CSL
Computer Systems Laboratory

ゼミ発表３回目

清　恵人

Graph Neural Network

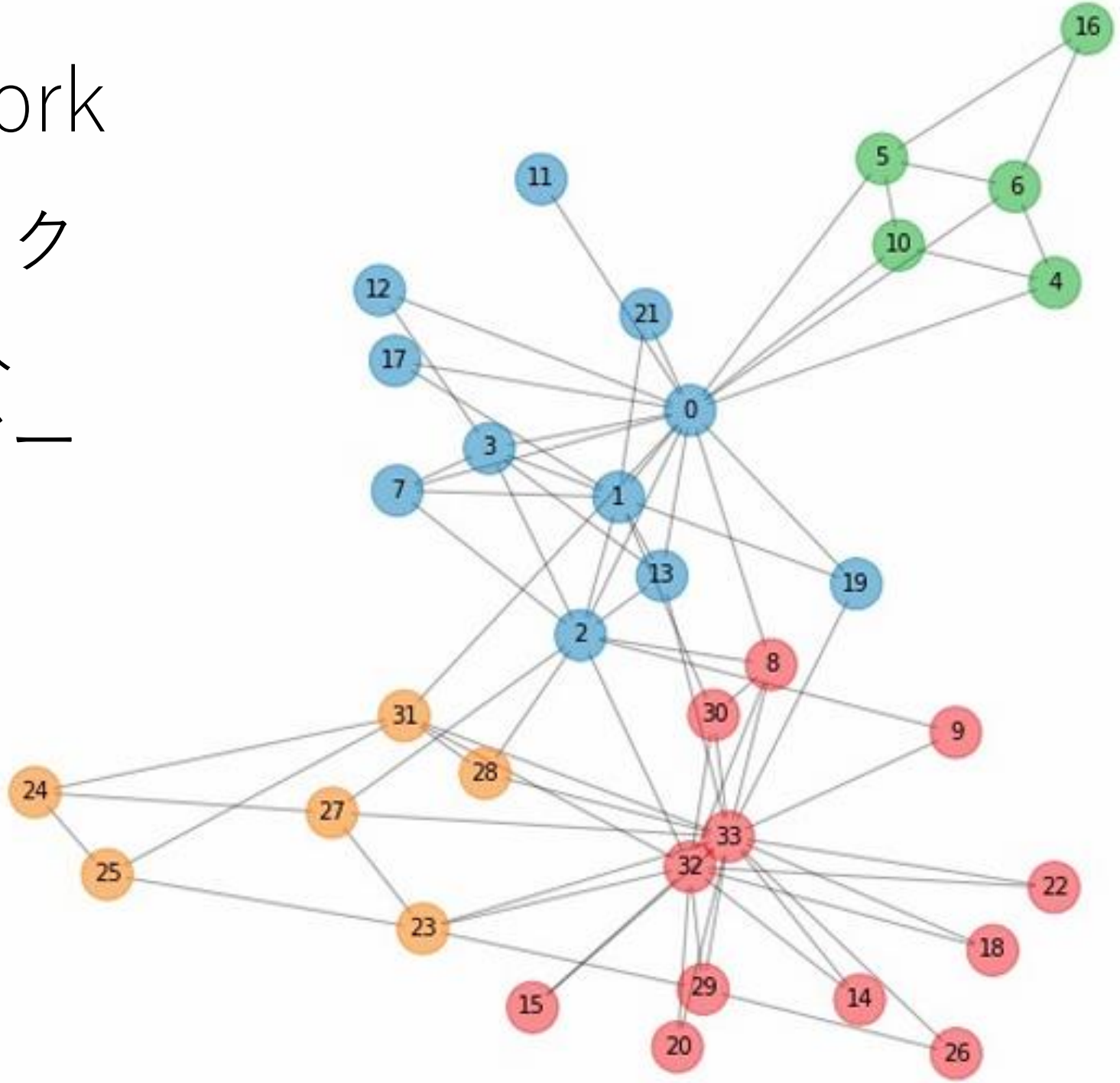
空手クラブネットワーク

・ノード数：会員34人

・特徴量：会員ナンバー
ワンホットベクトル

・ラベル：4グループ

・エッジ：会員同士の
交友関係１５６本



グラフ構造: Data(x=[34, 34], edge_index=[2, 156], y=[34], train_mask=[34])

