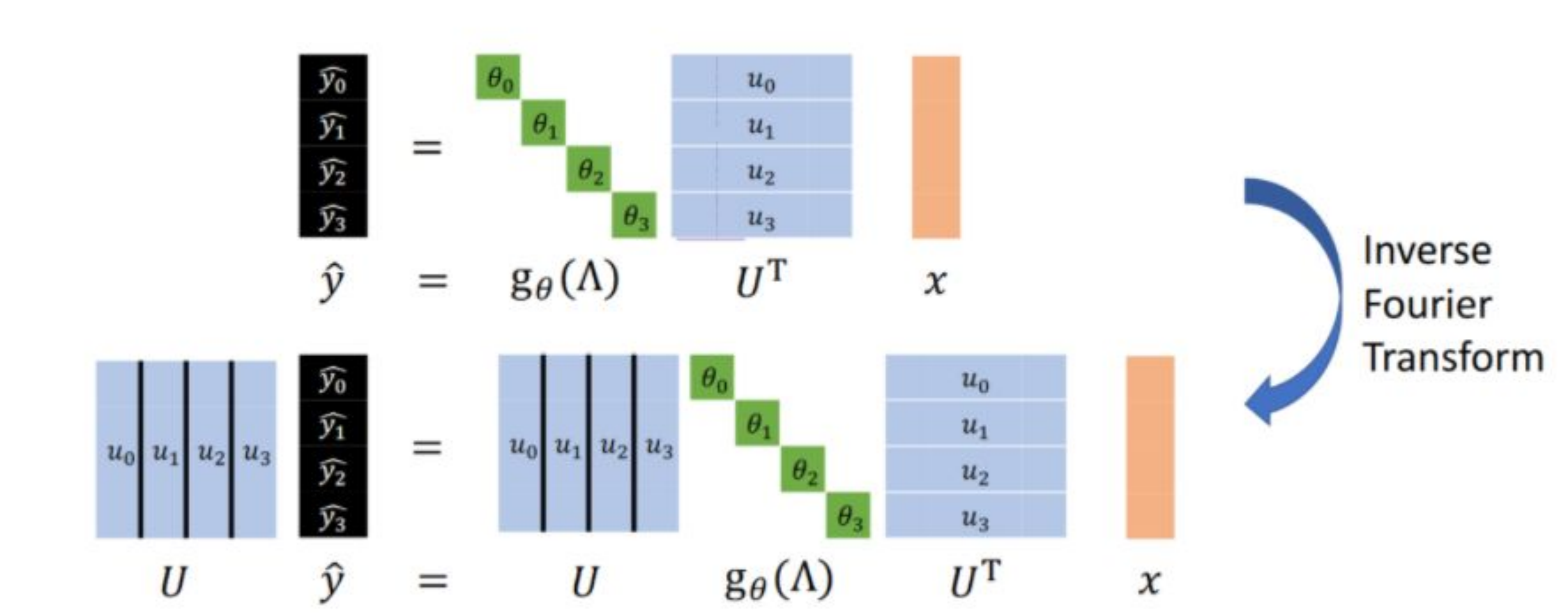
– frequency shaping, frequency selective



