Edge Varying Graph Neural Networks And Their Stability

Elvin Isufi

e.isufi-1@tudelft.nl

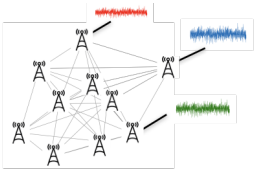
April 25, 2024

Acknowledgements

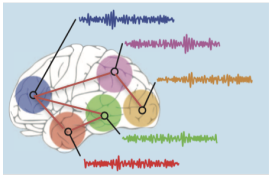
- Collaborators
  - ⇒ Zhan Gao
  - ⇒ Mario Coutino
  - ⇒ Fernando Gama
  - ⇒ Amanda Prorok
  - ⇒ Alejandro Ribeiro
  - ⇒ Geert Leus
- Funding
  - ⇒ NWO – GraSPA project
  - ⇒ TU Delft AI Innitiative
  - ⇒ TU Delft - Shell AI – TKI project: GNN for Renewable Energy

Graphs & Data

Capture nontrivial complex relationships between nodes



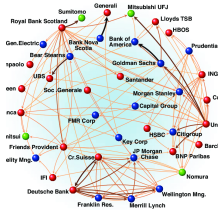
Communication



Brain

Social

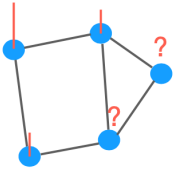
Water networks



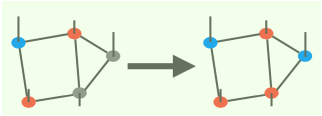
Financial

Transport

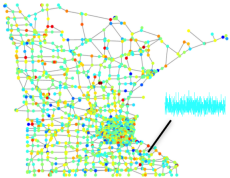
Data Processing and Learning Tasks



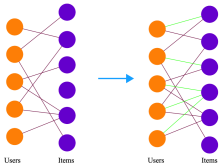
Reconstruction/interpolation Ex. Find missing values in a sensor network



Node classification
Ex. Classify users of a social network from their friends labels



Prediction
Ex. Predict traffic



Link prediction
Ex. Recommend items

Graph Neural Networks

- De facto state-of-the-art solution for graph-based learning tasks

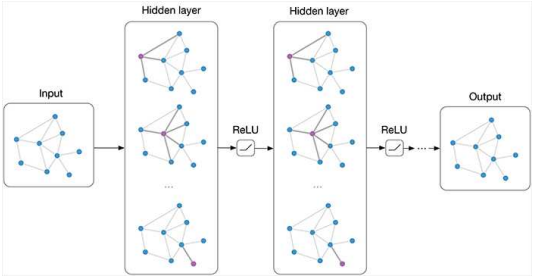


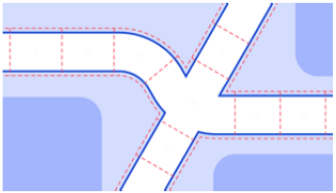
fig. credit T. Kipf

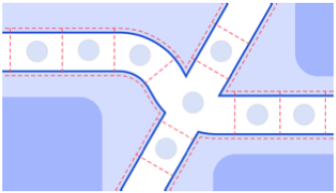
- Composition of filter banks and non-linearities
- Leverage the graph as an inductive bias

P. Battaglia, et al., Relational inductive biases, deep learning, and graph networks. arXiv:1806.01261, 2018

Graph Neural Networks

Regular success stories





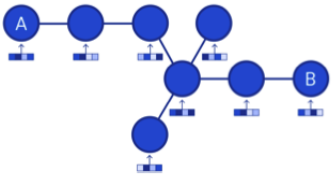
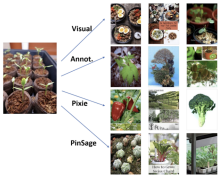


fig. credit DeepMind



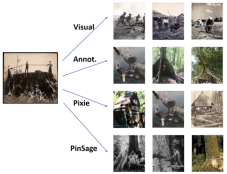


fig. credit J. Leskovec

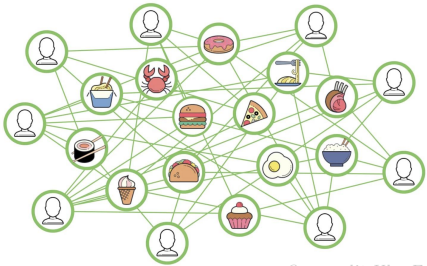
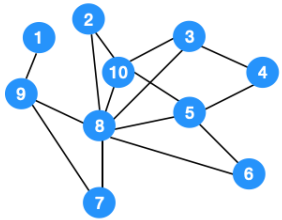


fig. credit UberEats

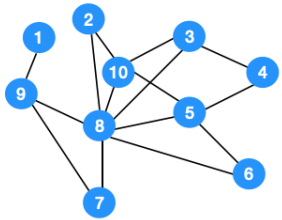
GNN Open Directions

- GNN architectures: There is a surge of GNN architectures
  - ⇒ Graph convolutional networks
  - ⇒ Graph attention networks
  - ⇒ Various orthogonal polynomial networks (ChebNet, JacobiNet...)



GNN Open Directions

- GNN architectures: There is a surge of GNN architectures
  - ⇒ Graph convolutional networks
  - ⇒ Graph attention networks
  - ⇒ Various orthogonal polynomial networks (ChebNet, JacobiNet...)



- Q: How to represent them in a unified framework to highlight the advantages and limitations of each solution?

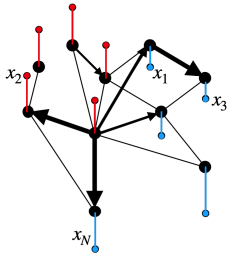
Outline

- Graph Convolutional Neural Networks
- Edge Varying Graph Neural Networks
- Representational Capacity - Stability Trade-off
- Conclusion & Future Work

Graph Convolutional Neural Networks

Graphs and Signals

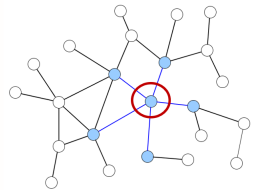
- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ $\Rightarrow$ $\mathcal{V}$: set of nodes, $\mathcal{E}$: set of edges
  - $\Rightarrow$ Graph shift operator $\mathbf{S}$ (Adjacency matrix $\mathbf{A}$, Laplacian matrix $\mathbf{L}$)
  - $\Rightarrow$ Graph signals: Associate a value to each node $\mathbf{x} : \mathcal{V} \to \mathbb{R}$



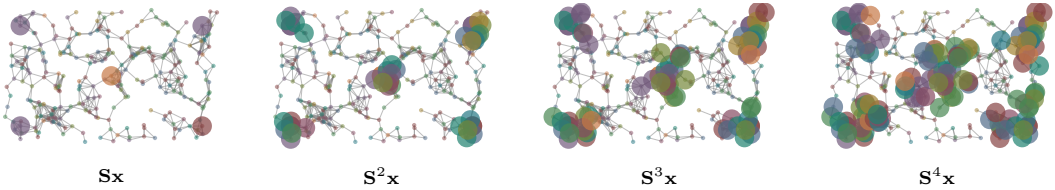
- Graph shift operation can be locally computed at each node i

$$[\mathbf{S}\mathbf{x}]_i = \sum_{j=1}^{n} [\mathbf{S}]_{ij}[\mathbf{x}]_i = \sum_{j \in \mathcal{N}_i} [\mathbf{S}]_{ij}[\mathbf{x}]_i$$

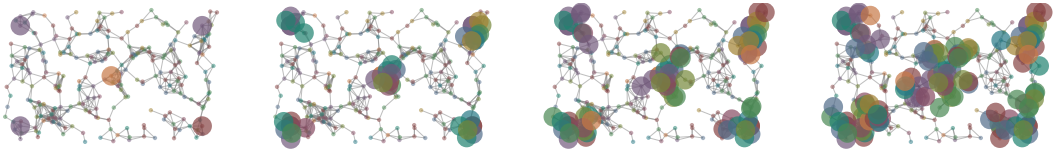
  - $\Rightarrow$ exploits the sparsity and locality of the graph



Graph Convolutional Filter



Graph Convolutional Filter



- Graph convolutional filter $\Rightarrow$ Linear combination of shifted versions of the signal

$$\mathbf{H}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \, \mathbf{S}^k \mathbf{x}$$

$\Rightarrow$ Local operation

Graph Convolutional Neural Networks

- Graph filters have limited expressive power because they can only learn linear maps
- An alternative choice is the graph perceptron $\Rightarrow \mathbf{\Phi}(\mathbf{x}) = \mathbf{\Phi}(\mathbf{x}; \mathbf{S}, \mathbf{h}) = \sigma\left[\sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x}\right]$

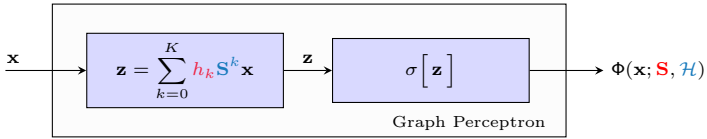


fig. credit A. Ribeiro

Graph Convolutional Neural Networks

- Graph filters have limited expressive power because they can only learn linear maps
- An alternative choice is the graph perceptron $\Rightarrow \mathbf{\Phi}(\mathbf{x}) = \mathbf{\Phi}(\mathbf{x}; \mathbf{S}, \mathbf{h}) = \sigma\left[\sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x}\right]$