epoch:100
最適化:Adam,lr=0.01
損失：クロスエントロピー
pytroch_geometirc

```
====== ノードの特徴量:x ======
tensor([[1., 0., 0.,  ..., 0., 0., 0.],
        [0., 1., 0.,  ..., 0., 0., 0.],
        [0., 0., 1.,  ..., 0., 0., 0.],
        ...,
        [0., 0., 0.,  ..., 1., 0., 0.],
        [0., 0., 0.,  ..., 0., 1., 0.],
        [0., 0., 0.,  ..., 0., 0., 1.]])
====== ノードのクラス:y ======
tensor([1, 1, 1, 1, 3, 3, 3, 1, 0, 1, 3, 1, 1, 1, 0, 0, 3, 1, 0, 1, 0, 1, 0, 0,
        2, 2, 0, 0, 2, 0, 0, 2, 0, 0])
========= エッジ形状 =========
tensor([[ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,
          1,  1,  1,  1,  1,  1,  1,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  3,
          3,  3,  3,  3,  3,  4,  4,  4,  5,  5,  5,  5,  6,  6,  6,  6,  7,  7,
          7,  7,  8,  8,  8,  8,  8,  9,  9, 10, 10, 10, 11, 12, 12, 13, 13, 13,
         13, 13, 14, 14, 15, 15, 16, 16, 17, 17, 18, 18, 19, 19, 19, 20, 20, 21,
         21, 22, 22, 23, 23, 23, 23, 23, 24, 24, 24, 25, 25, 25, 26, 26, 27, 27,
         27, 27, 28, 28, 28, 29, 29, 29, 29, 30, 30, 30, 30, 31, 31, 31, 31, 31,
         31, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 33, 33, 33, 33, 33,
         33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33],
        [ 1,  2,  3,  4,  5,  6,  7,  8, 10, 11, 12, 13, 17, 19, 21, 31,  0,  2,
          3,  7, 13, 17, 19, 21, 30,  0,  1,  3,  7,  8,  9, 13, 27, 28, 32,  0,
          1,  2,  7, 12, 13,  0,  6, 10,  0,  6, 10, 16,  0,  4,  5, 16,  0,  1,
          2,  3,  0,  2, 30, 32, 33,  2, 33,  0,  4,  5,  0,  0,  3,  0,  1,  2,
          3, 33, 32, 33, 32, 33,  5,  6,  0,  1, 32, 33,  0,  1, 33, 32, 33,  0,
```

実験結果　すべて正解

モデルの予測

- [1, 1, 1, 1, 3, 3, 3, 1, 0, 1, 3, 1, 1, 1, 0, 0, 3, 1, 0, 1, 0, 1, 0, 0, 2, 2, 0, 0, 2, 0, 0, 2, 0, 0]