# Filtering

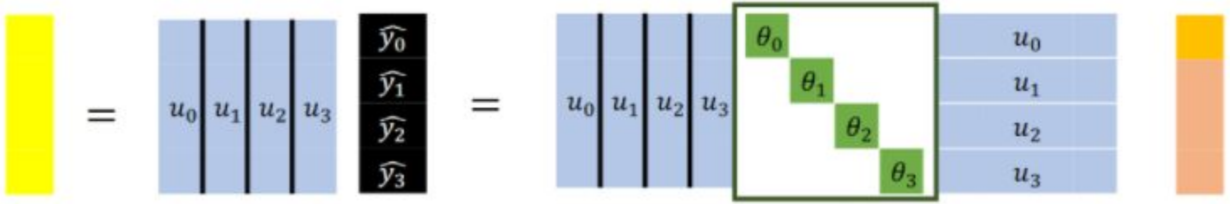

• 定義一$g(\theta)$ function，作為Convolution Function，讓Spectral Domain Signal進行捲積
Spectral Graph Theory
✓ Filtering: Convolution in time domain is multiplication in frequency domain
✓ $\hat{y} = g_\theta(\Lambda)\,\hat{x}$
$\hat{y} = g_\theta(\Lambda)\ \hat{x}$
$g_\theta(\lambda_i) = \theta_i$
$\hat{x}$
$g_\theta$
(頻率響應)
$\hat{y}$
77

# Inverse Graph Fourier Transform



# Spectral Graph Theory

- 最後得到下圖式子，但這樣會遇到第一個問題，整體會受限於Input Data的N維大小，效率為O(N)




$$y \quad U \quad \hat{y} \quad = \quad U \quad g_\theta(\Lambda) \quad U^{\mathrm{T}} \quad x$$

$$= g_\theta\left(U\Lambda U^{\mathrm{T}}\right)x$$

$$= g_\theta(L)x$$

Parameters to be learnt

$$y = g_\theta(L)x$$

$g_\theta(\cdot)$ can be any function. For example,

$$g_\theta(L) = \log(\mathrm{I} + L) = L - \frac{L^2}{2} + \frac{L^3}{3} \dots, \lambda_{max} < 1$$

Problem 1: Learning complexity is $O(N)$!!!

# Spectral Graph Theory

- 但若考慮g = $L^N$ function，則會考慮到N個Node的影響，因此缺乏Localize特性

✓ If we select $g_\theta(L) = cos(L) = I - \frac{L^2}{2!} + \frac{L^4}{4!} \ldots$

✓ If a connected graph has $N$ nodes, then $L^N$ will make the all nodes be able to share their signals with each other

Problems 2: Not localize!!!

# ChebNet

- 將上述g function引入ChebNet機制，但會產生第三個問題為高Time Complexity

✓ Solution to Problem 1 and 2:

➢ Use polynomial to parametrize $g_\theta(L)$

$$g_\theta(L) = \sum_{k=0}^{\boxed{K}} \boxed{\theta_k} L^k$$

Now it is K-localized

Parameters to be learnt: $O(K)$

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

Problem 3:

Time complexity: $O(N^2)$

$$y = U g_\theta(\Lambda) U^T x = U\left(\sum_{k=0}^{K} \theta_k \Lambda^k\right) U^T x$$

# GCN

# GCN

https://openreview.net/pdf?id=SJU4ayYgl

✓ $y = g_{\theta'}(L)x = \sum_{k=0}^{K} \theta'_k T_k(\tilde{L})x \;, K = 1$

$$y = g_{\theta'}(L)x = \theta'_0 x + \theta'_1 \tilde{L} x \qquad \because \tilde{L} = \frac{2L}{\lambda_{max}} - I$$

$$= \theta'_0 x + \theta'_1 (\frac{2L}{\lambda_{max}} - I)x \qquad \because \lambda_{max} \approx 2$$

$$= \theta'_0 x + \theta'_1 (L - I)x \qquad \because L = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$$

$$= \theta'_0 x - \theta'_1 (D^{-\frac{1}{2}} A D^{-\frac{1}{2}})x \qquad \because \theta = \theta'_0 = -\theta'_1$$

$$= \theta(I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}})x$$

*renormalization trick:* $I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)}\right)$$

# GCN

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$$

Can be rewritten as:

$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} Wx_u + b\right), \quad \forall v \in \mathcal{V}.$$

