Graph Neural Networks
Architectures, Fundamental Properties and Applications

Navid NaderiAlizadeh, Alejandro Ribeiro, Luana Ruiz, Zhiyang Wang

Web: gnn.seas.upenn.edu/aaai-2025/

Feb 26 2025

Graph Neural Networks on Large-Scale Graphs

Graph Neural Networks tutorial – AAAI 2025

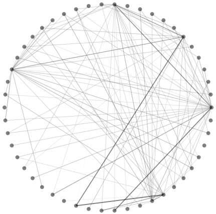
Luana Ruiz

Jonhs Hopkins University

lrubini1@jh.edu
https://luanaruiz9.github.io/

- Need to process information on very large graphs arises in a wide range of applications

  ⇒ E.g., product recommendation systems, control of teams of autonomous agents

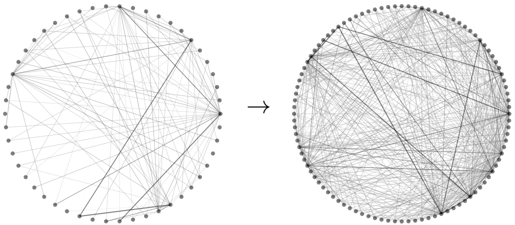


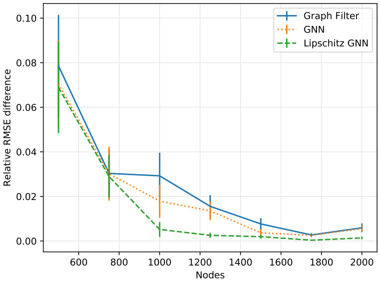
product similarity graph

robot swarm (Tolstaya, E. et al., 2019)

- GNNs are the solution of choice ⇒ have been shown to outperform other existing solutions

- Training model on a subsampled graph and transferring it for inference on the large graph





- Transferability of graph neural networks (GNNs) is useful in practice $\Rightarrow$ recommendation system

- Performance difference on training and target graphs decreases as size of training graph grows

Q1: We have empirically observed that GNNs scale. Why do they scale?

Q2: Can success of GNNs on moderate-size graphs be used to create success at large-scale?

- To answer these questions, turn to CNNs $\Rightarrow$ known to scale well for images and time sequences

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143×95 $\rightarrow$

205×136 $\rightarrow$

294×195 $\rightarrow$

600×399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image

- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143 × 95 →

205 × 136 →

294 × 195 →

600 × 399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image
- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32 × 32

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143×95 $\rightarrow$ 205×136 $\rightarrow$ 294×195 $\rightarrow$ 600×399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image
- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

$143 \times 95 \rightarrow 205 \times 136 \rightarrow 294 \times 195 \rightarrow 600 \times 399$

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image

- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143×95 $\rightarrow$

205×136 $\rightarrow$

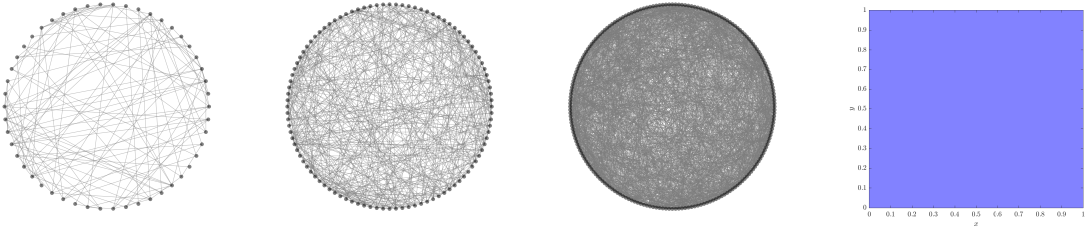
294×195 $\rightarrow$

600×399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image
- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon

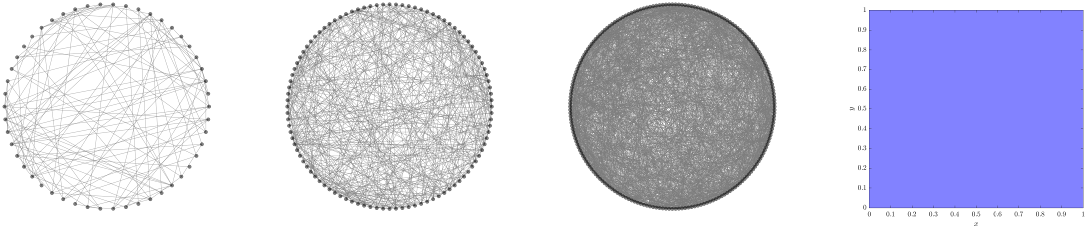


$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- A graphon can be thought of as a graph with an uncountable number of nodes

Graphons

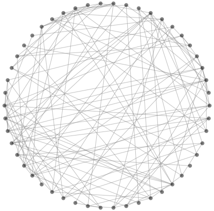
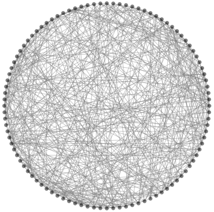
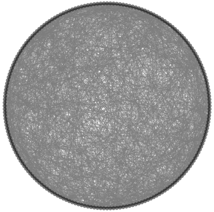
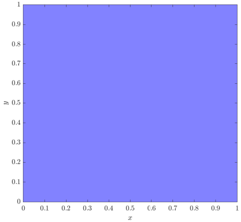
- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon



$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- A graphon can be thought of as a graph with an uncountable number of nodes

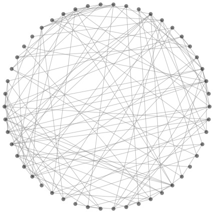
- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon

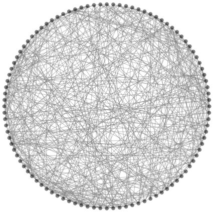





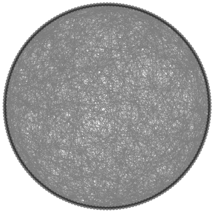
$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

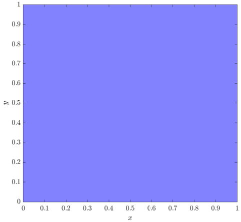
- A graphon can be thought of as a graph with an uncountable number of nodes

- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon







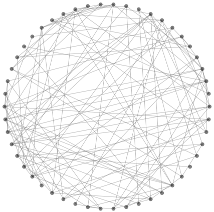


$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

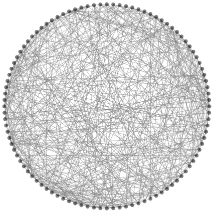
- A graphon can be thought of as a graph with an uncountable number of nodes

Graphons

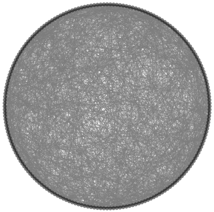
- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon



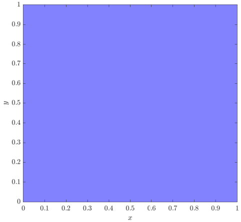
$n = 50$ nodes $\rightarrow$



$n = 100$ nodes $\rightarrow$



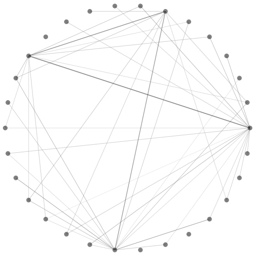
$n = 200$ nodes $\rightarrow$



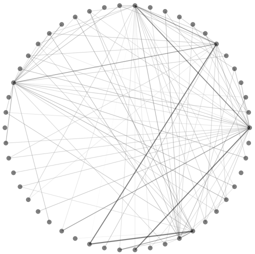
Graphon $W(u, v) = p$

- A graphon can be thought of as a graph with an uncountable number of nodes

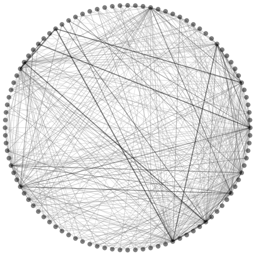
- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products



$n = 50$ products

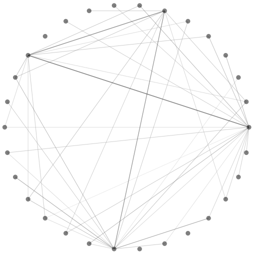


$n = 100$ products

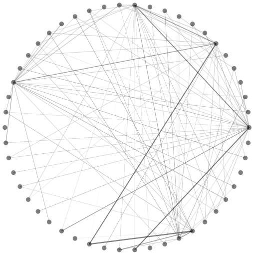
- So do graph convolutions and graph neural networks converge to limits on the graphon?