ラベル

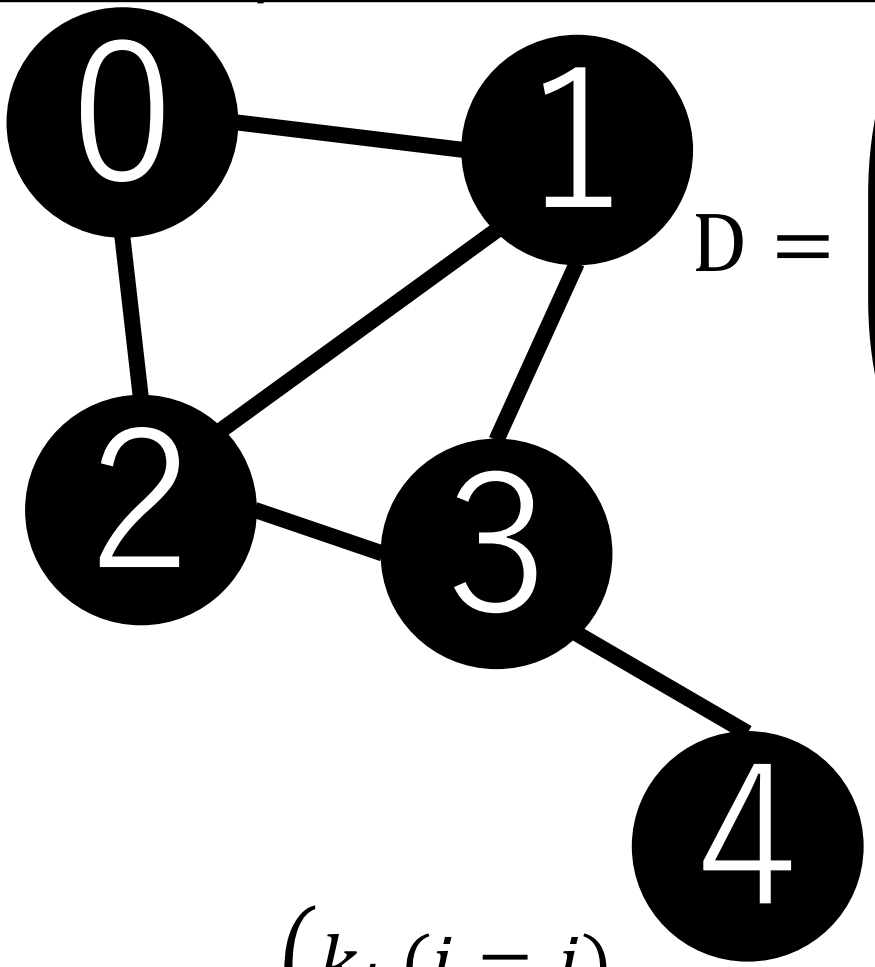
- [1, 1, 1, 1, 3, 3, 3, 1, 0, 1, 3, 1, 1, 1, 0, 0, 3, 1, 0, 1, 0, 1, 0, 0,2, 2, 0, 0, 2, 0, 0, 2, 0, 0]

```
class Net(torch.nn.Module):
    def __init__(self):
        super(Net,self).__init__()
        hidden_size = 5
        self.conv1=GCNConv(dataset.num_node_features,hidden_size)
        self.conv2=GCNConv(hidden_size,hidden_size)
        self.linear=torch.nn.Linear(hidden_size,dataset.num_classes)
```

```
    def forward(self,data):
        x = data.x
        edge_index = data.edge_index
        x = self.conv1(x,edge_index)
        x=F.relu(x)
        x=self.conv2(x,edge_index)
        x=F.relu(x)
        x=self.linear(x)
        return x
```

各ノードの特徴ベクトル（会員情報）
エッジ情報を取り出す。
１層目のグラフ畳み込み
活性化関数ReLU
２層目のグラフ畳み込み
活性化関数ReLu
３層目

Graph neural network convolution



$$\mathrm{D} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

次数行列

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

隣接行列

$$L' = D - A$$

$$L'_{ij} = \begin{cases} k_i \ (i = j) \\ -1 (i \neq j \ A_{ij} = 1) \\ 0 \ (else) \end{cases}$$

$$L' = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 \\ -1 & -1 & 3 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

グラフラプラシアン

正規化グラフラプラシアン行列

$$\mathcal{L} = D^{-\frac{1}{2}} L' D^{-\frac{1}{2}}$$

$$\mathrm{D}^{-\frac{1}{2}} = \begin{pmatrix} 0.707 & 0 & 0 & 0 & 0 \\ 0 & 0.577 & 0 & 0 & 0 \\ 0 & 0 & 0.577 & 0 & 0 \\ 0 & 0 & 0 & 0.577 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$\mathcal{L}$: 正規化グラフラプラシアン行列