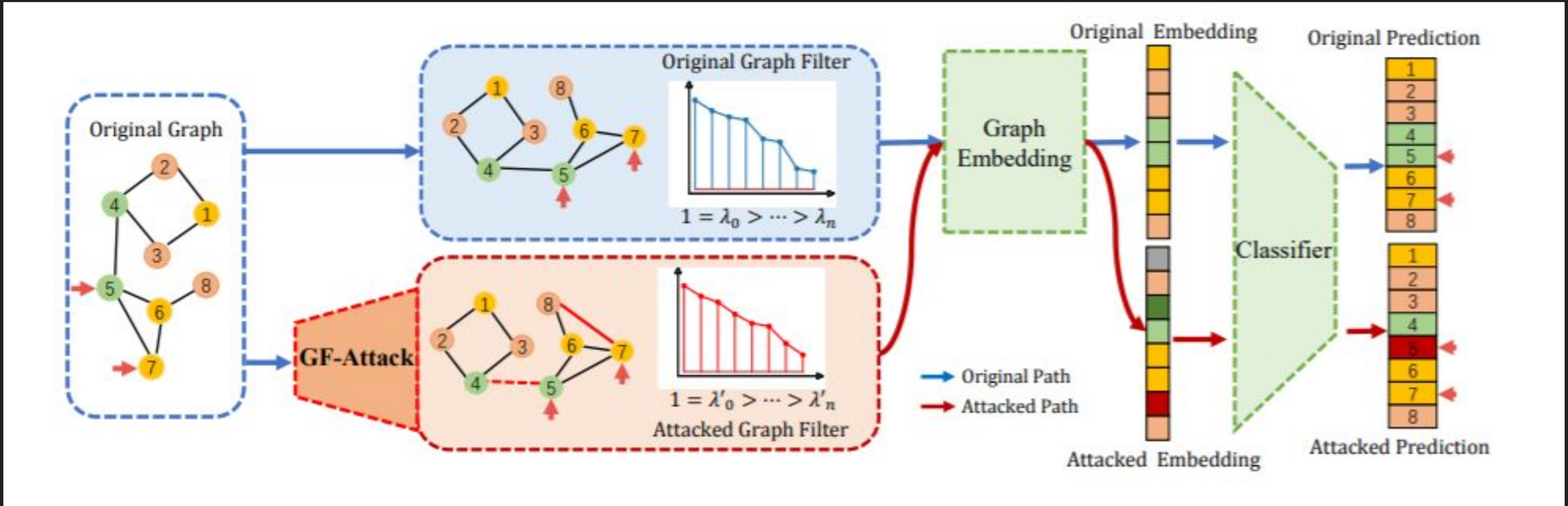
# Methodology

# GF Attack

- Attack on $\mathcal{V}$: Add/delete vertices in graphs. This operation may change the dimension of the adjacency matrix $A$.
- Attack on $A$: Add/delete edges in graphs. This operation would lead to the changes of entries in the adjacency matrix $A$. This kind of attack is also known as *structural attack*.
- Attack on $X$: Modify the attributes attached on vertices.

Here, we mainly focus on adversarial attacks on graph structure $A$, since attacking $A$ is more practical than others in real applications (Tong et al. 2012).

# Architechture



# Adversarial Attack Definition

- Adversarial Attack數學表示，beta指budget

$$\arg\max_{A'} \mathscr{L}(A', Z) \tag{1}$$
$$\text{s.t. } Z = \mathscr{M}_{\Theta}(A', X),$$
$$\Theta^* = \arg\min_{\Theta} \mathcal{L}(\Theta; A', X), \|A' - A\| = 2\beta,$$

# Methodologies

- Graph Signals According to Graph Filter H together with feature transformation.

$$\tilde{X} = \mathscr{H}(X),\ X' = \sigma(\tilde{X}\Theta), \tag{2}$$

- 攻擊目標為quality of the output embedding Z，S'指干擾後的Graph Shift Filter

$$\mathscr{L}(A', Z) = \|h(S')X - h(S')_T X\|_F^2, \tag{3}$$

- 根據Low-rank Approximation可得下方式子

$$\mathscr{L}(A', Z) = \|\sum_{i=T+1}^{n} \lambda_i' \mathbf{u}_i \mathbf{u}_i^T X\|_F \leq \sum_{i=T+1}^{n} \lambda_i'^2 \cdot \sum_{i=T+1}^{n} \|\mathbf{u}_i^T X\|_2^2, \tag{4}$$