Large-Scale Graphs

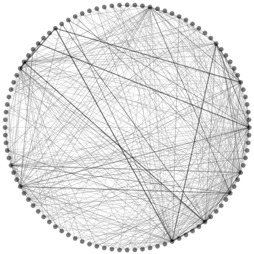
- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products



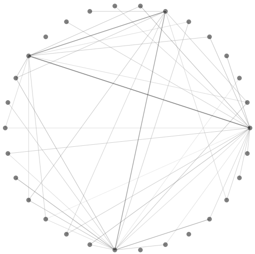
$n = 50$ products



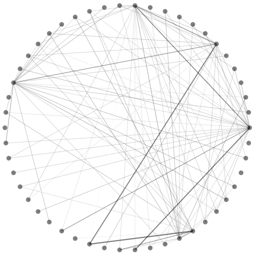
$n = 100$ products

- So do graph convolutions and graph neural networks converge to limits on the graphon?

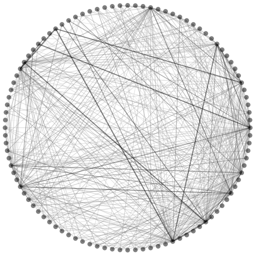
- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products



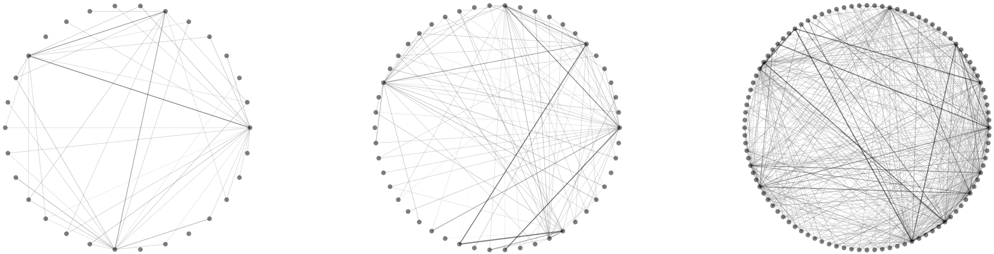
$n = 50$ products



$n = 100$ products

- So do graph convolutions and graph neural networks converge to limits on the graphon?

- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products | $n = 50$ products | $n = 100$ products

- So do graph convolutions and graph neural networks converge to limits on the graphon?

Q1: We have empirically observed that GNNs scale. Why do they scale?

- A1: Because graph convolutions and GNNs have well-defined limits on graphons

Ruiz, L., Chamon, L. F. O., Ribeiro, A., *Graphon Signal Processing*, IEEE TSP, 2021

Q2: Can success of GNNs on moderate-size graphs be used to create success at large-scale?

- A2: Yes, as GNNs are transferable $\Rightarrow$ can be trained on moderate-size and executed on large-scale

Ruiz, L., Chamon, L. F. O., Ribeiro, A., *Transferability Properties of Graph Neural Networks*, Submitted to IEEE TSP

Graphons

Definition (Graphon) (Borgs, C., Chayes, J., Lovász, L., Sós, V., Vesztergombi, K., 2008)

A graphon **W** is a bounded symmetric measurable function $\Rightarrow \mathbf{W} : [0,1]^2 \rightarrow [0,1]$

- Can think of a graphon as a weighted symmetric graph with an uncountable number of nodes

  $\Rightarrow$ Labels are graphon arguments $u \in [0,1]$, weights are graphon values $W(u,v) = W(v,u)$

- Interpreted as the limit of a sequence of graphs in the sense that densities of motifs converge

- Interpreted as a generative model of graph families by sampling edges $(u_i, u_j) \sim \mathcal{B}(\mathbf{W}(u_i, u_j))$

Definition (Graphon) (Borgs, C., Chayes, J., Lovász, L., Sós, V., Vesztergombi, K., 2008)

A graphon $\mathbf{W}$ is a bounded symmetric measurable function $\Rightarrow \mathbf{W} : [0,1]^2 \rightarrow [0,1]$

- Can think of a graphon as a weighted symmetric graph with an uncountable number of nodes

  $\Rightarrow$ Labels are graphon arguments $u \in [0,1]$, weights are graphon values $W(u,v) = W(v,u)$

- Interpreted as the limit of a sequence of graphs in the sense that densities of motifs converge

- Interpreted as a generative model of graph families by sampling edges $(u_i, u_j) \sim \mathcal{B}(\mathbf{W}(u_i, u_j))$

Definition (Graphon) (Borgs, C., Chayes, J., Lovász, L., Sós, V., Vesztergombi, K., 2008)

A graphon **W** is a bounded symmetric measurable function $\Rightarrow \mathbf{W} : [0,1]^2 \rightarrow [0,1]$

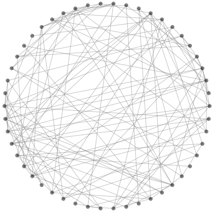
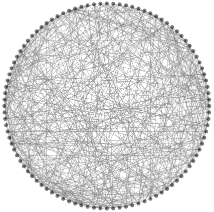
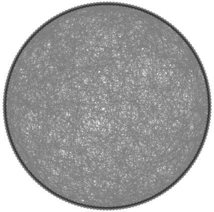
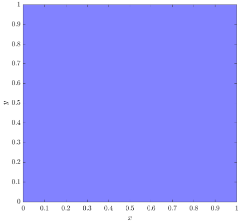
- Can think of a graphon as a weighted symmetric graph with an uncountable number of nodes

  $\Rightarrow$ Labels are graphon arguments $u \in [0,1]$, weights are graphon values $W(u,v) = W(v,u)$

- Interpreted as the limit of a sequence of graphs in the sense that densities of motifs converge

- Interpreted as a generative model of graph families by sampling edges $(u_i, u_j) \sim \mathcal{B}(\mathbf{W}(u_i, u_j))$

- A sequence of Erdős-Rényi graphs converges to Erdős-Rényi graphons

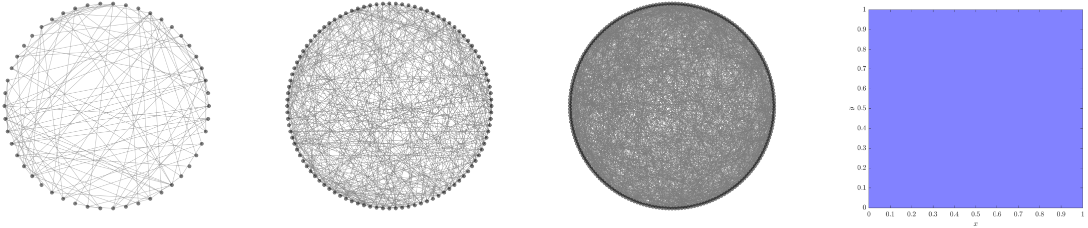





$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- The Erdős-Rényi graphon can be used to sample uniform graphs with 200, 100, and 50 nodes

Uniform Graphon as a Limit Object

- A sequence of Erdős-Rényi graphs converges to Erdős-Rényi graphons

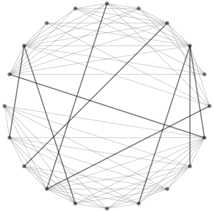


$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

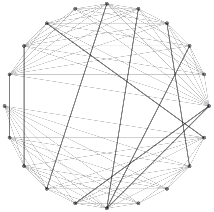
- The Erdős-Rényi graphon can be used to sample uniform graphs with 200, 100, and 50 nodes

SBM as a Limit Object

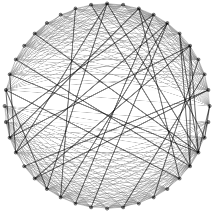
- A sequence of stochastic block model graphs converges to stochastic block model graphons



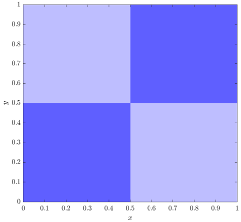
$n = 20$ nodes →



$n = 30$ nodes →



$n = 40$ nodes →

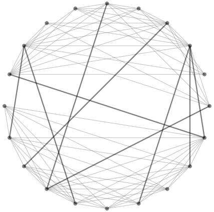


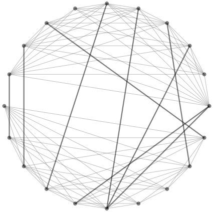
Graphon $W(u, v)$

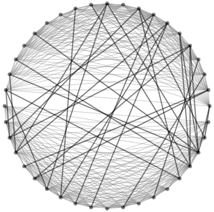
- The stochastic block model graphon can be used to sample SBM graphs with 40, 30, and 20 nodes

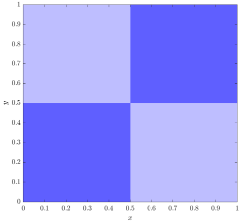
SBM as a Limit Object

- A sequence of stochastic block model graphs converges to stochastic block model graphons