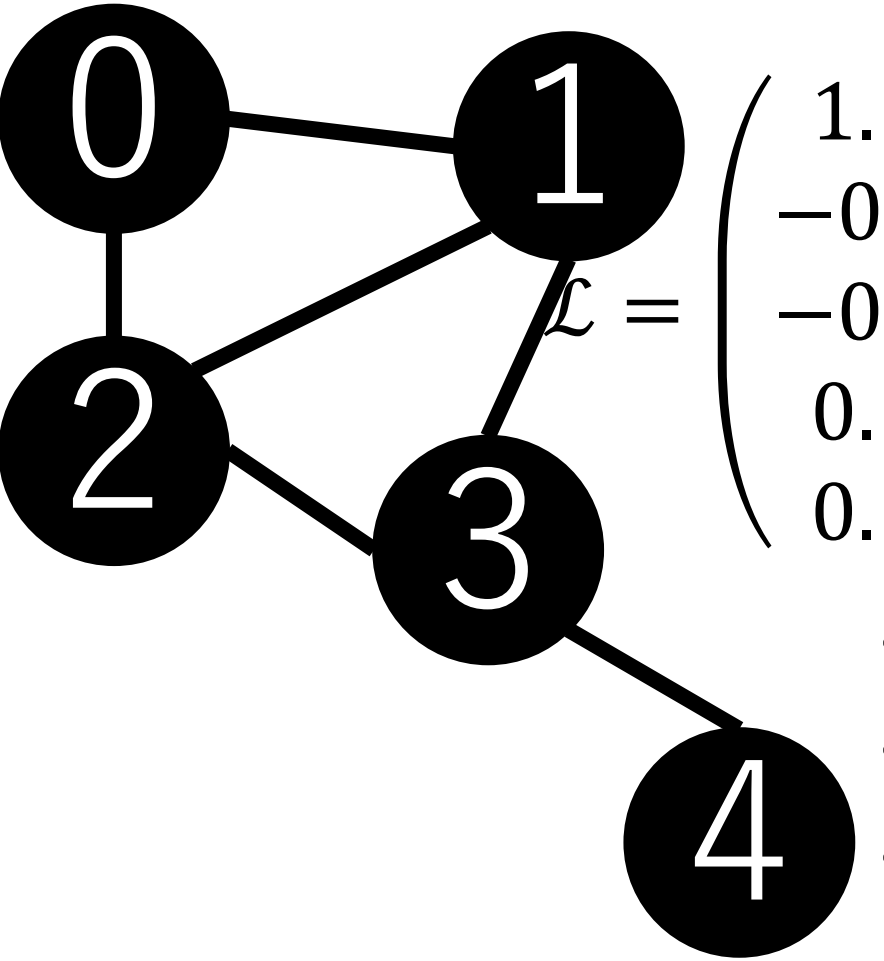
$D^{-\frac{1}{2}}$：次数行列平方根の逆数

L'：グラフラプラシアン行列

$$\mathcal{L} = \begin{pmatrix} 1. & -0.408 & -0.408 & 0. & 0. \\ -0.408 & 1. & -0.333 & -0.333 & 0. \\ -0.408 & -0.333 & 1. & -0.333 & 0. \\ 0. & -0.333 & -0.333 & 1. & -0.577 \\ 0. & 0. & 0. & -0.577 & 1. \end{pmatrix}$$