# Methodologies

- 以Upper Bound作為最後 Loss function

$$\arg\max_{A'} \sum_{i=T+1}^{n} {\lambda'_i}^2 \cdot \sum_{i=T+1}^{n} \|\mathbf{u}_i^T X\|_2^2,$$
$$\text{s.t. } \|A' - A\| = 2\beta. \tag{5}$$

# Algorithm

---

**Algorithm 1** Graph Filter Attack (GF-Attack) adversarial attack algorithm under RBA setting

---

**Input:**

Adjacent Matrix $A$; feature matrix $X$; target vertex $t$; number of top-$T$ smallest singular values/vectors selected $T$; order of graph filter $K$; fixed budget $\beta$.

**Output:**

Perturbed adjacent Matrix $A'$.

Initial the candidate flips set as $\mathcal{C} = \{(v, t)|v \neq t\}$, eigenvalue decomposition of $\hat{A} = U_{\hat{A}} \Lambda_{\hat{A}} U_{\hat{A}}^T$;
**for** $(v, t) \in \mathcal{C}$ **do**
  Approximate $\Lambda'_{\hat{A}}$ resulting by removing/inserting edge $(v, t)$ via Equation (9);
  Update $Score_{(v,t)}$ from loss Equation (8) or Equation (12);
**end for**
$\mathcal{C}_{sel} \leftarrow$ edge flips with top-$\beta$ $Score$;
$A' \leftarrow A \pm \mathcal{C}_{sel}$;
**return** $A'$

---

# Experiment Result

# Dataset

scientific publications

links

unique words

classes

# Dataset

| | Cora | Citeseer | Pubmed |
|---|---|---|---|
| **Task** | Transductive | Transductive | Transductive |
| **# Nodes** | 2708 (1 graph) | 3327 (1 graph) | 19717 (1 graph) |
| **# Edges** | 5429 | 4732 | 44338 |
| **# Features/Node** | 1433 | 3703 | 500 |
| **# Classes** | 7 | 6 | 3 |
| **# Training Nodes** | 140 | 120 | 60 |
| **# Validation Nodes** | 500 | 500 | 500 |
| **# Test Nodes** | 1000 | 1000 | 1000 |

# Steps

train GCN

GF Attack

result

# Our result

Cora

The test accuracy performance is: 0.8390

Final accuracy after attack is: 0.8109

Attack performance: -0.0281

Citeseer

The test accuracy performance is: 0.7465

Final accuracy after attack is: 0.73

Attack performance: -0.0165

Table 1: Summary of the change in classification accuracy (in percent) compared to the clean/original graph. Single edge perturbation under RBA setting. Lower is better.

| Dataset | Cora | | | | Citeseer | | | | Pubmed | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Models | GCN | SGC | DeepWalk | LINE | GCN | SGC | DeepWalk | LINE | GCN | SGC | DeepWalk | LINE |
| (unattacked) | 80.20 | 78.82 | 77.23 | 76.75 | 72.50 | 69.68 | 69.68 | 65.15 | 80.40 | 80.21 | 78.69 | 72.12 |
| *Random* | -1.90 | -1.22 | -1.76 | -1.84 | -2.86 | -1.47 | -6.62 | -1.78 | -1.75 | -1.77 | -1.25 | -1.01 |
| *Degree* | -2.21 | -4.42 | -3.08 | -12.40 | -4.68 | -5.21 | -9.67 | -12.55 | -3.86 | -4.44 | -2.43 | -13.05 |
| *RL-S2V* | -5.20 | -5.62 | -5.24 | -10.38 | -6.50 | -4.08 | -12.13 | -20.10 | -6.40 | -6.11 | -6.10 | -13.21 |
| $\mathcal{A}_{class}$ | -3.62 | -2.96 | **-6.29** | -7.55 | -3.48 | -2.83 | **-12.56** | -10.28 | -4.21 | -2.25 | -3.05 | -6.75 |
| *GF-Attack* | **-7.60** | **-9.73** | -5.31 | **-13.27** | **-7.78** | **-6.19** | -12.50 | **-22.11** | **-7.96** | **-7.20** | **-7.43** | **-14.16** |

