Diffusion Improves Graph Learning

Type 1

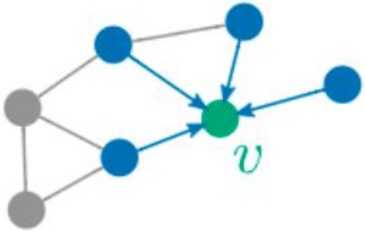
Assem Zhunis 20170906

Problem Statement

Graph convolution is usually done by passing message between direct neighbors.

Limitations:

- Only 1-hop neighbors. Severe limitation, real graphs are noisy.
- Real graphs are usually homophilic: neighbors are similar.



$$m_v^{(t+1)} = \sum_{w \in \mathcal{N}(v)} f_{\text{message}}^{(t)}(h_v^{(t)}, h_w^{(t)}, e_{vw})$$

$$h_v^{(t+1)} = f_{\text{update}}^{(t)}(h_v^{(t)}, m_v^{(t+1)})$$

Problem Statement

Graph convolution is usually done by passing message between direct neighbors.

Limitations:

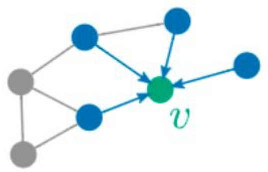
- Only 1-hop neighbors. Severe limitation, real graphs are noisy.
- Real graphs are usually homophilic: neighbors are similar.

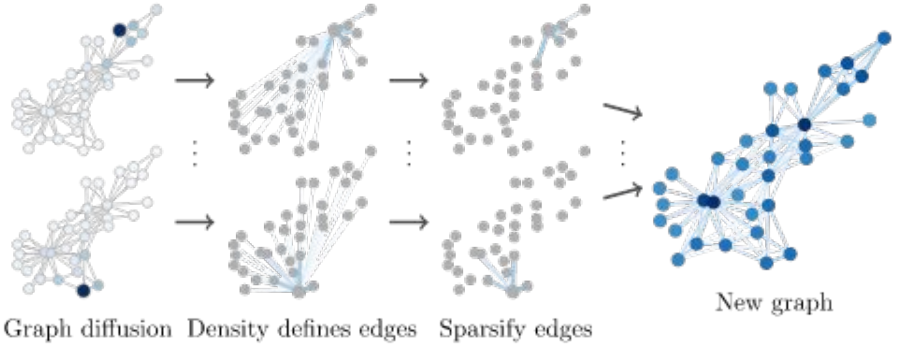
Your friends are likely to have similar interests.

Proposed method

This restriction can be removed by the **GDC** (Graph diffusion convolution) method proposed by J Klicpera et al.

- Generate more informative neighborhood by graph diffusion.



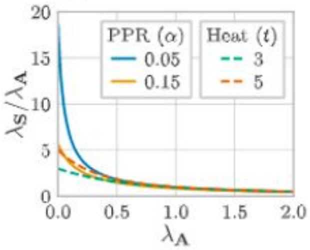


Klicpera, Johannes, Stefan Weißenberger, and Stephan Günnemann. "Diffusion improves graph learning." *arXiv preprint arXiv:1911.05485* (2019).

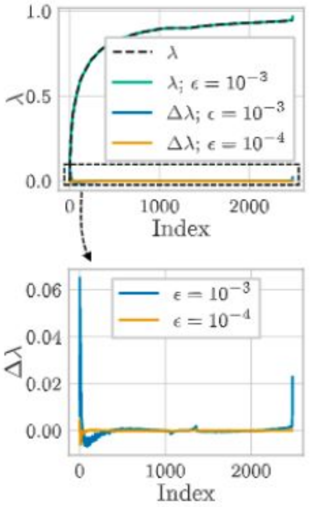
Why does GDC work?