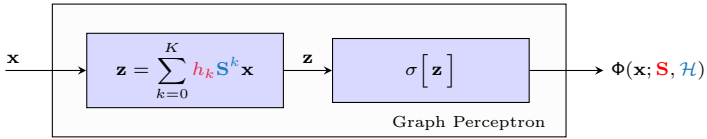


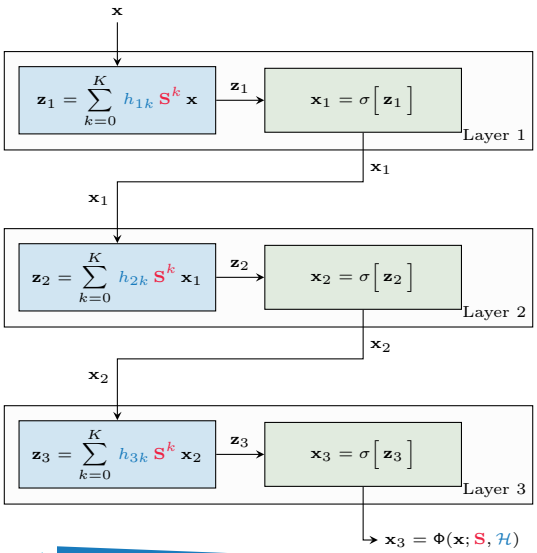
fig. credit A. Ribeiro

- Optimal regressor restricted to perceptron class $\Rightarrow \mathbf{h}^\star = \arg\min_{\mathbf{h}} \sum_{\mathbf{x} \in \mathcal{T}} \mathcal{L}\Big(\mathbf{\Phi}(\mathbf{x}; \mathbf{S}, \mathbf{h})\Big)$

  $\Rightarrow$ Perceptron allows learning of nonlinear maps $\Rightarrow$ More expressive.

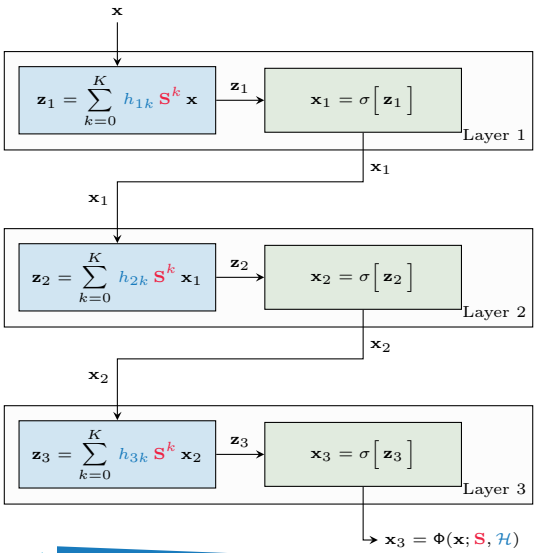
Gama, I., Leus, & Ribeiro, Graphs, convolutions, and neural networks. IEEE SPM, 2020.

Graph Convolutional Neural Network



- Cascade a few graph perceptrons (3 in the figure)
- Last layer output is the GNN output $\Rightarrow \mathbf{\Phi}(\mathbf{x}; \mathbf{S}, \mathcal{H})$
  $\Rightarrow$ Parametrized on filter tensor $\mathcal{H} = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$

Graph Convolutional Neural Network

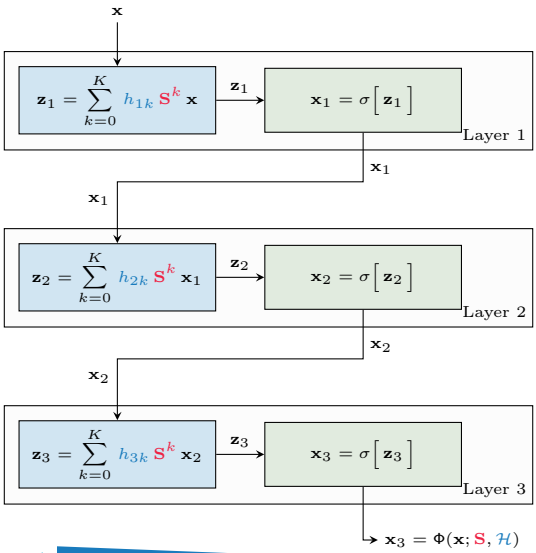


- Cascade a few graph perceptrons (3 in the figure)
- Last layer output is the GNN output $\Rightarrow \mathbf{\Phi}(\mathbf{x}; \mathbf{S}, \mathcal{H})$
  $\Rightarrow$ Parametrized on filter tensor $\mathcal{H} = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$
- Layer ℓ propagation rule

$$\mathbf{x}_\ell = \sigma[\mathbf{z}_\ell] = \sigma\left[\sum_{k=0}^{K} h_{\ell k} \mathbf{S}^k \mathbf{x}_{\ell-1}\right]$$

  $\Rightarrow$ Trainable parameters: $\mathcal{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_L]$

Graph Convolutional Neural Network



- Cascade a few graph perceptrons (3 in the figure)
- Last layer output is the GNN output $\Rightarrow \boldsymbol{\Phi}(\mathbf{x}; \mathbf{S}, \mathcal{H})$
  $\Rightarrow$ Parametrized on filter tensor $\mathcal{H} = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$
- Layer ℓ propagation rule

$$\mathbf{x}_\ell = \sigma[\mathbf{z}_\ell] = \sigma\left[\sum_{k=0}^{K} h_{\ell k}\mathbf{S}^k\mathbf{x}_{\ell-1}\right]$$

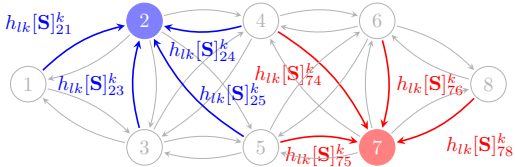
  $\Rightarrow$ Trainable parameters: $\mathcal{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_L]$

- The graph is in the filter component of the GNN
  $\Rightarrow$ Change the filter $\Rightarrow$ change the GNN
  $\Rightarrow$ Analyze the filter $\Rightarrow$ obtain insights on the GNN

Parameter Sharing in GCNNs

- Graph convolutions share a scalar weight h_k on all neighbors
  $\Rightarrow$ Consider indifference towards the values of different neighbors

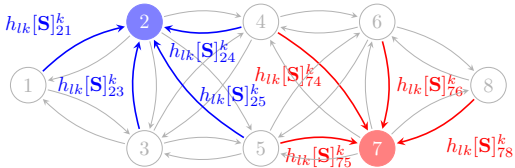
$$\mathbf{H}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \, \mathbf{S}^k \mathbf{x}$$



Parameter Sharing in GCNNs

- Graph convolutions share a scalar weight h_k on all neighbors
  - $\Rightarrow$ Consider indifference towards the values of different neighbors

$$\mathbf{H}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \, \mathbf{S}^k \mathbf{x}$$



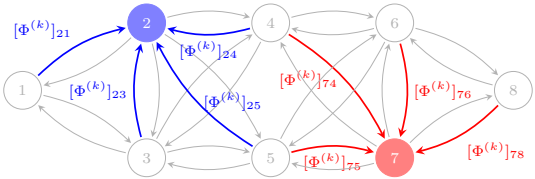
- Advantages:
  - $\Rightarrow$ Limits the number of trainable parameters
  - $\Rightarrow$ Allows permutation equivariance and inductive transfer
- Limitations: Limited representational capacity, i.e., the expressive power

Edge Varying Graph Neural Networks

Edge Varying Graph Neural Network

- Edge varying Graph Neural Networks (EdgeNets) are GNNs with edge varying filters (EdgeGFs)

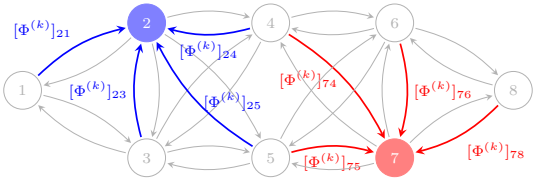
  $\Rightarrow$ Different parameters per node and edge



Edge Varying Graph Neural Network

- Edge varying Graph Neural Networks (EdgeNets) are GNNs with edge varying filters (EdgeGFs)

  $\Rightarrow$ Different parameters per node and edge



- EdgeGF is a linear mapping of graph signals

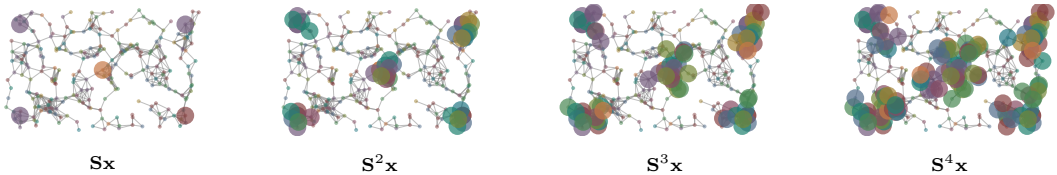
$$\mathbf{H}_{\text{Edge}}(\mathbf{S})\mathbf{x} := \sum_{k=0}^{K} \mathbf{\Phi}^{(k)} \mathbf{S}^k \mathbf{x}$$

  $\Rightarrow$ Filter parameter matrices $\mathbf{\Phi}^{(k)}$ share the same support of $\mathbf{S} + \mathbf{I}$

Coutino, I. & Leus, G. Advances in distributed graph filtering. IEEE T-SP, 2019

Edge Varying Graph Neural Network

- EdgeGF allocates parameters per edge on shifted signals $\mathbf{S}\mathbf{x}$, $\mathbf{S}^2\mathbf{x}$, ..., $\mathbf{S}^K\mathbf{x}$
  - $\Rightarrow$ Diffuse $\mathbf{x}$ over $\mathbf{S}$ to collect multi-hop neighborhood information up to a radius K
  - $\Rightarrow$ Give different importance to the information coming from different neighbors.