$$\boxed{\mathcal{L} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}}$$

正規化グラフラプラシアン行列



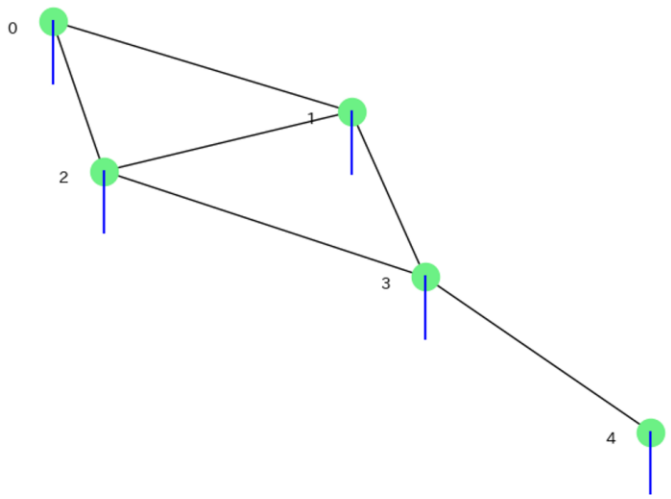
$$\mathcal{L} = \begin{pmatrix} 1. & -0.408 & -0.408 & 0. & 0. \\ -0.408 & 1. & -0.333 & -0.333 & 0. \\ -0.408 & -0.333 & 1. & -0.333 & 0. \\ 0. & -0.333 & -0.333 & 1. & -0.577 \\ 0. & 0. & 0. & -0.577 & 1. \end{pmatrix}$$

- ノード１（次数３）とノード２（次数３）**-0.333**
- ノード０（次数２）とノード１（次数３）**-0.408**
- ノード３（次数３）とノード４（次数１）**-0.577**

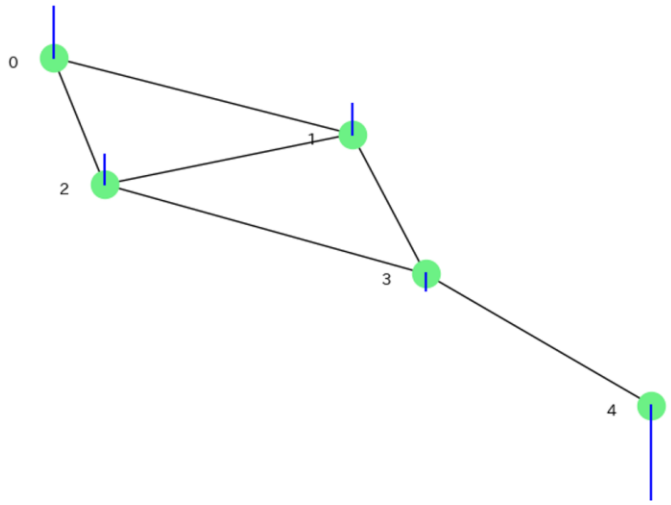
次数の高いノード間を繋ぐと重みが小さく、
次数の低いノード間を繋ぐエッジほど重みが大きい