GDC = Denoising filter

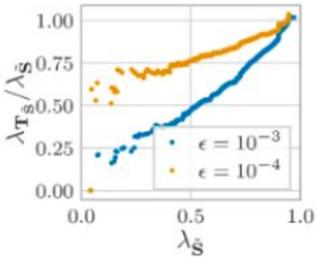
1. Graph diffusion
Low-pass filter

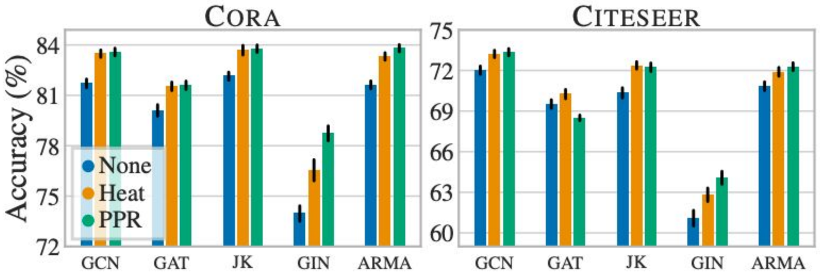


2. Sparsification
Almost no impact on eigenvalues



3. Transition matrix
Weak high-pass filter





High-level Idea

Conducting multiple experiments on GDC with different diffusion coefficients.

Paper used:

- PPR $\longrightarrow$ $\theta_k^{\text{PPR}} = \alpha(1-\alpha)^k$
- Heat kernel $\longrightarrow$ $\theta_k^{\text{HK}} = e^{-t}\frac{t^k}{k!}$

$$\boldsymbol{S} = \sum_{k=0}^{\infty} \theta_k \boldsymbol{T}^k,$$

Weighting coefficient

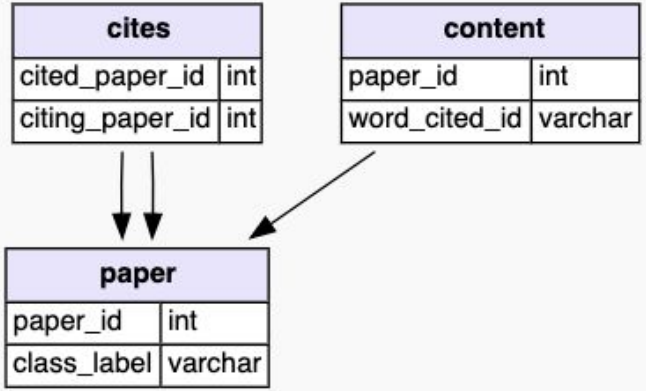
Fouss, François, et al. "An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification." *Neural networks* 31 (2012): 53-72.

Kernels for experiments

- Adjacency matrix based kernels:
  - **Katz**: Katz kernel (a.k.a. Walk, Von Neumann diffusion kernel)
  - **Comm**: Communicability kernel (a.k.a. Exponential diffusion kernel)
  - **DFS**: Double Factorial similarity
- Laplacian based kernels:
  - **For**: Forest kernel (a.k.a. Regularized Laplacian kernel)
  - **Heat:** Heat kernel (a.k.a. Laplacian exponential diffusion kernel)
  - **NHeat**: Normalized Heat kernel
  - **Abs**: Absorption kernel
- Markov matrix based kernels and measures:
  - **PPR**: Personalized PageRank
  - **MPPR**: Modified Personalized PageRank
  - **HPR**: PageRank heat similarity measure
- Sigmoid Commute Time:
  - **SCT**: Sigmoid Commute Time
  - **CCT**: Corrected Commute Time
  - **SCCT**: Sigmoid Corrected Commute Time

Datasets:

1. Cora



CORA

The Cora dataset consists of 2708 scientific publications classified into one of seven classes. The citation network consists of 5429 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1433 unique words.

Original source: linqs.cs.umd.edu

Versions

CORA (by Arnaud Barragao)