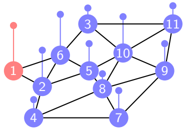


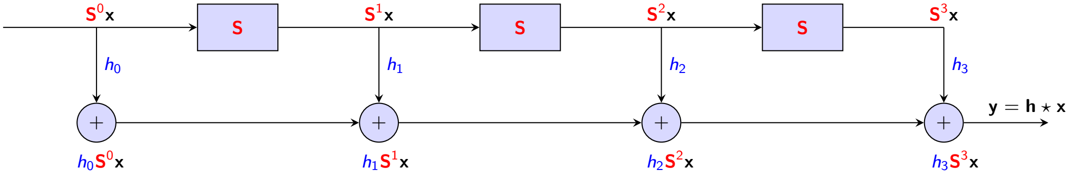
$n = 20$ nodes → $n = 30$ nodes → $n = 40$ nodes → Graphon $W(u, v)$

- The stochastic block model graphon can be used to sample SBM graphs with 40, 30, and 20 nodes

Graphon Convolutions

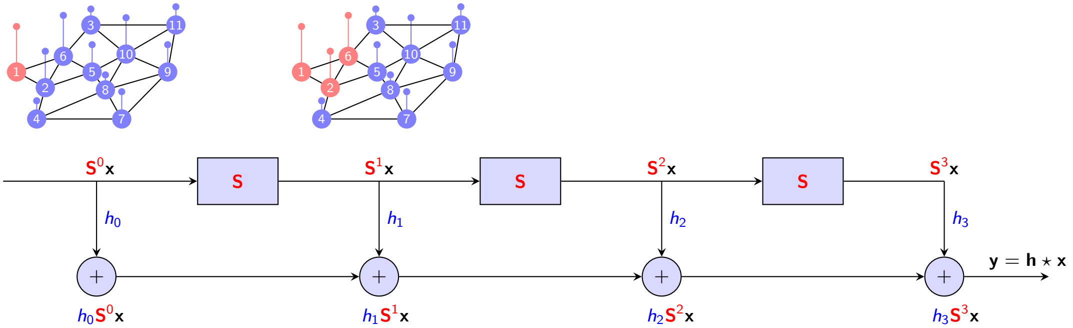
- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$





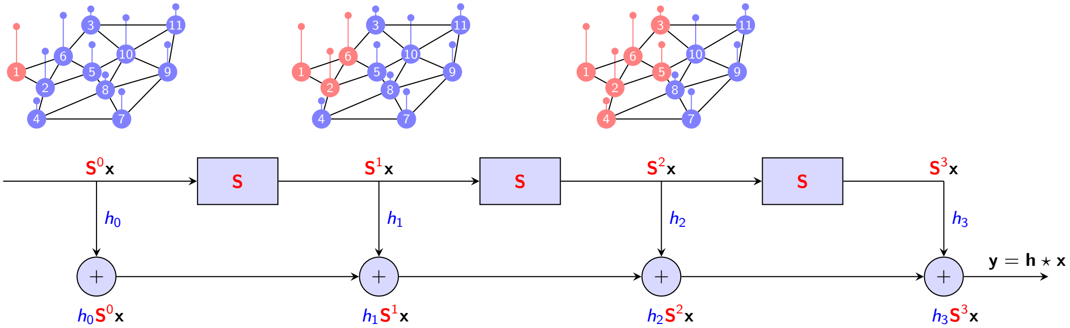
- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0 \, \mathbf{S}^0 \mathbf{x} \; + h_1 \, \mathbf{S}^1 \mathbf{x} \; + h_2 \, \mathbf{S}^2 \mathbf{x} \; + h_3 \, \mathbf{S}^3 \mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k \, \mathbf{S}^k \mathbf{x}$



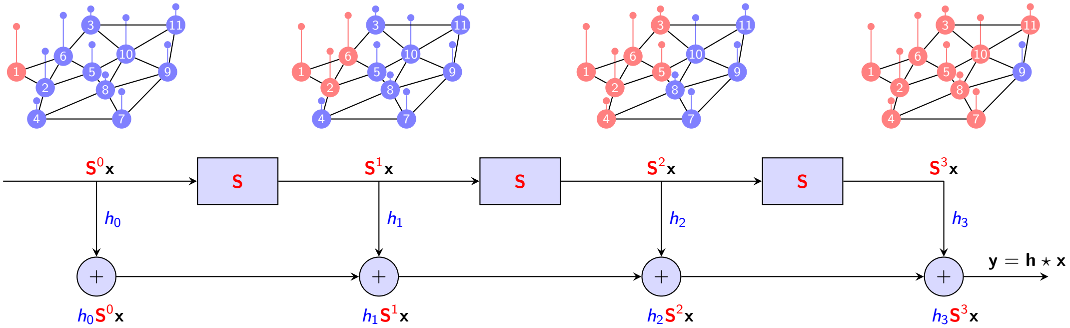
- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1} \Rightarrow$ graph shift operator

- Graph convolution $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$



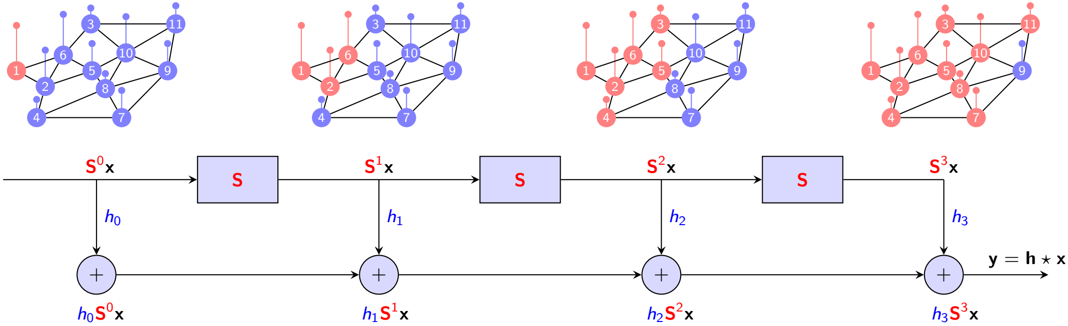
- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

- Graph convolution $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

- Graphon convolutions are analogously parametrized by the graphon shift operator

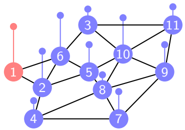
Definition (Graphon Shift Operator) (Ruiz, L., Chamon, L. F. O., Ribeiro A., TSP'21)

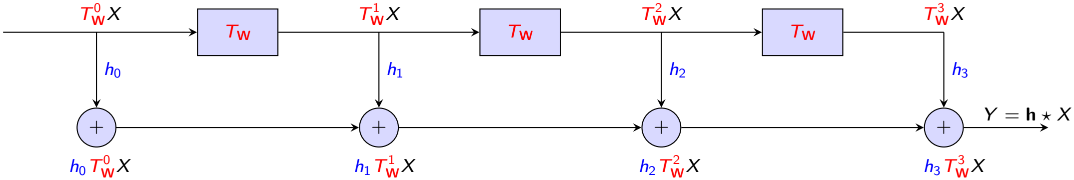
The graphon shift operator of a graphon $\mathbf{W}$ is defined as

$$Y(v) = (T_{\mathbf{W}}X)(v) = \int_0^1 \mathbf{W}(u,v)X(u)du.$$

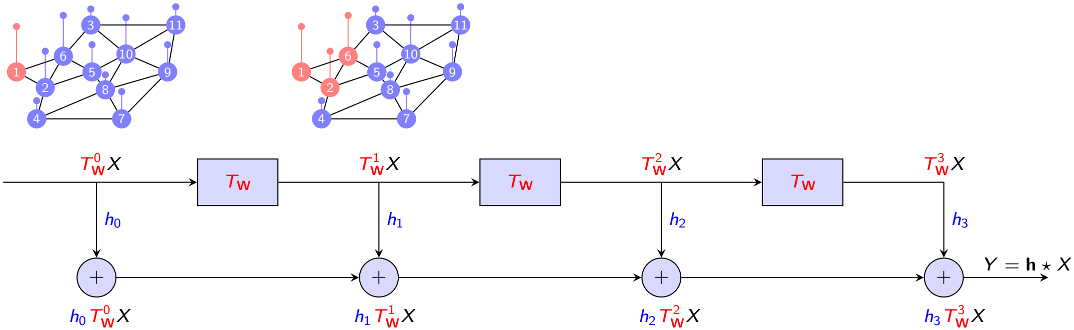
- The graphon shift operator is an integral linear operator with kernel given by the graphon $\mathbf{W}$

- Graphon convolution $\Rightarrow Z = \; h_0 \, T_{\mathbf{W}}^0 X \; + h_1 \, T_{\mathbf{W}}^1 X \; + h_2 \, T_{\mathbf{W}}^2 X \; + h_3 \, T_{\mathbf{W}}^3 X \; \ldots = \sum_{k=0}^{K-1} h_k \, T_{\mathbf{W}}^k X$