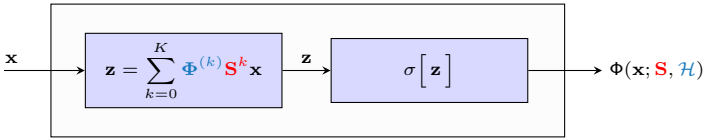


- Each edge weight matrix $\mathbf{\Phi}^{(k)}$ is parametric / learnable

Edge Varying Graph Neural Network

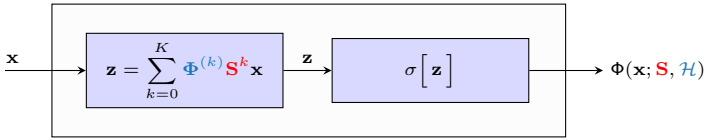
- EdgeNet is composed of EdgeGFs and pointwise nonlinearity

  $\Rightarrow$ Replace graph convolutional filters with EdgeGFs at each layer



Edge Varying Graph Neural Network

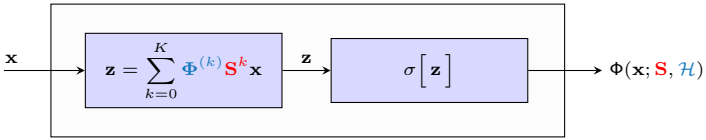
- EdgeNet is composed of EdgeGFs and pointwise nonlinearity
  - $\Rightarrow$ Replace graph convolutional filters with EdgeGFs at each layer



- Advantages: Adapts the edge weights to the task
  - $\Rightarrow$ more flexibility for the parameter space – better representational capacity to capture local details
  - $\Rightarrow$ Fewer parallel filters and shallower networks

Edge Varying Graph Neural Network

- EdgeNet is composed of EdgeGFs and pointwise nonlinearity
  - ⇒ Replace graph convolutional filters with EdgeGFs at each layer



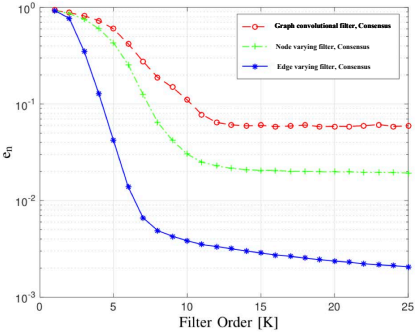
- Advantages: Adapts the edge weights to the task
  - ⇒ more flexibility for the parameter space – better representational capacity to capture local details
  - ⇒ Fewer parallel filters and shallower networks
- Limitations: Complexity of order $\mathcal{O}(MKF^2L)$
  - ⇒ Architecture complexity depends on graph size
  - ⇒ EdgeNet may overfit and require more training data (penalize $\|\mathbf{\Phi}^{(k)}\|_2$)

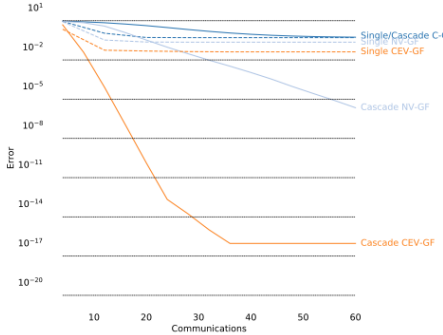
I., Gama, & Ribeiro EdgeNets: Edge varying graph neural networks. IEEE T-PAMI, 2021

Distributed Approximate Finite-Time Consensus

- Consensus matrix $\tilde{\mathbf{H}} = \mathbf{1}\mathbf{1}^\top/n$ with graph size $n = 256$

  $\Rightarrow$ Performance is measured by the fitness error $\mathbf{e}_n = \|\tilde{\mathbf{H}} - \mathbf{H}_{fit}(\boldsymbol{\Theta})\|$





- Machine precision can be achieved in tens of iterations with linear layers

Coutino, & Leus. A cascaded structure for generalized graph filters. IEEE T-SP, 2021

Results: Edge varying Graph Neural Network

- Source localization: Find which subnetwork is the source of a diffused signal
- Authorship attribution: Classify if a text belongs to a specific author or not

Architecture	mean	std. dev.
GCNN	4.0%	13.0%
Edge varying	**1.5%**	8.4%
Node varying	6.0%	15.8%
Hybrid edge var.	6.6%	15.9%

Architecture	Austen	Brontë	Poe
GCNN	7.2(±2.0)%	12.9(±3.5)%	14.3(±6.4)%
Edge varying	7.1(±2.2)%	13.1(±3.9)%	**10.7(±4.3)%**
Node varying	7.4(±2.1)%	14.6(±4.2)%	11.7(±4.9)%
Hybrid edge var.	**6.9(±2.6)%**	14.0(±3.7)%	11.7(±4.8)%

- When the data is scarse (authorships) we need to resort to hybrid solutions
  $\Rightarrow$ Limit the number of parameters

I., Gama, & Ribeiro EdgeNets: Edge varying graph neural networks. IEEE T-PAMI, 2021

Results: Edge varying Graph Neural Network

- Signal denoising: Unrolling GNNs with edgevarying filters
- Forecasting: Nonlinear autoregressive models with edge varying filters

Metric	Temperature NMSE		Traffic NMSE		cora Error Rate		cora F1 score	
Method	1	365	1	24	1	7	1	7
Baseline	0.34	0.377	0.392	0.377	0.095	0.099	0.829	0.695
GLD	0.045	0.024	0.248	0.255	0.055	0.032	0.609	0.793
GTF	0.079	0.036	0.202	0.176	0.06	0.039	0.484	0.532
GFT	0.065	0.053	0.257	0.231	0.079	0.053	0.459	0.489
SGWT	0.069	0.11	0.184	0.162	0.074	0.073	0.551	0.569
QMF	0.26	0.31	**0.18**	0.185	0.087	0.087	0.51	0.512
CSFB	0.07	0.061	0.344	0.36	0.129	0.143	0.43	0.437
MLP	0.142	0.027	0.31	0.169	0.095	0.072	0.829	0.695
GCN	0.041	0.033	0.293	0.279	0.042	0.025	0.903	0.901
GAT	0.044	0.031	0.267	0.264	0.041	0.032	0.909	0.873
GUSC	**0.037**	**0.016**	0.324	0.178	**0.04**	**0.024**	**0.91**	**0.906**
GUTF	0.053	0.019	0.266	**0.158**	0.041	0.03	0.906	0.885

Models	Solar			Wiki			Traffic		
	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)
VAR Watson [1993]	0.184	0.234	577.10	0.057	0.094	96.58	0.535	1.133	550.12
SFM Zhang et al. [2017]	0.161	0.283	362.89	0.081	0.156	104.47	0.029	0.044	59.33
LSTNet Lai et al. [2018]	0.148	0.200	132.95	0.054	0.090	118.24	0.026	0.057	**25.77**
TCN Bai et al. [2018]	0.176	0.222	142.23	0.094	0.142	99.66	0.052	0.067	-
DeepGLO Sen et al. [2019]	0.178	0.400	346.78	0.110	0.113	119.60	0.025	0.037	33.32
Reformer Kitaev et al. [2020]	0.234	0.292	128.58	0.048	0.085	73.61	0.029	0.042	112.58
Informer Zhou et al. [2021]	0.151	0.199	128.45	0.051	0.086	80.50	0.020	0.033	59.34
Autoformer Wu et al. [2021]	0.150	0.193	103.79	0.069	0.103	121.90	0.029	0.043	100.02
FEDformer Zhou et al. [2022]	0.139	0.182	**100.92**	0.068	0.098	123.10	0.025	0.038	85.12
GraphWaveNet Wu et al. [2019]	0.183	0.238	603	0.061	0.105	136.12	0.013	0.034	33.78
StemGNN Cao et al. [2020]	0.176	0.222	128.39	0.190	0.255	117.92	0.080	0.135	64.51
MTGNN Wu et al. [2020]	0.151	0.207	507.91	0.101	0.140	122.96	0.013	0.030	29.53
AGCRN Bai et al. [2020]	0.123	0.214	353.03	0.044	0.079	78.52	0.084	0.166	31.73
EV-FGN(ours)	**0.120**	**0.162**	116.48	**0.041**	**0.076**	**64.50**	**0.011**	**0.023**	28.71
Improvement	2.4%	11.0%	-	6.8%	3.8%	12.4%	15.4%	23.3%	-

- Versatile solution in different tasks

  $\Rightarrow$ Often we need to work with subcategories of EdgeNets

Chen, Eldar, & Zhao. Graph unrolling networks: Interpretable neural networks for graph signal denoising. IEEE T-SP, 2021

Zhang et al., Edge-Varying Fourier Graph Networks for Multivariate Time Series Forecasting. arXiv:2210.03093, 2022

Edge varying Graph Neural Network

- EdgeNet: a general framework that unifies a set of GNN solutions
  - $\Rightarrow$ Assign different weights to different edges at different shifts.
  - $\Rightarrow$ Imposing different restrictions on edge weight matrices yields different GNNs

Edge varying Graph Neural Network

- EdgeNet: a general framework that unifies a set of GNN solutions
  - $\Rightarrow$ Assign different weights to different edges at different shifts.
  - $\Rightarrow$ Imposing different restrictions on edge weight matrices yields different GNNs

- Graph attention networks (GATs): Parameterize the EV learnable matrices with attention
  - $\Rightarrow$ GATs act as order one convolutions on a learnable graph

Edge varying Graph Neural Network

- EdgeNet: a general framework that unifies a set of GNN solutions
  - $\Rightarrow$ Assign different weights to different edges at different shifts.
  - $\Rightarrow$ Imposing different restrictions on edge weight matrices yields different GNNs

- Graph attention networks (GATs): Parameterize the EV learnable matrices with attention
  - $\Rightarrow$ GATs act as order one convolutions on a learnable graph
- Natural graph convolutions: Allow for EV parameters to a few nodes