固有値

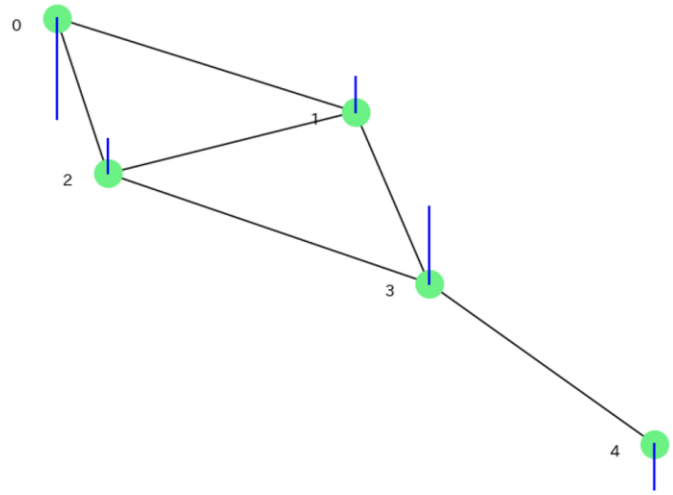
1．固有値＝ 0



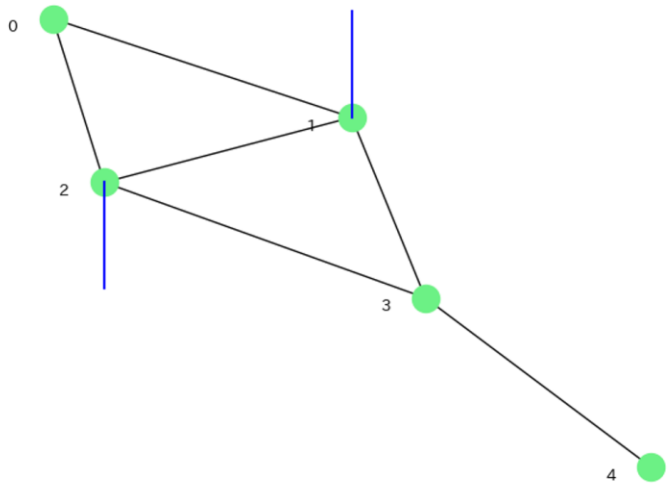
2固有値=0.82991



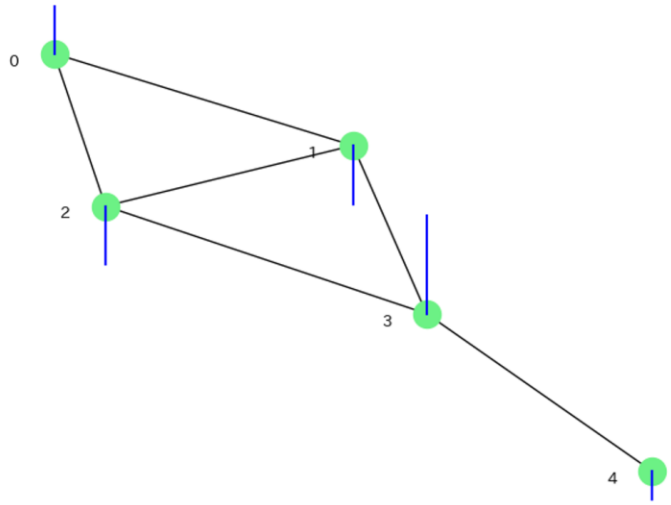
3固有値=2.6889



4固有値=4.0



5固有値=4.48119



固有値小

↓

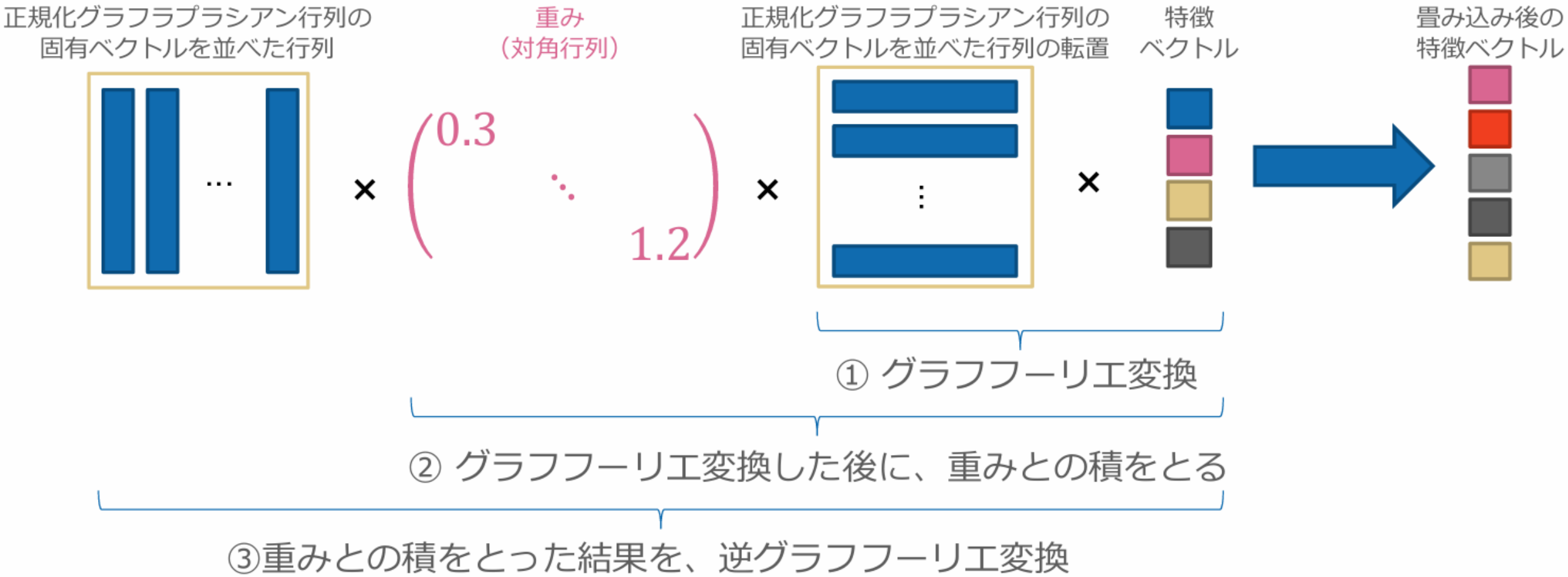
変化小

固有値大

↓

変化大

グラフフーリエ変換



- $\mathcal{F}_G(x) = Q^T x$　　$\mathcal{F}_G^{-1}(x) = Qx$ (逆グラフフーリエ変換)

Q:正規化グラフラプラシアン行列の固有ベクトルを並べた行列

x：全ノードの特徴を並べたもの

Spectral Convolution

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

- 特徴ベクトルxとフィルタgを用いた畳み込み

$$x * g = \mathcal{F}_G^{-1}\left(\mathcal{F}_G(x) \odot \mathcal{F}_G(g)\right)$$

$$\mathcal{F}_G = Q^T x, F_G^{-1} = Qx \text{ より}$$

$$x * g = Q(Q^T x \odot Q^T g) = Q(Q^T g \odot Q^T x) = QQ^T g \odot Q^T x$$

$Q^T g = \theta$とおき、これを重み（パラメーター）とする

$$x * g = Q\theta \odot Q^{Tx}$$

要素積をなくすために、θの要素を対角行列にまとめg_θとおく

$$x * g = Q g_\theta Q^T x$$

グラフフーリエ変換

逆グラフフーリエ変換

重みパラメーター

ChebNet

- 固有値の計算　行列Qに対して$O(N^2)$と高い
- 固有値の計算の多項式　チェビシェフ多項式の近似

K次までのチェビシェフ多項式の近似

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

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

$$\check{\Lambda} = \frac{2}{\lambda_{\max}} \mathcal{L} - I$$