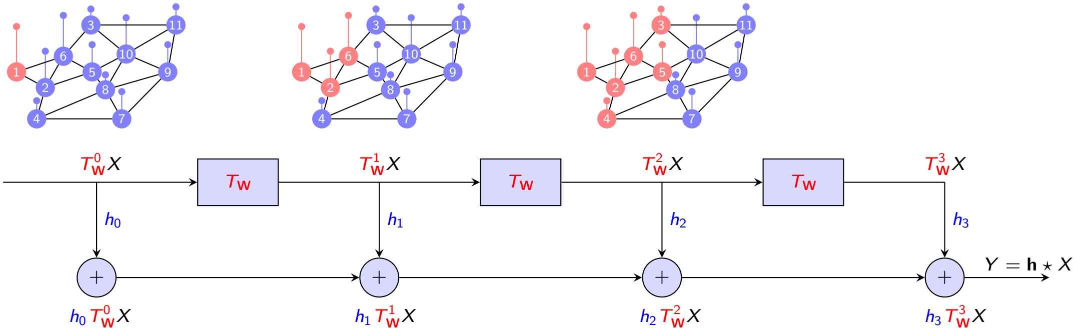




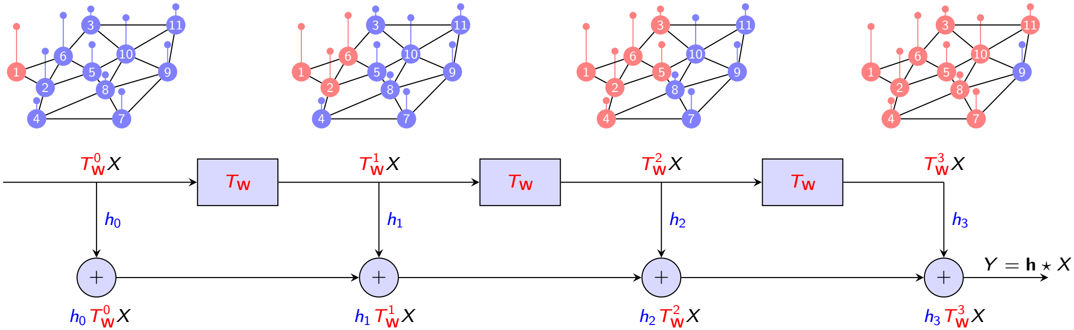
- Graphon convolution $\Rightarrow Z = h_0\, T_{\mathbf{W}}^0 X + h_1\, T_{\mathbf{W}}^1 X + h_2\, T_{\mathbf{W}}^2 X + h_3\, T_{\mathbf{W}}^3 X \ldots = \sum_{k=0}^{K-1} h_k\, T_{\mathbf{W}}^k X$



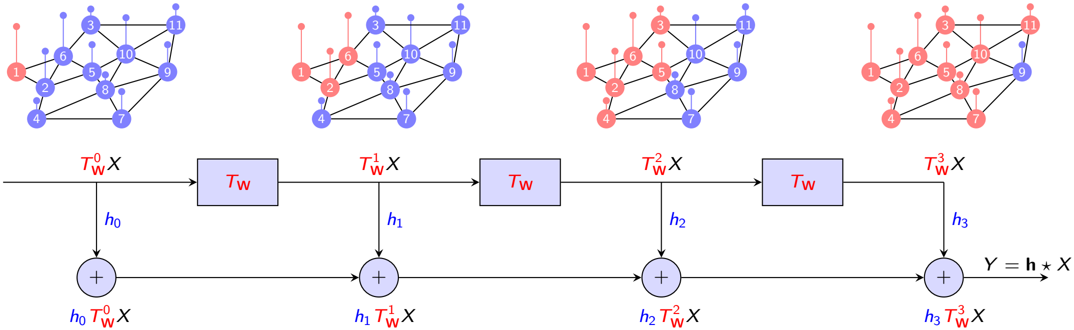
- Graphon convolution $\Rightarrow Z = h_0\, T_W^0 X + h_1\, T_W^1 X + h_2\, T_W^2 X + h_3\, T_W^3 X \ldots = \sum_{k=0}^{K-1} h_k\, T_W^k X$



- Graphon convolution $\Rightarrow Z = h_0\, T_W^0 X + h_1\, T_W^1 X + h_2\, T_W^2 X + h_3\, T_W^3 X \ldots = \sum_{k=0}^{K-1} h_k\, T_W^k X$



- Graphon convolution $\Rightarrow Z = h_0\, T_{\mathbf{W}}^0 X + h_1\, T_{\mathbf{W}}^1 X + h_2\, T_{\mathbf{W}}^2 X + h_3\, T_{\mathbf{W}}^3 X \ldots = \sum_{k=0}^{K-1} h_k\, T_{\mathbf{W}}^k X$



- The graph (which is symmetric) admits the eigenvector decomposition $\mathbf{S}_n = \mathbf{V}_n \boldsymbol{\Lambda}_n \mathbf{V}_n^H$

Theorem (Graph frequency representation of graph filters)

Consider graph filter with coefficients h_k, graph signal $\mathbf{x}_n$ and the filtered signal $\mathbf{y}_n = \sum_{k=0}^{K-1} h_k \mathbf{S}_n^k \mathbf{x}_n$.

The Graph Fourier Transforms $\tilde{\mathbf{x}}_n = \mathbf{V}_n^H \mathbf{x}_n$ and $\tilde{\mathbf{y}}_n = \mathbf{V}_n^H \mathbf{y}_n$ are related by

$$\tilde{y}_{nj} = \sum_{k=0}^{K-1} h_k \lambda_{nj}^k \tilde{x}_{nj} \qquad \Rightarrow \qquad \tilde{h}(\lambda) = \sum_{k=0}^{K-1} h_k \lambda^k$$

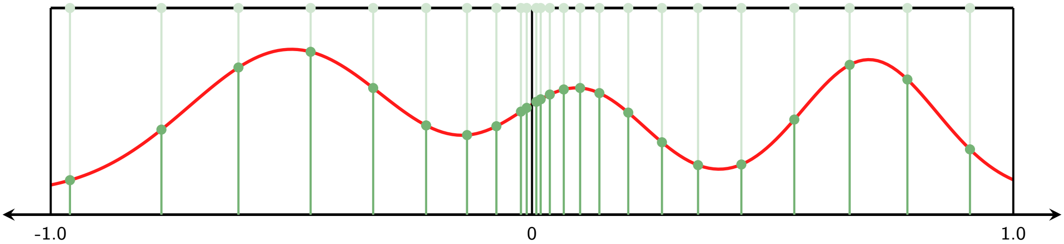
- This is a simple eigenvalue decomposition of the graph filter polynomial $\Rightarrow$ Nonetheless interesting

$\Rightarrow$ It is not only that the operator is pointwise, it also decouples the filter from the graph

Frequency Representation of Graph Convolutions

- The frequency response is independent of the graph. It is a polynomial on a scalar variable λ

- Graph determines eigenvalues at which response is instantiated $\Rightarrow \tilde{y}_{nj} = \sum_{k=0}^{K-1} h_k \lambda_{nj}^k \tilde{x}_{nj} = h(\lambda_{nj})\tilde{x}_{nj}$



- Since graphon shifts are Hilbert-Schmidt operators, the same can be done for graphon filters
- The eigenfunction representation of the graphon shift is $W(u, v) = \sum_{j\in\mathbb{Z}\setminus\{0\}} \lambda_j \phi_j(u)\varphi_j(v)$

Theorem (Graphon frequency representation of graphon filters)

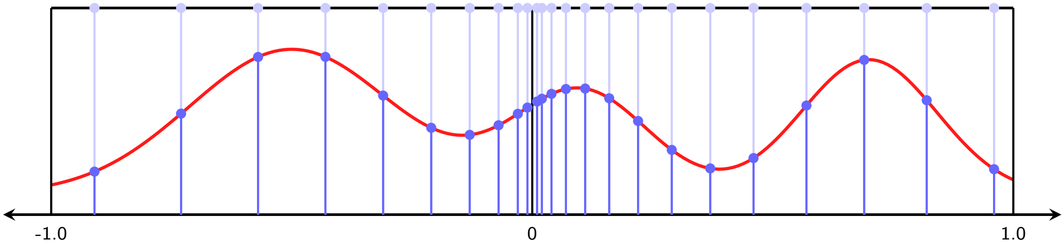
Consider graphon filter with coefficients h_k, graphon signal X and the filtered signal Y. The Graphon Fourier Transforms $\tilde{X}_j = \int_0^1 \varphi_j(u)X(u)du$ and $\tilde{Y}_j = \int_0^1 \varphi_j(u)Y(u)du$ are related by

$$\tilde{Y}_j = \sum_{k=0}^{K-1} h_k \lambda_j^k \tilde{X}_j \qquad \Rightarrow \qquad \tilde{h}(\lambda) = \sum_{k=0}^{K-1} h_k \lambda^k$$