Edge varying Graph Neural Network

- EdgeNet: a general framework that unifies a set of GNN solutions
  - $\Rightarrow$ Assign different weights to different edges at different shifts.
  - $\Rightarrow$ Imposing different restrictions on edge weight matrices yields different GNNs

- Graph attention networks (GATs): Parameterize the EV learnable matrices with attention
  - $\Rightarrow$ GATs act as order one convolutions on a learnable graph
- Natural graph convolutions: Allow for EV parameters to a few nodes
- Multi-hop GATs: Generalize GATs in a principled way towards fully EV solutions

Edge varying Graph Neural Network

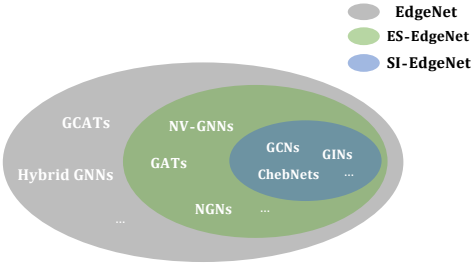
- EdgeNet: a general framework that unifies a set of GNN solutions
  - $\Rightarrow$ Assign different weights to different edges at different shifts.
  - $\Rightarrow$ Imposing different restrictions on edge weight matrices yields different GNNs

- Graph attention networks (GATs): Parameterize the EV learnable matrices with attention
  - $\Rightarrow$ GATs act as order one convolutions on a learnable graph
- Natural graph convolutions: Allow for EV parameters to a few nodes
- Multi-hop GATs: Generalize GATs in a principled way towards fully EV solutions
- Each offers different restrictions of the parameter space – limits the representational capacity

Velickovic et al., Graph attention networks. ICLR, 2018
de Haan, Cohen, Welling, Natural graph convolutions, NeurIPS, 2020
Wang, Ying, Huang, & Leskovec Multi-hop attention graph neural network, arXiv:2009.14332., 2020

General Edge Varying Graph Neural Network

- General EdgeNet imposes no restriction on edge varying parameter matrices
  $\Rightarrow$ Enjoys the strongest representational capacity
- Taxonomize based on degrees of freedom of the parameter space
  $\Rightarrow$ Proxy for representational capacity



Gao, Prorok, & I., On the Trade-Off between Stability and Representational Capacity in Graph Neural Networks, arXiv, 2023

Shift-Invariant Edge Varying Graph Neural Network

- SI-EdgeNet consists of EdgeGFs that are invariant to graph shift operation

$$\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{S}\mathbf{x} = \mathbf{S}\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{x}$$

  $\Rightarrow$ Require the eigenvectors of $\mathbf{\Phi}^{(k)}$ to coincide with those of $\mathbf{S}$

Shift-Invariant Edge Varying Graph Neural Network

- SI-EdgeNet consists of EdgeGFs that are invariant to graph shift operation

$$\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{S}\mathbf{x} = \mathbf{S}\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{x}$$

  $\Rightarrow$ Require the eigenvectors of $\mathbf{\Phi}^{(k)}$ to coincide with those of $\mathbf{S}$

- GCNNs are in the subclass of SI-EdgeNets $\mathbf{\Phi}^{(k)} = h_k \mathbf{I}$

$$\mathbf{H}_{\mathrm{SI,Conv}}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \, \mathbf{S}^k \mathbf{x}$$

Shift-Invariant Edge Varying Graph Neural Network

- SI-EdgeNet consists of EdgeGFs that are invariant to graph shift operation

$$\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{S}\mathbf{x} = \mathbf{S}\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{x}$$

  $\Rightarrow$ Require the eigenvectors of $\mathbf{\Phi}^{(k)}$ to coincide with those of $\mathbf{S}$

- GCNNs are in the subclass of SI-EdgeNets $\mathbf{\Phi}^{(k)} = h_k\mathbf{I}$

$$\mathbf{H}_{\mathrm{SI,Conv}}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k\ \mathbf{S}^k\mathbf{x}$$

- Transferable to unseen graphs – apply the trained model on different graphs during testing

Shift-Invariant Edge Varying Graph Neural Network

- SI-EdgeNet consists of EdgeGFs that are invariant to graph shift operation

$$\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{S}\mathbf{x} = \mathbf{S}\mathbf{H}_{\mathrm{SI,Edge}}(\mathbf{S})\mathbf{x}$$

  $\Rightarrow$ Require the eigenvectors of $\mathbf{\Phi}^{(k)}$ to coincide with those of $\mathbf{S}$

- GCNNs are in the subclass of SI-EdgeNets $\mathbf{\Phi}^{(k)} = h_k\mathbf{I}$

$$\mathbf{H}_{\mathrm{SI,Conv}}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k\ \mathbf{S}^k\mathbf{x}$$

- Transferable to unseen graphs – apply the trained model on different graphs during testing
- However, it restricts the parameter space – representation capacity

Shift-Invariant Edge Varying Graph Neural Network

- SI-EdgeNet consists of EdgeGFs that are invariant to graph shift operation

$$\mathbf{H}_{\text{SI,Edge}}(\mathbf{S})\mathbf{S}\mathbf{x} = \mathbf{S}\mathbf{H}_{\text{SI,Edge}}(\mathbf{S})\mathbf{x}$$

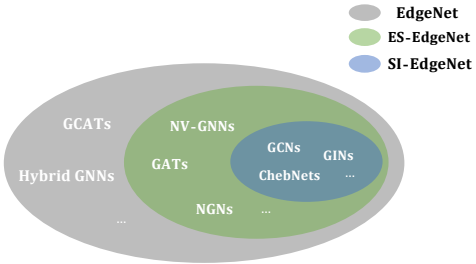
  $\Rightarrow$ Require the eigenvectors of $\mathbf{\Phi}^{(k)}$ to coincide with those of $\mathbf{S}$

- GCNNs are in the subclass of SI-EdgeNets $\mathbf{\Phi}^{(k)} = h_k\mathbf{I}$

$$\mathbf{H}_{\text{SI,Conv}}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k\ \mathbf{S}^k\mathbf{x}$$

- Transferable to unseen graphs – apply the trained model on different graphs during testing
- However, it restricts the parameter space – representation capacity
- Other GNNs in the SI-EdgeNet subclass include GCNs, ChebNets, JKNets, and GINs, ...

Edge Varying Graph Neural Network



- Each subclass restricts the parameter space

Eigenvector-Sharing Edge Varying Graph Neural Network

- ES-EdgeNet require edge weight matrices $\mathbf{\Phi}^{(k)}$ to share the eigenvectors $\mathbf{U}$
  $\Rightarrow$ $\mathbf{U}$ can be different from the eigenvectors of $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$

Eigenvector-Sharing Edge Varying Graph Neural Network

- ES-EdgeNet require edge weight matrices $\mathbf{\Phi}^{(k)}$ to share the eigenvectors $\mathbf{U}$
  $\Rightarrow$ $\mathbf{U}$ can be different from the eigenvectors of $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$

- Node varying GNNs are one instance of ES-EdgeNets

$$\mathbf{H}_{\mathrm{ES,NV}}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} \mathbf{D}^{(k)}\mathbf{S}^{k}\mathbf{x}.$$

  $\Rightarrow$ Edge weight matrices is the diagonal matrices $\mathbf{\Phi}^{(k)} = \mathbf{D}^{(k)}$
  $\Rightarrow$ Aggregate neighborhood information in a way that they treat node features differently

Eigenvector-Sharing Edge Varying Graph Neural Network

- ES-EdgeNet require edge weight matrices $\mathbf{\Phi}^{(k)}$ to share the eigenvectors $\mathbf{U}$
  $\Rightarrow$ $\mathbf{U}$ can be different from the eigenvectors of $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$

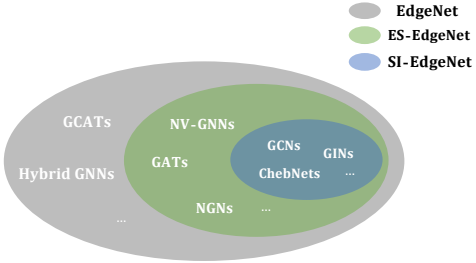
- Node varying GNNs are one instance of ES-EdgeNets

$$\mathbf{H}_{\mathrm{ES,NV}}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} \mathbf{D}^{(k)}\mathbf{S}^{k}\mathbf{x}.$$

  $\Rightarrow$ Edge weight matrices is the diagonal matrices $\mathbf{\Phi}^{(k)} = \mathbf{D}^{(k)}$
  $\Rightarrow$ Aggregate neighborhood information in a way that they treat node features differently

- Other GNNs in the ES-EdgeNet subclass include GATs, NGNs, ...

Edge Varying Graph Neural Network

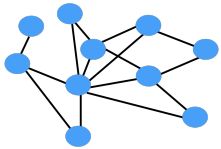


- Each subclass restricts the parameter space
- Q: What are the implications the stability of GNNs?