λ ：Lの最大値の固有値

$$cos2\theta = 2\cos^2\theta - 1$$
$$T_2(x) = 2x^2 - 1$$
$$cos3\theta = 4\cos^3\theta - 3cos\theta$$
$$T_3(x) = 4x^3 - 3x$$
$$T_0 = 1, T_1 = x$$
$$T_{k+2} = 2xT_{k+1}(x) - T_k(x)$$

$$x * g = Q g_\theta Q^T x$$
$$= \sum_{k=0}^{K} \theta_k T_k(\check{L}) x$$

Graph Convolutional Networks

- 1次のチェビシェフの近似
- λ max=2による式の単純化
- パラメーター数の減少
- renormalization trick

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

$\theta_0 = \theta_1$とする

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

$$I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \check{D}^{-\frac{1}{2}} \check{A} \check{D}^{-\frac{1}{2}}$$

$$x * g = \sum_{k=0}^{K} \theta_k T_k(\check{L}) x$$

$$\check{L} = \frac{2}{\lambda_{\max}} \mathcal{L} - I$$

$$T_k(\check{L}) = 2\check{L}\widetilde{x_{k-1}} - \widetilde{x_{k-2}}$$

$\check{x}_0 = x, \check{x}_1 = \check{L}x$　漸化式

$\mathcal{L} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$より

$$x' = D\check{}^{-\frac{1}{2}} A D\check{}^{-\frac{1}{2}} \theta x$$

今後について

- Graph Neural Convolutional が画像の畳み込みでも
- 複雑なタスクについて
- 層数を増やす

```
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
torch.Size([34, 5])
```