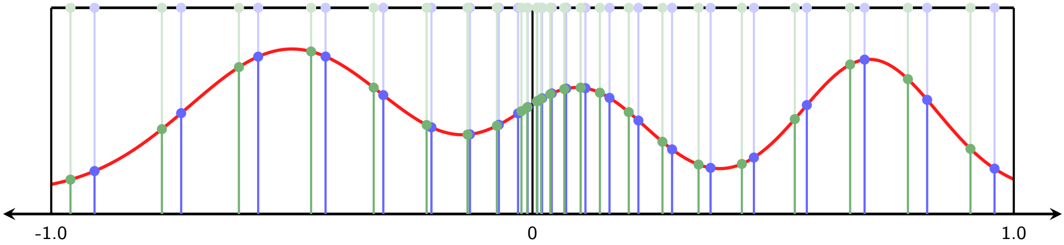
- Like graph filters, graphon filters have pointwise spectra and are decoupled from the graphon

- Graphon-independent. More importantly the same as the graph response for the same coefficients h_k

- Graphon determines eigenvalues at which response is instantiated $\Rightarrow \tilde{Y}_j = \sum_{k=0}^{K-1} h_k \lambda_j^k \tilde{X}_j = h(\lambda_j)\tilde{X}_j$



- Spectral response of graph and graphon convolution is given by the same function $h(\lambda)$



- Spectral response of the graph convolution determined by evaluating $h(\lambda)$ at graph eigenvalues

- Spectral response of the graphon convolution determined by evaluating $h(\lambda)$ at graphon eigenvalues

- Graph convolutions converge to graphon convolutions $\Rightarrow$ provided that $h(\lambda)$ is Lipschitz

Theorem (Convergence of Graph Convolutions) (Ruiz, L. et al., EUSIPCO'20, TSP'21)

Given convergent graph signal sequence $(G_n, \mathbf{x}_n) \to (W, X)$ and convolutions $\mathbf{H}(\mathbf{S}_n)$ and T_H

generated by the same coefficients h_k, if the spectral response $h(\lambda)$ is Lipschitz,

$$(\mathbf{G}_n, \mathbf{y}_n) \to (\mathbf{W}, Y)$$

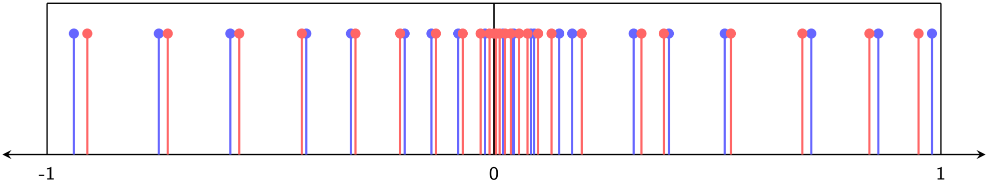
i.e., the sequence of output graph signals converges to the output graphon signal.

- Lipschitz continuity restriction better understood in the graph and graphon spectral domain

- Due to $T_{\mathbf{W}}$ being compact, graphon eigenvalues accumulate at $\lambda = 0 \Rightarrow \lim_{i\to\infty} \lambda_i = \lim_{i\to\infty} \lambda_{-i} = 0$

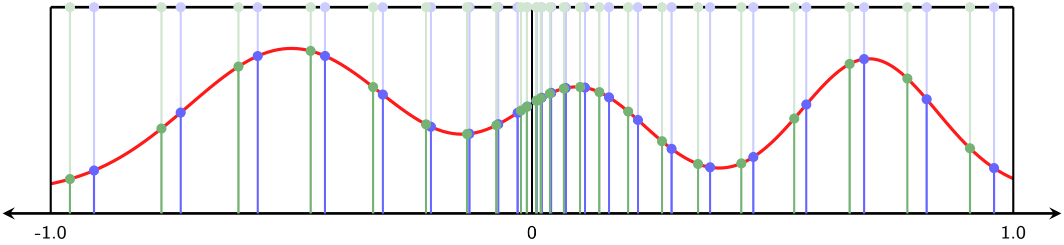
If a graph sequence $\{\mathbf{G}_n\}$ converges to a graphon $\mathbf{W}$, then

$$\lim_{n\to\infty} \frac{\lambda_j(\mathbf{S}_n)}{n} = \lambda_j(T_{\mathbf{W}}) \text{ for all } j$$ (Borgs, C. et al., 2012)



- But for $\neq j$, $\neq n_0$ are needed to show that $\exists\, n_0$ s.t. for all $n > n_0$, $\left| \frac{\lambda_j(\mathbf{S}_n)}{n} - \lambda_j(T_{\mathbf{W}}) \right| < \epsilon$

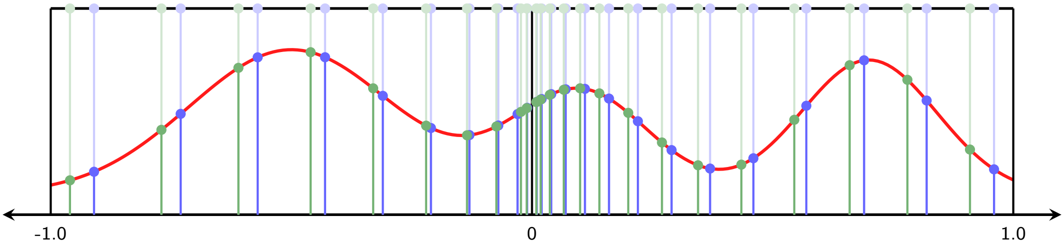
- Because eigenvalues converge, we can expect graph convolutions to converge



- But convergence near $\lambda = 0$ is complicated by eigenvalue convergence not being uniform

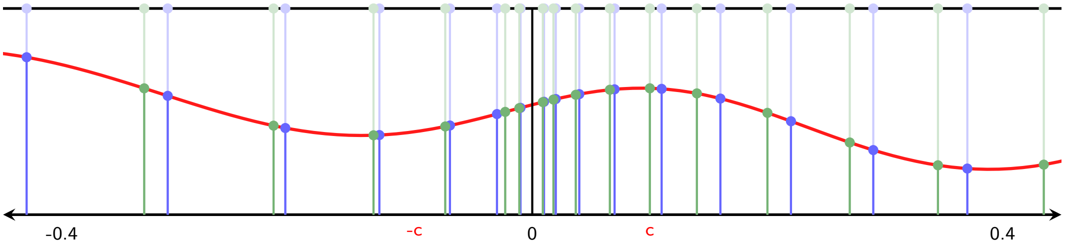
- Filters attempting to discriminate spectral components near $\lambda = 0$ do not converge

- This problem can be solved if we amplify these spectral components similarly for $|\lambda| \leq c$



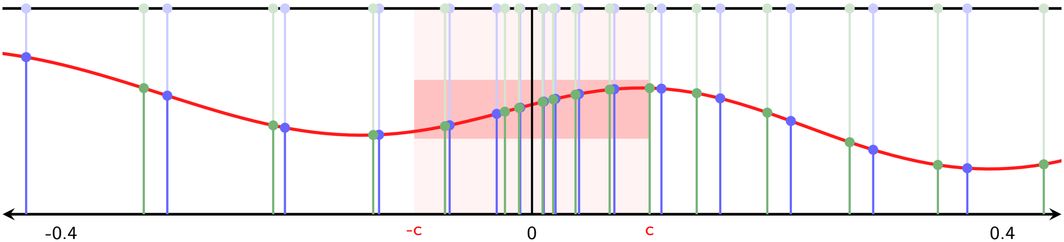
- Lipschitz filters ensure no mismatch between eigenspaces of $|\lambda_j(\mathbf{S}_n)| \leq c$ and $|\lambda_j(\mathbf{W})| \leq c$
- Lipschitz condition means that convergence comes at the cost of spectral discriminability

- This problem can be solved if we amplify these spectral components similarly for $|\lambda| \leq c$



- Lipschitz filters ensure no mismatch between eigenspaces of $|\lambda_j(\mathbf{S}_n)| \leq c$ and $|\lambda_j(\mathbf{W})| \leq c$
- Lipschitz condition means that convergence comes at the cost of spectral discriminability

- This problem can be solved if we amplify these spectral components similarly for $|\lambda| \leq c$