Representational Capacity – Stability Trade-Off

Spectral Analysis of GCNNs

- Investigate the filter behavior in the spectral domain
  $\Rightarrow$ Rely on tools from graph signal processing
- GSO eigendecomposition: $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$



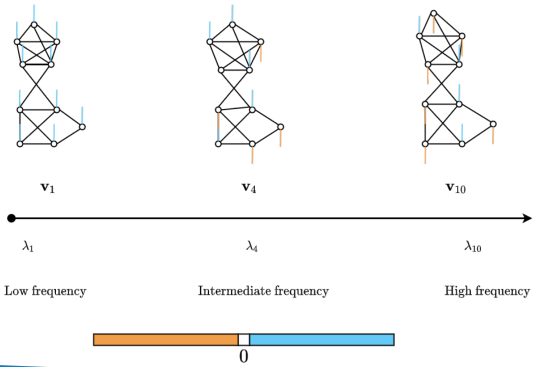
Eigendecomposition of GSO matrix

$$\mathbf{S} = \begin{pmatrix} | & & | \\ \mathbf{v}_1 & \dots & \mathbf{v}_N \\ | & & | \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_N \end{pmatrix} \begin{pmatrix} - & \mathbf{v}_1 & - \\ & \vdots & \\ - & \mathbf{v}_N & - \end{pmatrix}$$

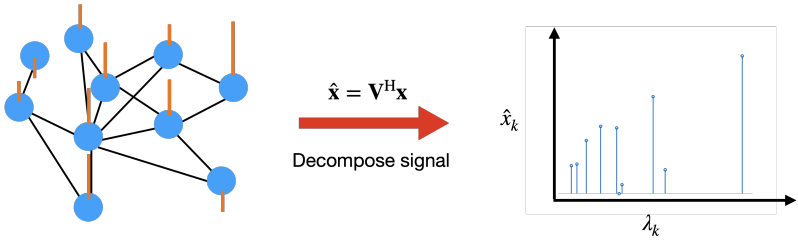
- $\mathbf{\Lambda} = \text{diag}([\lambda_1, \dots, \lambda_N])$: graph eigenvalues (frequencies)
- $\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_N)$: graph modes (analogous to complex exponentials)

- Work with (any) Laplacian matrix: $\mathbf{S} = \mathbf{L}$ $\Rightarrow$ Symmetric (Positive Semidefinite)
  - $\Rightarrow$ Always: $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$
  - $\Rightarrow$ Real positive eigenvalues $\Rightarrow 0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$
- Eigenvector variability: $\mathbf{v}_i^{\mathrm{H}}\mathbf{L}\mathbf{v}_i = \lambda_i \quad 0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N \qquad \mathbf{L}\mathbf{v}_i = \lambda_i$

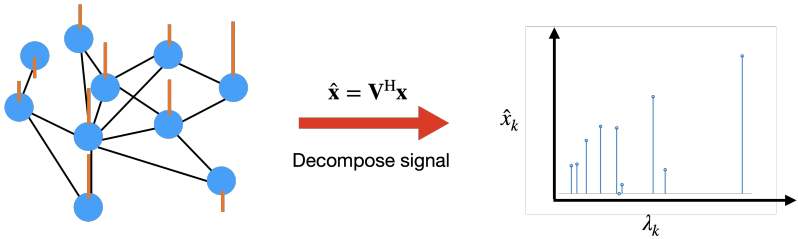
- Work with (any) Laplacian matrix: $\mathbf{S} = \mathbf{L}$ $\Rightarrow$ Symmetric (Positive Semidefinite)
  - $\Rightarrow$ Always: $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$
  - $\Rightarrow$ Real positive eigenvalues $\Rightarrow 0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$
- Eigenvector variability: $\mathbf{v}_i^{\mathrm{H}}\mathbf{L}\mathbf{v}_i = \lambda_i \quad 0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N \qquad \mathbf{L}\mathbf{v}_i = \lambda_i$



- Graph Fourier Transform: $\hat{\mathbf{x}} = \mathbf{V}^{\mathrm{H}}\mathbf{x}$



- Graph Fourier Transform: $\hat{\mathbf{x}} = \mathbf{V}^{\mathrm{H}}\mathbf{x}$



- Coefficient $\hat{x}_k$ measures the contribution of eigenvector $\mathbf{v}_k$ into representing the variability of $\mathbf{x}$

Convolutional Filter Frequency Response

- Transformed the filter input-output relation in the spectral domain $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$

$$\mathbf{z} = \mathbf{H}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \, \mathbf{S}^k \mathbf{x} \quad \rightarrow \quad \hat{\mathbf{z}} = \mathbf{H}(\mathbf{\Lambda})\hat{\mathbf{x}} = \sum_{k=0}^{K} h_k \, \mathbf{\Lambda}^k \hat{\mathbf{x}}$$

Convolutional Filter Frequency Response

- Transformed the filter input-output relation in the spectral domain $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$

$$\mathbf{z} = \mathbf{H}(\mathbf{S})\mathbf{x} = \sum_{k=0}^{K} h_k \, \mathbf{S}^k \mathbf{x} \quad \rightarrow \quad \hat{\mathbf{z}} = \mathbf{H}(\mathbf{\Lambda})\hat{\mathbf{x}} = \sum_{k=0}^{K} h_k \, \mathbf{\Lambda}^k \hat{\mathbf{x}}$$

- Filter frequency response or transfer function

$$\mathbf{H}(\mathbf{\Lambda}) = \sum_{k=0}^{K} h_k \, \mathbf{\Lambda}^k = \operatorname{diag}\left(\sum_{k=0}^{K} h_k \lambda_1^k, \ldots, \sum_{k=0}^{K} h_k \lambda_N^k \right)$$

  $\Rightarrow$ Diagonal matrix - polynomial in $\mathbf{\Lambda}$

Effect of Running the Same Filter on Different Graphs

- Given coefficients $\mathbf{h} = \{h_k\}$ the graph filter frequency response is determined $\mathsf{h}(\lambda) = \sum_{k=0}^{K} h_k \lambda^k$

  $\Rightarrow$ Universal for any graph

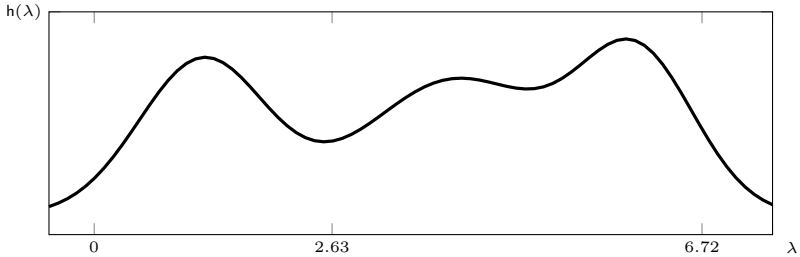


Fig. credit: F. Gama

Effect of Running the Same Filter on Different Graphs

- Given coefficients $\mathbf{h} = \{h_k\}$ the graph filter frequency response is determined $\mathbf{h}(\lambda) = \sum_{k=0}^{K} h_k \lambda^k$

  $\Rightarrow$ Universal for any graph
- For a specific graph, the response is instantiated on its specific eigenvalues λ_i

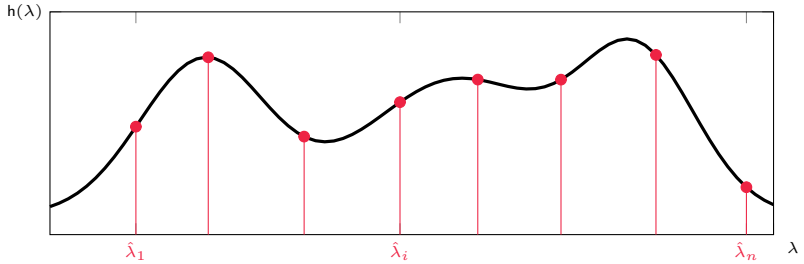


Fig. credit: F. Gama

Effect of Running the Same Filter on Different Graphs

- Given coefficients $\mathbf{h} = \{h_k\}$ the graph filter frequency response is determined $\mathbf{h}(\lambda) = \sum_{k=0}^{K} h_k \lambda^k$

  $\Rightarrow$ Universal for any graph
- For a specific graph, the response is instantiated on its specific eigenvalues λ_i
- For a different graph, the response is instantiated different eigenvalues λ_i

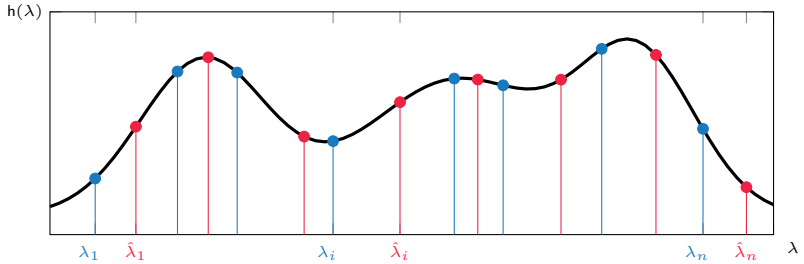


Fig. credit: F. Gama

Perturbation in the Spectrum

- If the graph is perturbed during testing

  $\Rightarrow$ We evaluate the filter on different eigenvalues

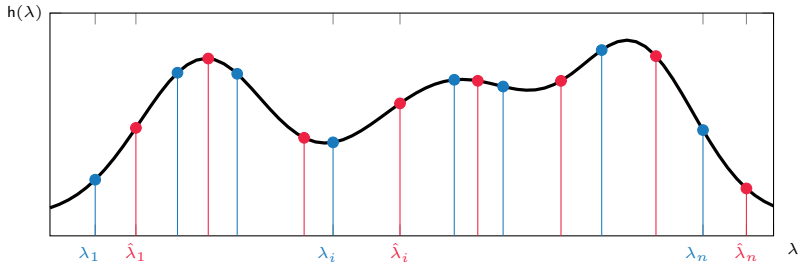


Fig. credit: F. Gama

Perturbation in the Spectrum

- If the graph is perturbed during testing

  $\Rightarrow$ We evaluate the filter on different eigenvalues

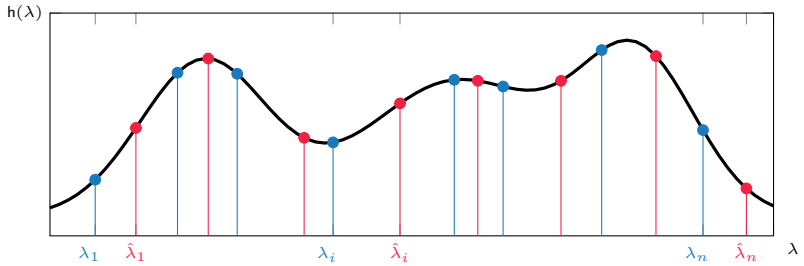


Fig. credit: F. Gama

- **[Relative Small Perturbation]** $\mathcal{E}(\mathbf{S}) = \{\mathbf{E} \in \mathbb{R}^{n \times n} : \widetilde{\mathbf{S}} = \mathbf{S} + \mathbf{E}\mathbf{S} + \mathbf{S}\mathbf{E}, \mathbf{E} = \mathbf{E}^{\top}\}$