Datasets:
2. Citeseer

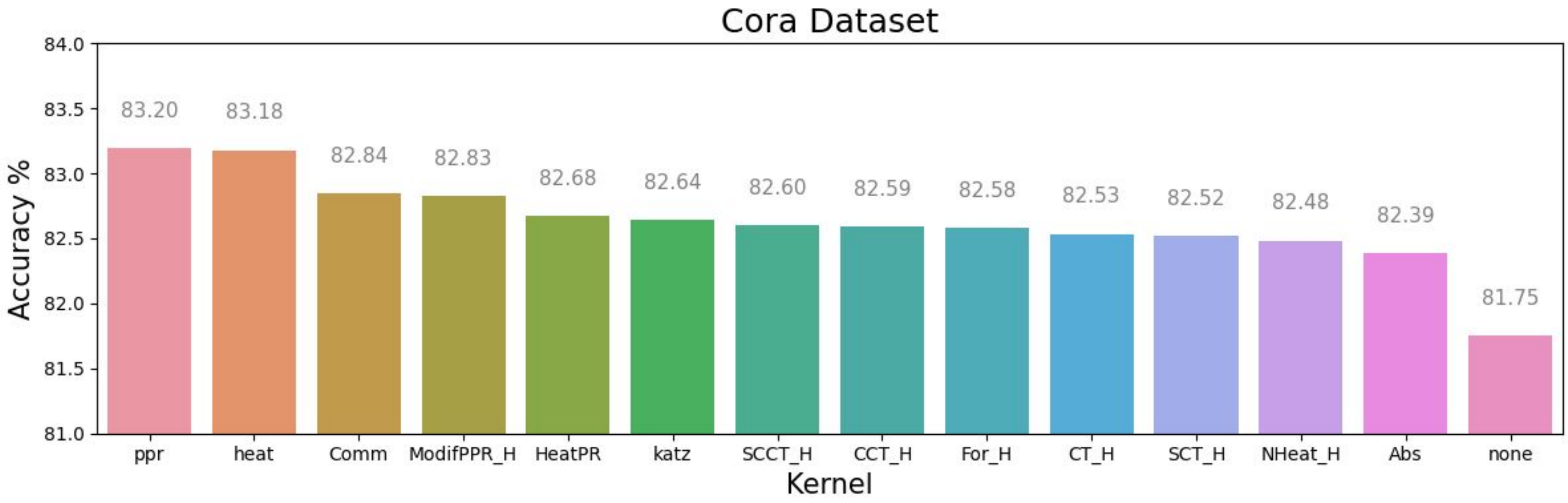
- **CiteSeer for Document Classification**
  - The CiteSeer dataset consists of 3312 scientific publications classified into one of six classes. The citation network consists of 4732 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 3703 unique words. The README file in the dataset provides more details.
  - Download link:
    - https://linqs-data.soe.ucsc.edu/public/lbc/citeseer.tgz
  - Related papers:
    - Qing Lu, and Lise Getoor. "Link-based classification." ICML, 2003.
    - Prithviraj Sen, et al. "Collective classification in network data." AI Magazine, 2008.

Method:

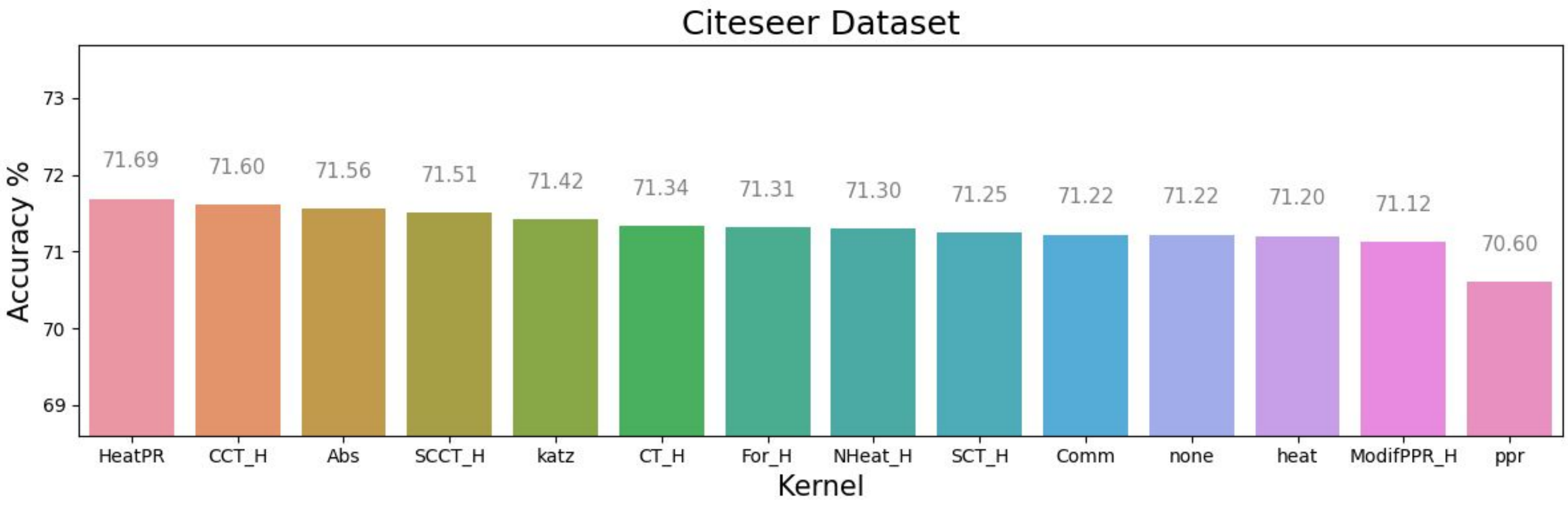
1. Preprocess 2 datasets with 13 kernels
2. Use GCN model for classification
3. For each kernel run with 20 different seeds

```
elif kernel == 'katz':
    return np.linalg.pinv(np.eye(num_nodes) - t * adj_matrix) #katz
elif kernel == 'Comm':
    return expm(t * adj_matrix) #Comm
elif kernel == 'CT_H':
    return np.linalg.pinv(L) # CT_H
elif kernel == 'For_H':
    return np.linalg.inv(np.eye(num_nodes) + t * L) #For_H
elif kernel == 'NHeat_H':
    D_12 = np.linalg.inv(np.sqrt(D))
    nL = D_12.dot(L).dot(D_12)
    return expm(-t * nL) #NHeat_H
elif kernel == "SCT_H":
    K_CT = np.linalg.pinv(L)
    sigma = K_CT.std()
    EPS  = 10 ** -10
    Kds = K_CT / (sigma + EPS) #EPS 10 ** -10
    return 1. / (1. + np.exp(-0.05 * Kds)) # SCT_H
elif kernel == 'ModifPPR_H':
    D = np.diag(np.sum(adj_matrix, axis=0)) # degree matrix
    return np.linalg.inv(D - 0.05 * adj_matrix) #ModifPPR_H
elif kernel == 'HeatPR':
    D = np.diag(np.sum(adj_matrix, axis=0)) # degree matrix
    P = np.linalg.inv(D).dot(adj_matrix)
    return expm(-t * (I - P)) # HeatPR
elif kernel == 'Abs':
    return np.linalg.pinv(t * adj_matrix + L) #Abs
```

Results: Cora

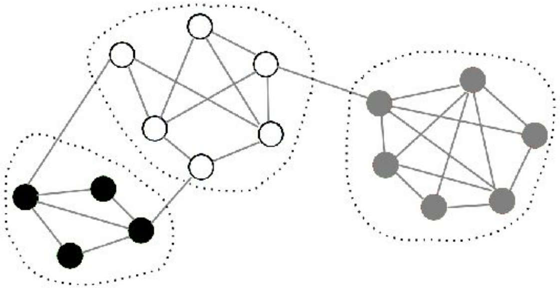


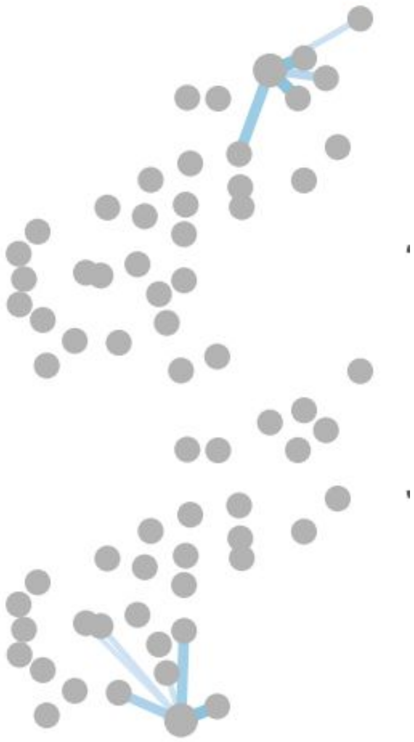
Result: Citeseer



Future improvements

- Use different sparsification techniques
- Try clustering instead of sparsification





Sparsify edges

Conclusion

- Message passing in the GNN can be enhanced by Graph Diffusion Convolution (GDC).
- Diffusion kernels used by GDC can be chosen depending on the dataset
  - For Cora: heat and PPR are the best
  - For Citeseer: heatPR overperformed authors’ results
- Results are available:
  - https://github.com/assemzh/Graphs_diffusion