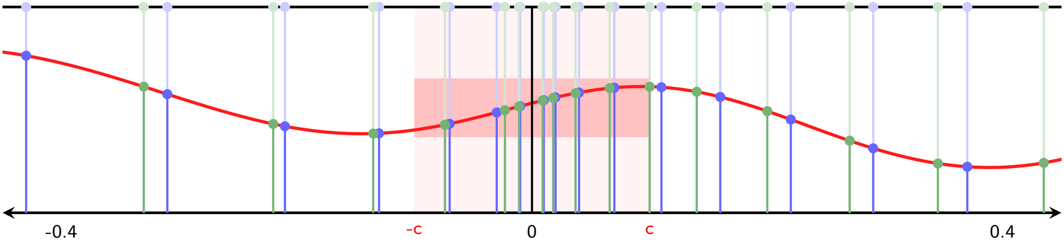


- Lipschitz filters ensure no mismatch between eigenspaces of $|\lambda_j(\mathbf{S}_n)| \leq c$ and $|\lambda_j(\mathbf{W})| \leq c$
- Lipschitz condition means that convergence comes at the cost of spectral discriminability

Transferability

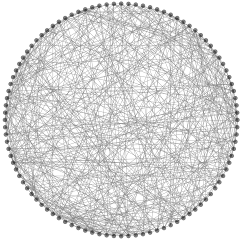
- Have established an asymptotic result $\Rightarrow$ graph convolutions converge, but with a condition

- Depending on the value of the Lipschitz constant of $h(\lambda)$, convergence may be faster or slower

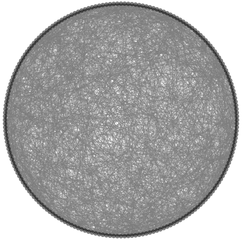


- In order to exploit this result in practice, need a non-asymptotic analysis for finite n

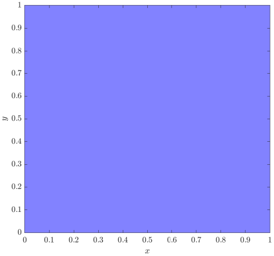
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



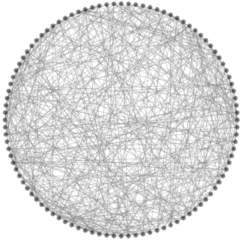
m nodes



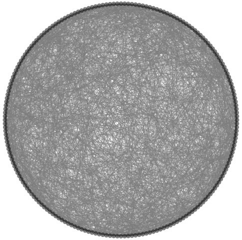
Graphon $W(u, v) = p$

- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

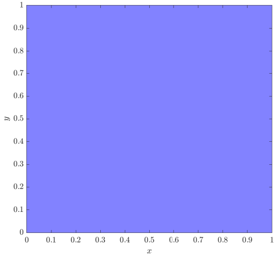
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



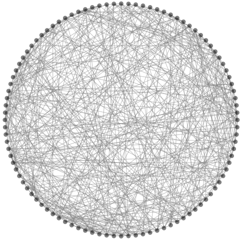
m nodes



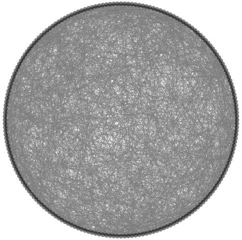
Graphon $W(u, v) = p$

- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

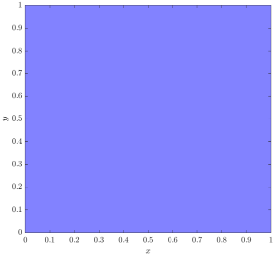
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



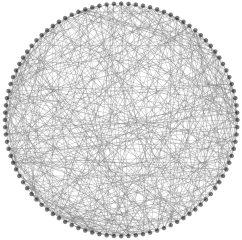
m nodes



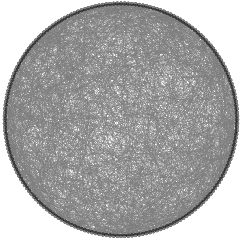
Graphon $W(u, v) = p$

- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

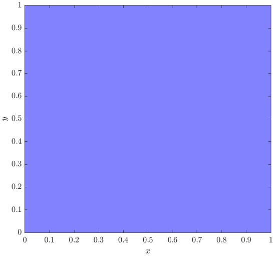
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



m nodes



Graphon $W(u, v) = p$

- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

Theorem (Transferability of Graph Convolutions)

Consider graph signals $(\mathbf{S}_n, \mathbf{x}_n)$ and $(\mathbf{S}_m, \mathbf{x}_m)$ sampled from graphon signal (W, X) along with filter outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $\mathbf{y}_m = \mathbf{H}(\mathbf{S}_m)\mathbf{x}_m$. The difference norm of the respective graphon induced signals is bounded by

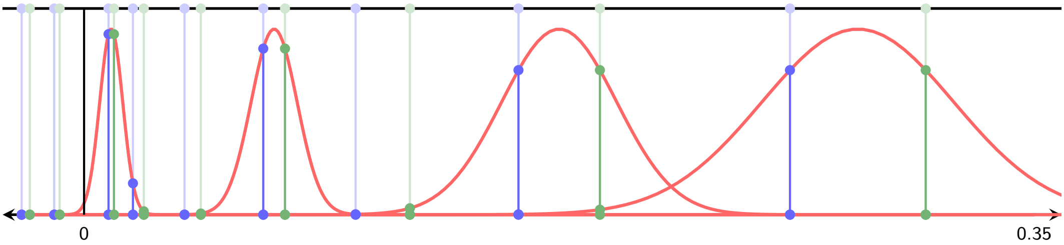
$$\|Y_n - Y_m\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}+\frac{1}{m}\right)\|X\| + A_x(A_h c + 2)\left(\frac{1}{n}+\frac{1}{m}\right) + 4A_h c\|X\|$$

- Lipschitz continuity restriction appears again $\Rightarrow$ transferability-discriminability tradeoff

Ruiz, L., Gama, F., Ribeiro, A., *Graph Neural Networks: Architectures, Stability and Transferability*, IEEE Proceedings, 2021

Ruiz, L., Chamon, L. F. O., Ribeiro, A., *Transferability Properties of Graph Neural Networks*, IEEE TSP, 2023

- If filter is sharp near $\lambda = 0$, spectral components of $\lambda_j(\mathbf{S}_n)$ and $\lambda_j(\mathbf{W})$ are amplified differently



- Transferability and discriminability are not compatible for graph convolutional filters

Graph Neural Networks

- So far we have talked at length about graph convolutions and graphon convolutions

⇒ Graph Convolution

$$\mathbf{z}_n = \sum_{k=0}^{K-1} h_k \mathbf{S}_n^k \mathbf{x}_n$$

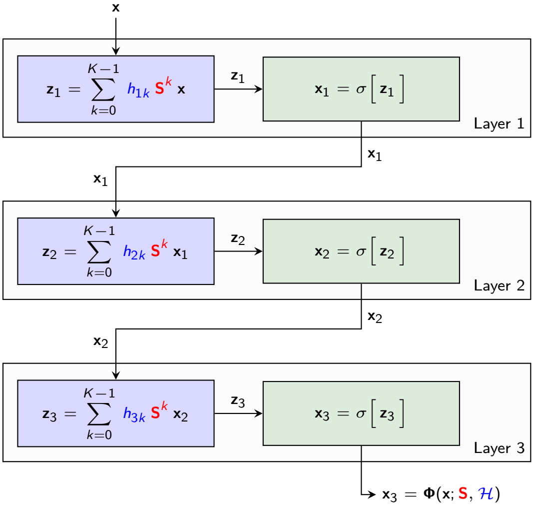
⇒ Graphon Convolution

$$Z = \sum_{k=0}^{K-1} h_k T_{\mathbf{W}}^{(k)} X$$

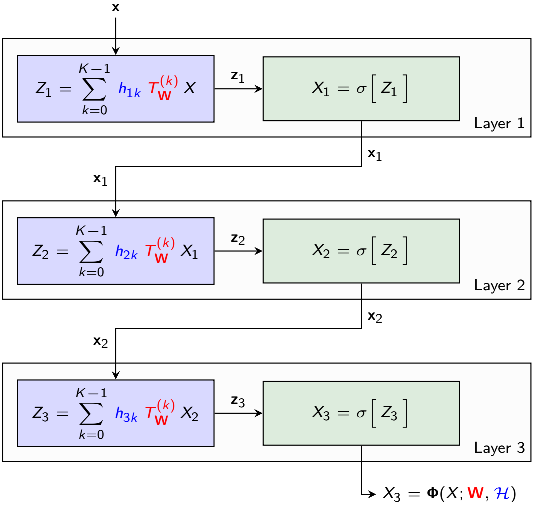
- But we have not talked much about graph neural networks and graphon neural networks

⇒ Graph and graphon NNs are a minor variation of graph convolutions and graphon convolutions