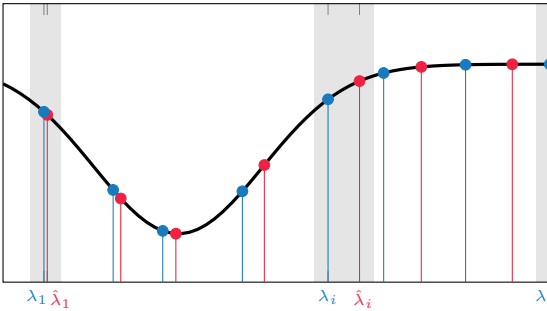
Levie, I., & Kutyniok, On the transferability of spectral graph filters, SampTA, 2019

Gama, Bruna, & Ribeiro, Stability properties of graph neural networks, IEEE T-SP, 2020.

Stability to Relative Perturbation

- If the filter is integral Lipschitz

$$|\lambda h'(\lambda)| \leq C_L$$



Gama, Bruna, & Ribeiro, Stability properties of graph neural networks, IEEE T-SP, 2020.
Levie, Huang, Bucci, Bronstein, & Kutyniok, Transferability of spectral graph convolutional neural networks, JMLR, 2021.

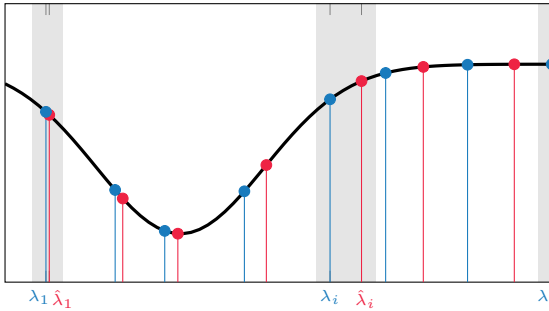
Stability to Relative Perturbation

- If the filter is integral Lipschitz

$$|\lambda h'(\lambda)| \leq C_L$$

- The graph convolutional filter and GCNN are stable to relative perturbation

  $\Rightarrow$ Filter: $\|\mathbf{H}(\mathbf{S})\mathbf{x}-\mathbf{H}(\tilde{\mathbf{S}})\mathbf{x}\|_2 \leq 2\sqrt{n}C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$,



Gama, Bruna, & Ribeiro, Stability properties of graph neural networks, IEEE T-SP, 2020.

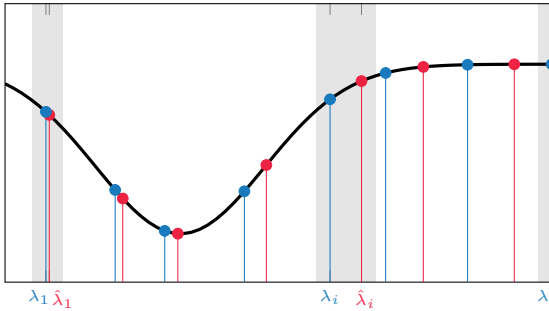
Levie, Huang, Bucci, Bronstein, & Kutyniok, Transferability of spectral graph convolutional neural networks, JMLR, 2021.

Stability to Relative Perturbation

- If the filter is integral Lipschitz

$$|\lambda h'(\lambda)| \leq C_L$$

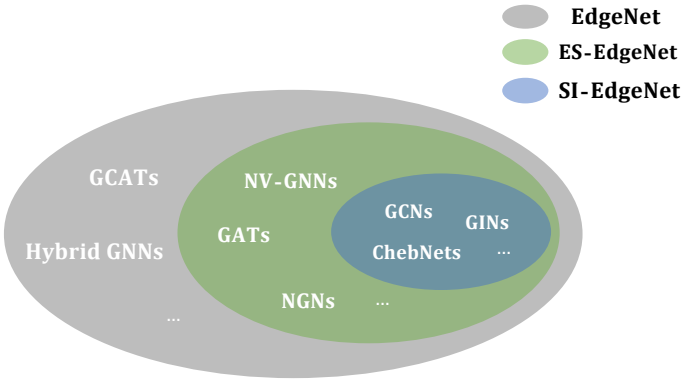
- The graph convolutional filter and GCNN are stable to relative perturbation
  - $\Rightarrow$ Filter: $\|\mathbf{H}(\mathbf{S})\mathbf{x}-\mathbf{H}(\tilde{\mathbf{S}})\mathbf{x}\|_2 \leq 2\sqrt{n}C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$,
  - $\Rightarrow$ GCNN: $\left\|\mathbf{\Phi}(\mathbf{x},\mathbf{S},\mathcal{H})-\mathbf{\Phi}(\mathbf{x},\tilde{\mathbf{S}},\mathcal{H})\right\|_2 \leq LF^{L-1}C_{\text{Filt}}\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$.
- Stability is affected by representational capacity
  - $\Rightarrow$ More layers L and filter banks F



Gama, Bruna, & Ribeiro, Stability properties of graph neural networks, IEEE T-SP, 2020.

Levie, Huang, Bucci, Bronstein, & Kutyniok, Transferability of spectral graph convolutional neural networks, JMLR, 2021.

Edge Varying Graph Neural Network



- Q: Does stability hold for other more general GNNs?

Stability of Shift-Invariant EdgeNet

- We analyze SI-EdgeGF in the frequency domain

$$\mathbf{y} = \mathbf{H}_{\mathrm{SI}}(\mathbf{S})\mathbf{x} := \sum_{k=0}^{K} \mathbf{\Phi}^{(k)} \mathbf{S}^k \mathbf{x}$$

- Using $\mathbf{S}^k = \mathbf{V}\mathbf{\Lambda}^k\mathbf{V}^{-1}$ and $\mathbf{\Phi}^{(k)} = \mathbf{V}\Phi^{(k)}\mathbf{V}^{-1}$ ($\mathbf{\Phi}^{(k)}$ shares the eigenvectors)

$$\hat{y}_i = \sum_{k=0}^{K} \phi_i^{(k)} \lambda_i^k \hat{x}_i, \text{ for } i = 1, \ldots, n$$

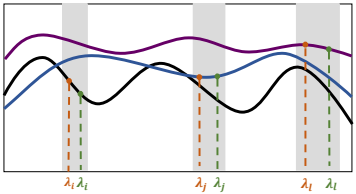
$\Rightarrow \phi_i^{(k)}$ and λ_i are the eigenvalues of $\mathbf{\Phi}^{(k)}$ and $\mathbf{S}$

Stability of Shift-Invariant EdgeNet

- **Definition.** Shift-invariant frequency response is a collection of n index-specific analytic functions

$$h_i(\lambda) := \sum_{k=0}^{K} \phi_i^{(k)} \lambda^k, \text{ for } i = 1, \dots, n$$

$\Rightarrow \phi_i^{(k)}$ are function parameters and λ is the function variable.

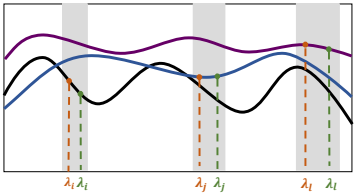


Stability of Shift-Invariant EdgeNet

- **Definition.** Shift-invariant frequency response is a collection of n index-specific analytic functions

$$h_i(\lambda) := \sum_{k=0}^{K} \phi_i^{(k)} \lambda^k, \text{ for } i = 1, \ldots, n$$

  $\Rightarrow$ $\phi_i^{(k)}$ are function parameters and λ is the function variable.



- There are n different spectral responses

  $\Rightarrow$ Each operating on a different graph frequency λ_i

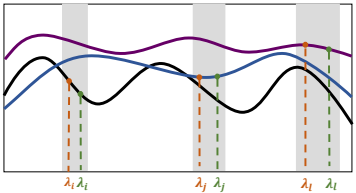
Stability of Shift-Invariant EdgeNet

- The shift-invariant edge varying filter is stable to relative perturbation

$$\|\mathbf{H}_{\mathrm{SI}}(\mathbf{x},\mathbf{S})-\mathbf{H}_{\mathrm{SI}}(\mathbf{x},\widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$$

$\Rightarrow$ only if all spectral responses are integral Lipschitz

$$|\lambda h_i'(\lambda)| \leq C_L, \text{ for all } i=1,\ldots,n.$$



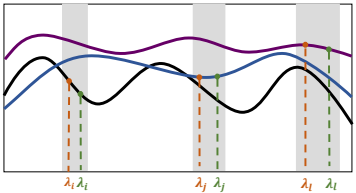
Stability of Shift-Invariant EdgeNet

- The shift-invariant edge varying filter is stable to relative perturbation

$$\|\mathbf{H}_{\mathrm{SI}}(\mathbf{x},\mathbf{S})-\mathbf{H}_{\mathrm{SI}}(\mathbf{x},\widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$$

  $\Rightarrow$ only if all spectral responses are integral Lipschitz

$$|\lambda h_i'(\lambda)| \leq C_L, \text{ for all } i=1,\ldots,n.$$

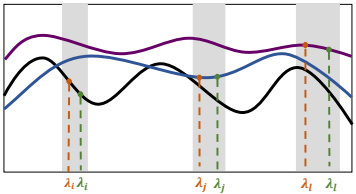


- The Lipschitz property of $h_i(\lambda)$ is determined by the eigenvalues of $\mathbf{\Phi}^{(k)}$

  $\Rightarrow$ Stability can be improved by designing $\mathbf{\Phi}^{(k)}$ to reduce variability of $h_i(\lambda)$ in nearby frequencies

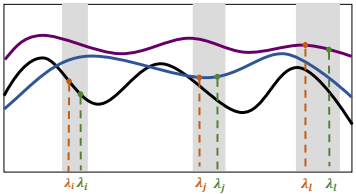
Stability of Shift-Invariant EdgeNet

- All spectral responses are integral Lipschitz $|\lambda h_i'(\lambda)| \leq C_L$, for all $i = 1, ..., n$.