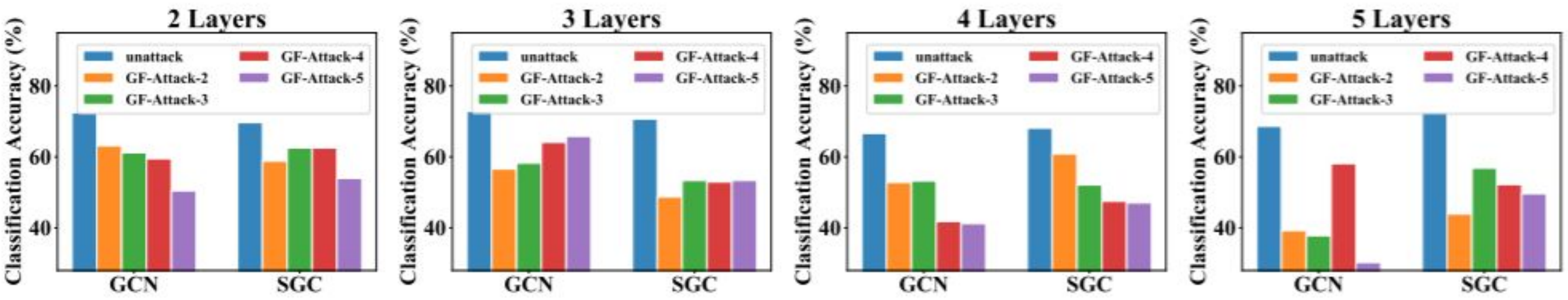



Figure 2: Comparison between order $K$ of *GF-Attack* and number of layers in GCN/SGC on Citeseer.

Table 2: Running time ($s$) comparison over all baseline methods on Citeseer. We report the 10 times average running time of processing single node for each model.

| Models | *Random* | *Degree* | *RL-S2V* | $\mathcal{A}_{class}$ | *GF-Attack* |
|---|---|---|---|---|---|
| Citeseer | 0.19 | 42.21 | 222.80 | 146.58 | 12.78 |

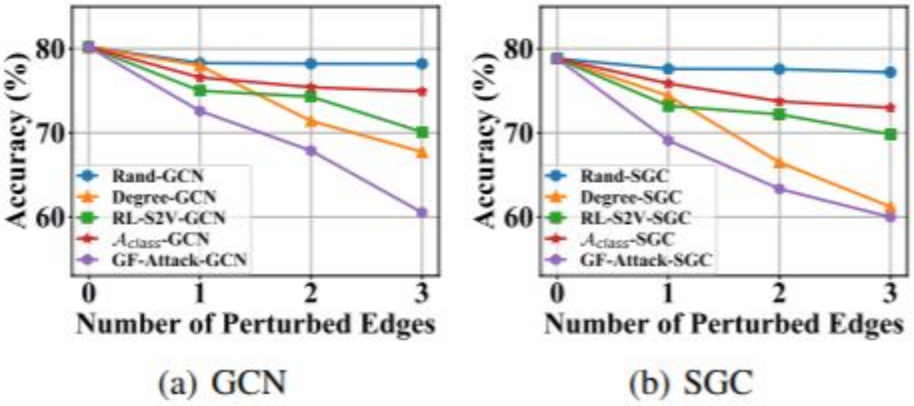



Figure 3: Multiple-edge attack results on Cora under RBA setting. Lower is better.