- A graph NN composes a cascade of layers
- Each of which are themselves compositions
  - $\Rightarrow$ Of graph convolutions $\mathbf{H}(\mathbf{S})$
  - $\Rightarrow$ With pointwise nonlinearities σ
- Define the learnable parameter set $\mathcal{H} = \{h_{kl}\}$
- GNN can be represented as $\mathbf{y} = \mathbf{\Phi}(\mathcal{H}; \mathbf{S}; \mathbf{x})$



- A graphon NN (WNN) composes layers
- Each of which are themselves compositions
  - ⇒ Of graphon convolutions $T_{\mathbf{H}}$
  - ⇒ With pointwise nonlinearities σ
- Define the learnable parameter set $\mathcal{H} = \{h_{kl}\}$
- WNN can be represented as $Y = \mathbf{\Phi}(\mathcal{H}; \mathbf{W}; X)$



- The transferability properties of graph filters are inherited by graph neural networks

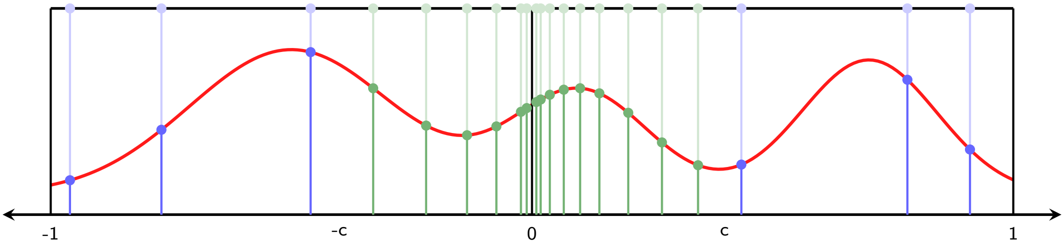
Theorem (GNN Transferability) (Ruiz, L. et al., NeurIPS'20, Proc. IEEE'21)

Consider graph signals $(\mathbf{S}_n, \mathbf{x}_n)$ and $(\mathbf{S}_m, \mathbf{x}_m)$ sampled from graphon signal (W, X) along with GNN outputs $\mathbf{y}_n = \Phi(\mathcal{H}; S_n, x_n)$ and $\mathbf{y}_m = \Phi(\mathcal{H}; S_m, x_m)$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y_m\| \leq$$

$$LF^{L-1} 2A_w \left(A_h + \pi \frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})} \right) \left(\frac{1}{n} + \frac{1}{m} \right) \|X\| + A_x (A_h c + 2) \left(\frac{1}{n} + \frac{1}{m} \right) + 4LF^{L-1} A_h c \|X\|$$

- The difference in GNNs is that the nonlinearities scatter spectral components all over the spectrum

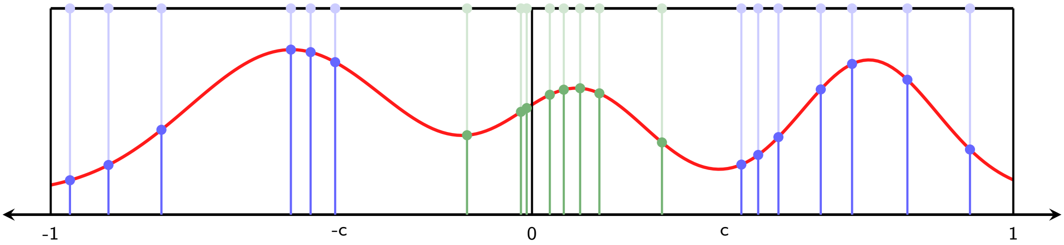


- Which allows increasing discriminability without hurting transferability. Hence:

  $\Rightarrow$ For the same level of transferability $\Rightarrow$ GNNs are more discriminative than graph filters

  $\Rightarrow$ For the same level of discriminability $\Rightarrow$ GNNs are more transferable than graph filters

- The difference in GNNs is that the nonlinearities scatter spectral components all over the spectrum

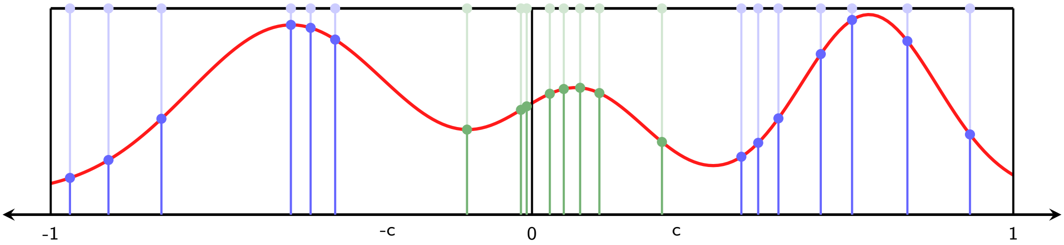


- Which allows increasing discriminability without hurting transferability. Hence:

  $\Rightarrow$ For the same level of transferability $\Rightarrow$ GNNs are more discriminative than graph filters

  $\Rightarrow$ For the same level of discriminability $\Rightarrow$ GNNs are more transferable than graph filters

- The difference in GNNs is that the nonlinearities scatter spectral components all over the spectrum

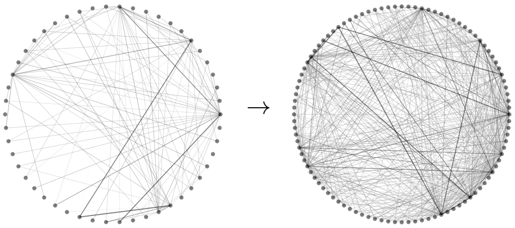


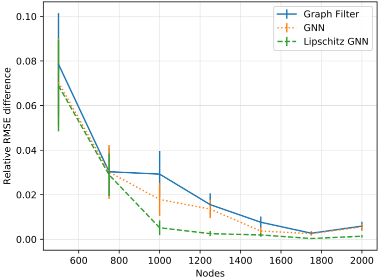
- Which allows increasing discriminability without hurting transferability. Hence:

$\Rightarrow$ For the same level of transferability $\Rightarrow$ GNNs are more discriminative than graph filters

$\Rightarrow$ For the same level of discriminability $\Rightarrow$ GNNs are more transferable than graph filters

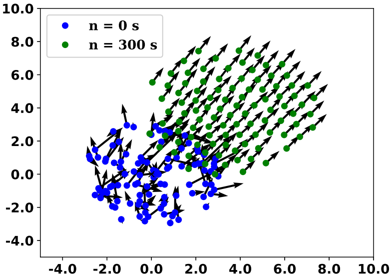
- Transferability of graph neural networks observed empirically ⇒ recommendation system

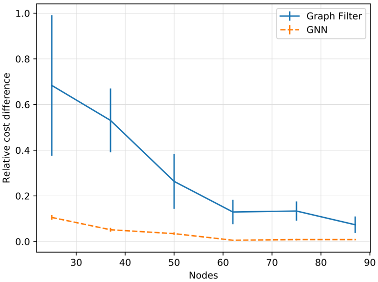




- Performance difference on training and target graphs decreases as size of training graph grows
- GNNs are more transferable than graph convolutional filters. Especially if their filters are Lipschitz

- Transferability of graph neural networks observed empirically $\Rightarrow$ decentralized robot control





- Performance difference on training and target graphs decreases as size of training graph grows

- GNNs are more transferable than graph convolutional filters. Especially if their filters are Lipschitz

GNNs are more transferable than graph convolutional filters

GNNs are more transferable because of their mixing properties

- Empirical and theoretical evidence support using GNNs for large-scale graph machine learning

- Using the transferability property to train GNNs for large graphs $\mathbf{G}_N$ might not be sufficient

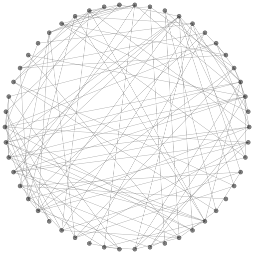
- While difference between the outputs of the same GNN decreases with the training graph size...