Stability of Shift-Invariant EdgeNet

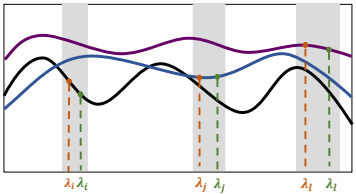
- All spectral responses are integral Lipschitz $|\lambda h_i'(\lambda)| \leq C_L$, for all $i = 1, ..., n$.



- This stability result holds uniformly for any graph $\mathbf{S}$. If $\mathbf{S}$ is fixed:
  - $\Rightarrow$ Each $h_i(\lambda)$ only instantiates on a single frequency λ_i of $\mathbf{S}$
  - $\Rightarrow$ The Lipschitz condition only needs to be satisfied w.r.t. a single value λ_i

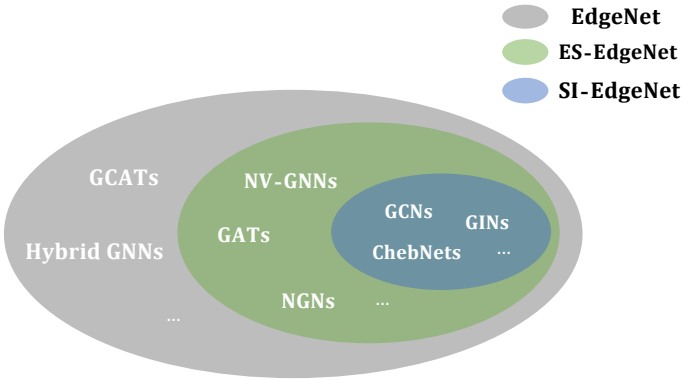
Stability of Shift-Invariant EdgeNet

- All spectral responses are integral Lipschitz $|\lambda h_i'(\lambda)| \leq C_L$, for all $i = 1, ..., n$.



- This stability result holds uniformly for any graph $\mathbf{S}$. If $\mathbf{S}$ is fixed:
  - $\Rightarrow$ Each $h_i(\lambda)$ only instantiates on a single frequency λ_i of $\mathbf{S}$
  - $\Rightarrow$ The Lipschitz condition only needs to be satisfied w.r.t. a single value λ_i
- For $\mathbf{\Phi}^{(k)} = h_k \mathbf{I}$ – stability bound reduces to the graph convolutional filter

Edge Varying Graph Neural Network



- Q: Does stability hold for other more general GNNs?

Stability of Eigenvector-Sharing EdgeNet

- ES-EdgeNet require edge weight matrices $\mathbf{\Phi}^{(k)}$ to share the eigenvectors $\mathbf{U}$
  $\Rightarrow$ $\mathbf{U}$ can be different from the eigenvectors of $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$

Stability of Eigenvector-Sharing EdgeNet

- ES-EdgeNet require edge weight matrices $\mathbf{\Phi}^{(k)}$ to share the eigenvectors $\mathbf{U}$
  $\Rightarrow$ $\mathbf{U}$ can be different from the eigenvectors of $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$

- **Definition [ε-misalignment of eigenvectors].** Let $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ and $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_n]$ be two sets of eigenvectors. Let each $\mathbf{v}_i$ is close to $\mathbf{u}_i$ up to misalignment ε.

Stability of Eigenvector-Sharing EdgeNet

- ES-EdgeNet require edge weight matrices $\mathbf{\Phi}^{(k)}$ to share the eigenvectors $\mathbf{U}$
  $\Rightarrow$ $\mathbf{U}$ can be different from the eigenvectors of $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{H}}$

- **Definition [ε-misalignment of eigenvectors].** Let $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ and $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_n]$ be two sets of eigenvectors. Let each $\mathbf{v}_i$ is close to $\mathbf{u}_i$ up to misalignment ε.
- For integral Lipschitz frequency responses, the eigenvector sharing edge varying filter is stable

$$\|\mathbf{H}_{\mathrm{ES}}(\mathbf{x}, \mathbf{S}) - \mathbf{H}_{\mathrm{ES}}(\mathbf{x}, \widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1 + n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon + \mathcal{O}(\epsilon^2)$$

Stability of Eigenvector-Sharing EdgeNet

- ES-EdgeNet require edge weight matrices $\boldsymbol{\Phi}^{(k)}$ to share the eigenvectors $\mathbf{U}$
  $\Rightarrow$ $\mathbf{U}$ can be different from the eigenvectors of $\mathbf{S} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^{\mathrm{H}}$

- **Definition [ε-misalignment of eigenvectors].** Let $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ and $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_n]$ be two sets of eigenvectors. Let each $\mathbf{v}_i$ is close to $\mathbf{u}_i$ up to misalignment ε.

- For integral Lipschitz frequency responses, the eigenvector sharing edge varying filter is stable

$$\|\mathbf{H}_{\mathrm{ES}}(\mathbf{x}, \mathbf{S}) - \mathbf{H}_{\mathrm{ES}}(\mathbf{x}, \widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1 + n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon + \mathcal{O}(\epsilon^2)$$

- The eigenvector misalignment ε influences stability
  $\Rightarrow$ Consequence of graph perturbations propagating through different eigenbases of $\mathbf{S}$ and $\boldsymbol{\Phi}^{(k)}$

Stability of Eigenvector-Sharing EdgeNet

- The eigenvector sharing edge varying filter is stable to relative perturbation

$$\|\mathbf{H}_{\mathrm{ES}}(\mathbf{x},\mathbf{S})-\mathbf{H}_{\mathrm{ES}}(\mathbf{x},\widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1+n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$$

  $\Rightarrow$ The smaller $\varepsilon \to$ the more stable the ES-EdgeGF (ε is smaller than one by definition)

Stability of Eigenvector-Sharing EdgeNet

- The eigenvector sharing edge varying filter is stable to relative perturbation

$$\|\mathbf{H}_{\mathrm{ES}}(\mathbf{x},\mathbf{S})-\mathbf{H}_{\mathrm{ES}}(\mathbf{x},\widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1+n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$$

  $\Rightarrow$ The smaller $\varepsilon \rightarrow$ the more stable the ES-EdgeGF (ε is smaller than one by definition)

- Trade-off:

  $\Rightarrow$ The increased flexibility of edge weight matrices $\mathbf{\Phi}^{(k)}$ – improved representational capacity

  $\Rightarrow$ The increased loss of filter output – reduced perturbation stability

Stability of Eigenvector-Sharing EdgeNet

- The eigenvector sharing edge varying filter is stable to relative perturbation

$$\|\mathbf{H}_{\mathrm{ES}}(\mathbf{x},\mathbf{S})-\mathbf{H}_{\mathrm{ES}}(\mathbf{x},\widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1+n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}(\epsilon^2)$$

  $\Rightarrow$ The smaller $\varepsilon \to$ the more stable the ES-EdgeGF (ε is smaller than one by definition)

- Trade-off:
  $\Rightarrow$ The increased flexibility of edge weight matrices $\mathbf{\Phi}^{(k)}$ – improved representational capacity
  $\Rightarrow$ The increased loss of filter output – reduced perturbation stability

- If $\mathbf{U}$ are aligned with $\mathbf{V}$, ES-EdgeGFs reduce to SI-EdgeGFs

Stability of General Edge Varying Filters

- General EdgeGF imposes no restriction on $\mathbf{\Phi}^{(k)}$

  $\Rightarrow$ Yield the strongest representational capacity

- General EdgeGF are stable

$$\|\mathbf{H}_{\text{Edge}}(\mathbf{x}, \mathbf{S}) - \mathbf{H}_{\text{Edge}}(\mathbf{x}, \widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1 + n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon + \mathcal{O}(\epsilon^2) + \mathcal{O}(\varepsilon^2)$$

  $\Rightarrow$ stronger integral Lipschitz conditions

Stability of General Edge Varying Filters

- General EdgeGF imposes no restriction on $\mathbf{\Phi}^{(k)}$

  $\Rightarrow$ Yield the strongest representational capacity

- General EdgeGF are stable

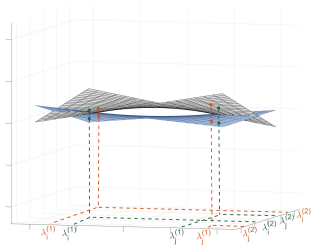
$$\|\mathbf{H}_{\text{Edge}}(\mathbf{x},\mathbf{S})-\mathbf{H}_{\text{Edge}}(\mathbf{x},\widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1+n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}\big(\epsilon^2\big)+\mathcal{O}\big(\varepsilon^2\big)$$

  $\Rightarrow$ stronger integral Lipschitz conditions

- Behave as a filter with K-dimensional frequency responses

$$h_i(\boldsymbol{\lambda}) = \sum_{k=0}^{K} \phi_i^{(k)} \prod_{\kappa=1}^{k} \lambda^{(\kappa)} \text{ for } i = 1,\dots,n,$$

  $\Rightarrow$ K arbitrary spectral misalignments between $\mathbf{\Phi}^{(k)}$ and $\mathbf{S}$



Stability of General Edge Varying Filters

- General EdgeGF imposes no restriction on $\mathbf{\Phi}^{(k)}$
  - $\Rightarrow$ Yield the strongest representational capacity
- General EdgeGF are stable

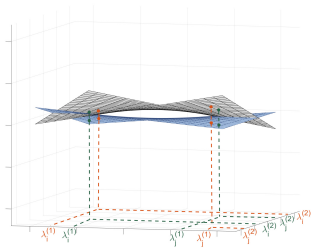
$$\|\mathbf{H}_{\text{Edge}}(\mathbf{x},\mathbf{S})-\mathbf{H}_{\text{Edge}}(\mathbf{x},\widetilde{\mathbf{S}})\|_2 \leq 2\sqrt{n}(1+n\varepsilon)C_L\|\mathbf{x}\|_2\epsilon+\mathcal{O}\big(\epsilon^2\big)+\mathcal{O}\big(\varepsilon^2\big)$$

  - $\Rightarrow$ stronger integral Lipschitz conditions

- Behave as a filter with K-dimensional frequency responses

$$h_i(\boldsymbol{\lambda}) = \sum_{k=0}^{K} \phi_i^{(k)} \prod_{\kappa=1}^{k} \lambda^{(\kappa)} \text{ for } i = 1,\dots,n,$$

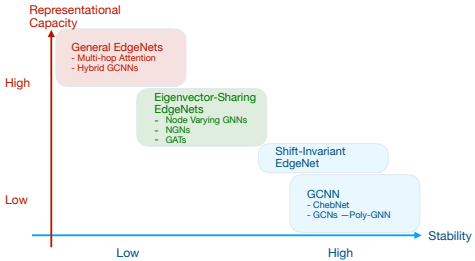
  - $\Rightarrow$ K arbitrary spectral misalignments between $\mathbf{\Phi}^{(k)}$ and $\mathbf{S}$
  - $\Rightarrow$ multidimensional integral Lipschitz filters



Trade-Off between Stability and Representational Capacity

- There exists an explicit trade-off between stability and representational capacity

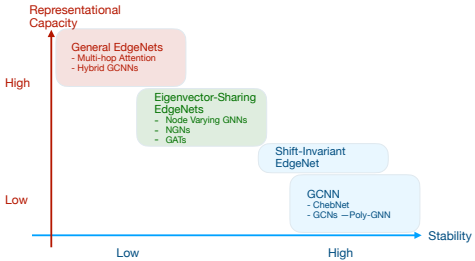
- Stability
  - ⇒ Low = stronger integral Lipschitz conditions
  - ⇒ Looser bounds
  - ⇒ Eigenvector misalignment contributes to it
- Representational capacity
  - ⇒ Low = more restriction on the parameter space
  - ⇒ General EdgeNets have more DoFs



Trade-Off between Stability and Representational Capacity

- There exists an explicit trade-off between stability and representational capacity

- Stability
  - $\Rightarrow$ Low = stronger integral Lipschitz conditions
  - $\Rightarrow$ Looser bounds
  - $\Rightarrow$ Eigenvector misalignment contributes to it
- Representational capacity
  - $\Rightarrow$ Low = more restriction on the parameter space
  - $\Rightarrow$ General EdgeNets have more DoFs

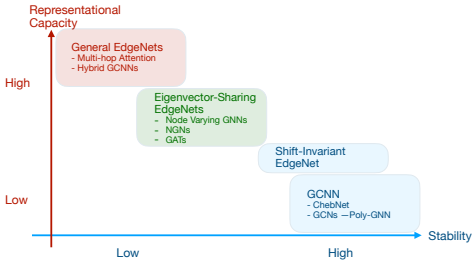


- Representational capacity increases with the width filters F and depth layers L
  - $\Rightarrow$ Stability degrades with the number of filters F and layers L

Trade-Off between Stability and Representational Capacity

- There exists an explicit trade-off between stability and representational capacity

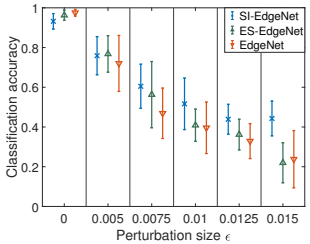
- Stability
  - $\Rightarrow$ Low = stronger integral Lipschitz conditions
  - $\Rightarrow$ Looser bounds
  - $\Rightarrow$ Eigenvector misalignment contributes to it
- Representational capacity
  - $\Rightarrow$ Low = more restriction on the parameter space
  - $\Rightarrow$ General EdgeNets have more DoFs

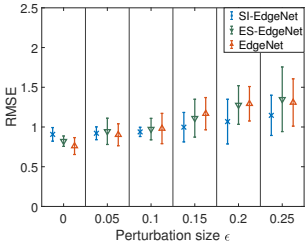


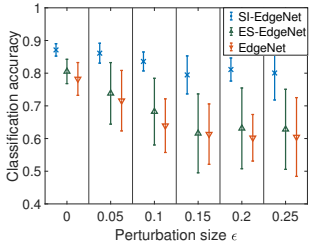
- Representational capacity increases with the width filters F and depth layers L
  - $\Rightarrow$ Stability degrades with the number of filters F and layers L
- Simpler GNNs may achieve a better trade-off in certain applications

Trade-Off between Stability and Representational Capacity

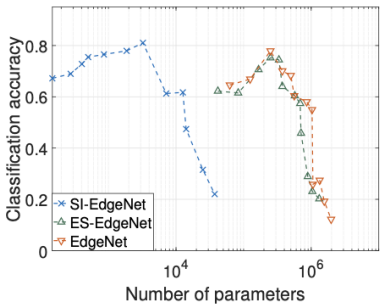
- Source localization: find the source community of a diffused signal over a stochastic block model graph
- Movie recommendation: predict movie ratings in the MovieLens100K dataset
- Authorship attribution: attribute texts to an author

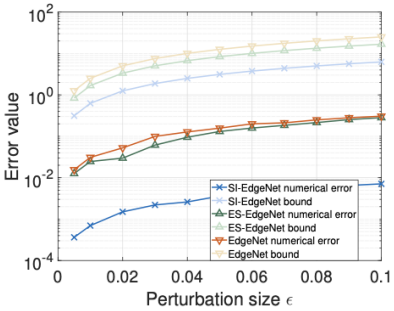






Trade-Off between Stability and Representational Capacity





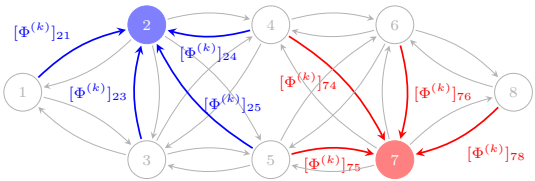
- We can increase representational capacity up to some point
  $\Rightarrow$ Over-parameterization pays on perturbed scenarios
- Universal bounds:Theory and experiments align in the trend
  $\Rightarrow$ Tighter bound can be achieved for specific graphs

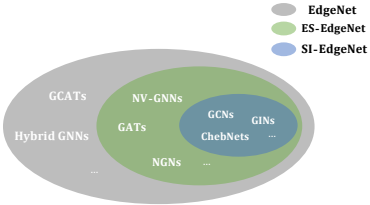
Conclusion & Future Work

Conclusion & Future Work

- EdgeNets represent a new paradigm to think of GNNs

$$\mathbf{H}_{\text{Edge}}(\mathbf{x}, \mathbf{S}) := \sum_{k=0}^{K} \mathbf{\Phi}^{(k)} \mathbf{S}^k \mathbf{x}$$





- General framework that includes many popular solutions and beyond

  $\Rightarrow$ Design new architectures in a principled manner

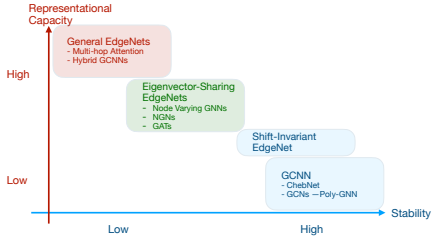
Conclusion & Future Work

- EdgeNets are stable to domain perturbations
  - $\Rightarrow$ Eigenvector misalignment between parameters $\mathbf{\Phi}^{(k)}$ and shift operator $\mathbf{S}$ is key

Conclusion & Future Work

- EdgeNets are stable to domain perturbations
  - $\Rightarrow$ Eigenvector misalignment between parameters $\mathbf{\Phi}^{(k)}$ and shift operator $\mathbf{S}$ is key

- Trade-off between stability and representational capacity

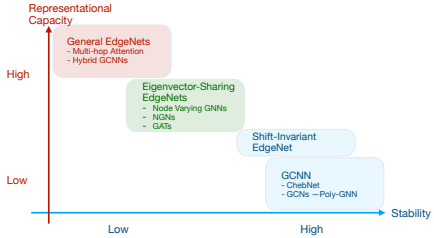


- FW1: More aggressive perturbation models
- FW2: Necessary conditions for stability
- FW3 Characterize representational capacity from approximation theory

Conclusion & Future Work

- EdgeNets are stable to domain perturbations
  - $\Rightarrow$ Eigenvector misalignment between parameters $\mathbf{\Phi}^{(k)}$ and shift operator $\mathbf{S}$ is key

- Trade-off between stability and representational capacity



- FW1: More aggressive perturbation models
- FW2: Necessary conditions for stability
- FW3 Characterize representational capacity from approximation theory

Plans for June 24th?