$\Rightarrow$ ... no guarantee that the learned GNN will match the ERM solution on the large graph

Idea: **Exploit transferability in the training algorithm, with theoretical guarantees**

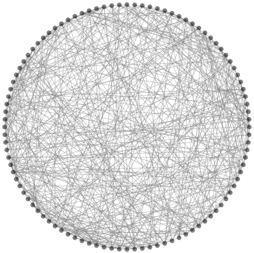
Learning GNNs on a Sequence of Growing Graphs

- We train GNNs on sequences of growing graphs $\Rightarrow$ trade-off between cost and performance



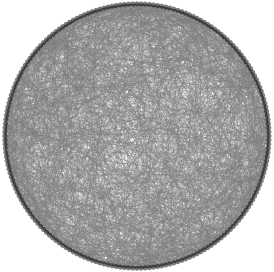
10^2 nodes

$\Rightarrow$



10^3 nodes

$\Rightarrow$



10^4 nodes

- Increase the graph to exploit transferability in gradient approximation $\Rightarrow$ learning by transference

- Goal: obtain coefficients $\mathcal{H}$ that minimize loss ℓ on graphon model $\mathbf{W}$ of large graph

  $\Rightarrow$ On graphon: predict graphon labels Y given graphon signal X

  $\Rightarrow$ On graph: predict node labels $\mathbf{y}$ given graph signal $\mathbf{x}$

Learning Problem on graphon

$$\underset{\mathcal{H}}{\text{minimize}} \quad \mathbb{E}\Big[\ell(Y, \mathbf{\Phi}(X; \mathcal{H}, \mathbf{W}))\Big]$$

Learning Problem on graph

$$\underset{\mathcal{H}}{\text{minimize}} \quad \mathbb{E}\Big[\ell(\mathbf{y}, \mathbf{\Phi}(\mathbf{x}; \mathcal{H}, \mathbf{S}))\Big]$$

- Given $\mathbf{G}_n \to \mathbf{W}$, and reasonable regularity assumptions on $\mathbf{W}$, the problems become increasingly close as $n \to \infty$

Theorem (Gradient Convergence)

For $(\mathbf{G}_n, \mathbf{x}_n) \sim (\mathbf{W}, X)$, under smoothness assumptions, it holds:

$$\mathbb{E}\|\nabla_{\mathcal{H}}\ell(Y, \mathbf{\Phi}(X; \mathcal{H}, \mathbf{W})) - \nabla_{\mathcal{H}}\ell(Y_n, \mathbf{\Phi}(X_n; \mathcal{H}, \mathbf{W}_n))\| \leq \alpha + O\left(\sqrt{\frac{\log(n^{3/2})}{n}}\right)$$

where α is a constant that depends on the GNN depth and width and on the graphon eigengap.

- The gradients $\nabla_{\mathcal{H}}\ell(\mathbf{y}_n, \mathbf{\Phi}(\mathbf{x}_n; \mathcal{H}, \mathbf{G}_n))$ converge to the gradients $\nabla_{\mathcal{H}}\ell(Y, \mathbf{\Phi}(X; \mathcal{H}, \mathbf{W}))$

Learning by Transference: Training GNNs on Growing Graphs

- We want to obtain the coefficients $\mathcal{H}$ that achieve the best performance on the large graph (graphon)

- Gradient convergence allows approximating graphon gradients with gradients on graphs of increasing size

- By successively increasing the number of nodes, we provably follow the learning direction on the graphon

Theorem (Learning by Transference Convergence)

Consider the ERM problem parametrized by the WNN $\mathbf{\Phi}(X; \mathcal{H}, \mathbf{W})$. Fix the step size $\eta < {}^{-1}$. If at step k of epoch e the number of nodes $n(e)$ verifies

$$\mathbb{E}[\|\nabla_{\mathcal{H}}\ell(Y, \mathbf{\Phi}(X; \mathcal{H}_k, \mathbf{W})) - \nabla_{\mathcal{H}}\ell(Y_{n(e)}, \mathbf{\Phi}(X_{n(e)}; \mathcal{H}_k, \mathbf{W}_{n(e)}))\|] + \epsilon < \|\nabla_{\mathcal{H}}\ell(Y, \mathbf{\Phi}(X; \mathcal{H}_k, \mathbf{W}))\|$$

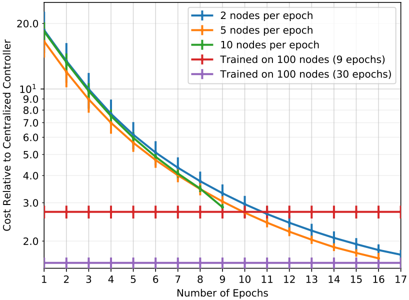
then in at most $k^* = \mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ iterations, the Learning by Transference algorithm converges to an $\alpha + \epsilon$-ball of the solution of the ERM.

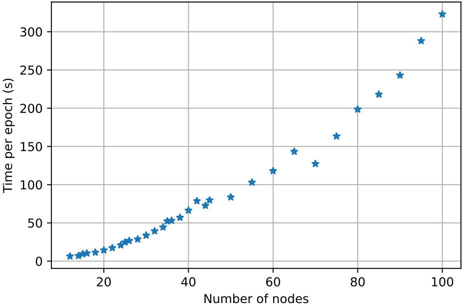
- The optimal WNN can be obtained by taking learning steps on growing GNNs $\Rightarrow$ more efficient

Cerviño, J., Ruiz, L., Ribeiro, A., *Learning by Transference: Training Graph Neural Networks on Growing Graphs*, IEEE TSP, 2023

Cerviño, J., Ruiz, L., Ribeiro, A., *Training Graph Neural Networks on Growing Stochastic Graphs*, ICASSP, 2023

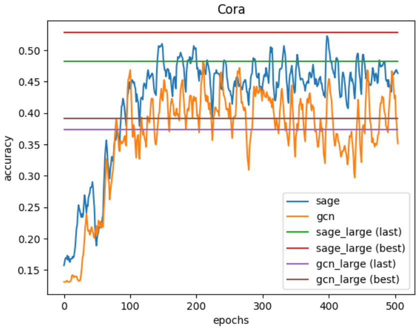
- **Learning by transference** works well in practice and substantially **reduces computational time**

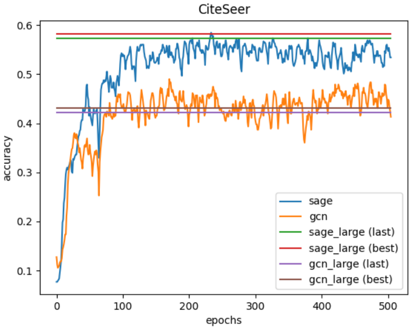




- Starting at 20 agents, we can match 9 epoch performance by training on up to as little as 48 agents
- Starting at 20 agents, we can match 30 epoch performance in less than half the training time

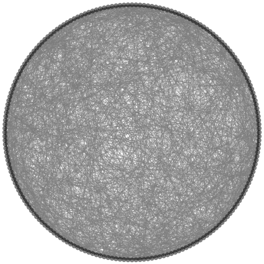
- Starting at 1000 nodes and adding 20 every 5 epochs vs. training on 3000-node graph





$\Rightarrow$ On Cora $\Rightarrow$ total running time decreased by 1.6s for GCN and by 1.3s for GraphSAGE

$\Rightarrow$ On CiteSeer $\Rightarrow$ total running time decreased by 3.6s for GCN and by 8.2s for GraphSAGE



- Graphons are only good limit models for dense graphs

  $\Rightarrow$ Degree is $\Theta(n)$, explodes as $n \to \infty$

- Graphons do not take into account the geometry of the node sample space

Idea: **Manifolds and geometric graphs (next block!)**

Further reading

- **Convergence/transferability papers:**

- **Extensions:**

1. Graphon NTK: helps understand learning dynamics at large-scale
2. Sampling: spectral graphon sampling for cheaper spectral sampling on large graphs
3. Sampling from the feature correlation matrix

1. 2. 3.

Thank you!

Luana Ruiz

Johns Hopkins University

lrubini1@jh.edu
https://luanaruiz9.github.io/

- We fix a bandwidth $c > 0$ to separate eigenvalues not close to $\lambda = 0$ and define

(D1) The c-band cardinality of G_n is the number of eigenvalues with absolute value larger than c

$$B_{nc} = \#\left\{ \lambda_{ni} \,:\, |\lambda_{ni}| > c \right\}$$

(D2) The c-eigenvalue margin of of graph G_n is the

$$\delta_{nc} = \min_{i, j \neq i} \left\{ |\lambda_{ni} - \lambda_j| \,:\, |\lambda_{ni}| > c \right\}$$

- Where λ_{ni} are eigenvalues of the shift operator $\mathbf{S}_n$ and λ_j are eigenvalues of graphon W

(A1) The graphon W is A_w-Lipschitz $\Rightarrow$ For all arguments (u_1, v_1) and (u_2, v_2), it holds

$$\Big| \mathbf{W}(u_2, v_2) - W(u_1, v_1) \Big| \leq A_w \Big(\big| u_2 - u_1 \big| + \big| v_2 - v_1 \big| \Big)$$

(A2) The filter's response is A_h-Lipschitz and normalized $\Rightarrow$ For all λ_1, λ_2 and λ we have

$$\Big| h(\lambda_2) - h(\lambda_1) \Big| \leq A_h \Big| \lambda_2 - \lambda_1 \Big| \quad \text{and} \quad \Big| h(\lambda) \Big| \leq 1$$

(A3) The graphon signal X is A_x-Lipschitz $\Rightarrow$ For all u_1 and u_2

$$\Big| X(u_2) - X(u_1) \Big| \leq A_x \Big| u_2 - u_1 \Big|$$