

Graph Signal Processing using Deep Neural Networks

Jayaraman J. Thiagarajan



Joint Work With



Rushil Anirudh
LLNL



Bhavya Kailkhura
LLNL



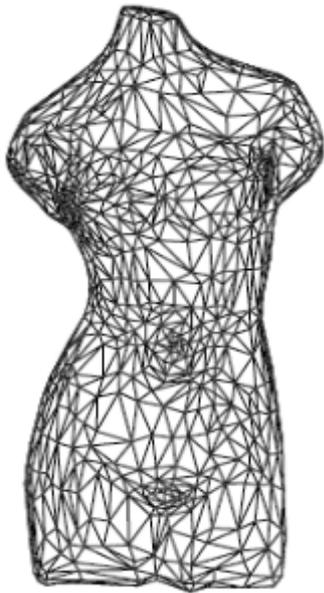
Prasanna Sattigeri
IBM Research



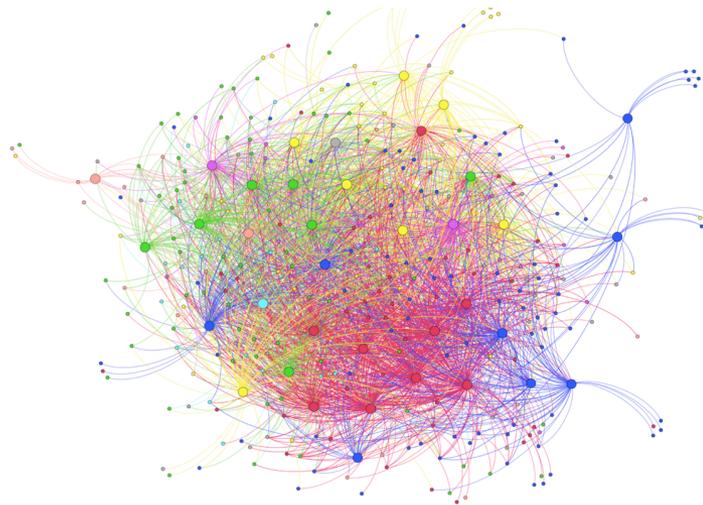
Karthi Ramamurthy
IBM Research

Graphs are Everywhere

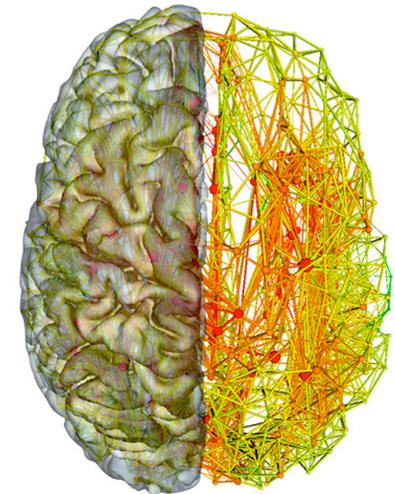
Graphics



Social Networks

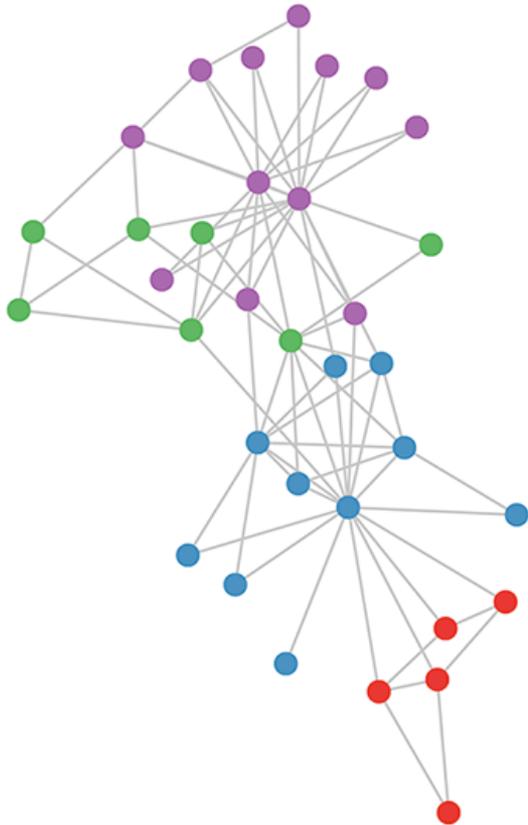


Medicine



Graphs can capture complex relational characteristics for analysis and inference

Undirected Graphs – Basic Notations



Graph $G = (\mathcal{V}, E, w)$.

Adjacency matrix \mathbf{A}

Degree matrix $\mathbf{D} = \text{diag}\{d_i\}$

Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{A}$.

Normalized Laplacian matrix

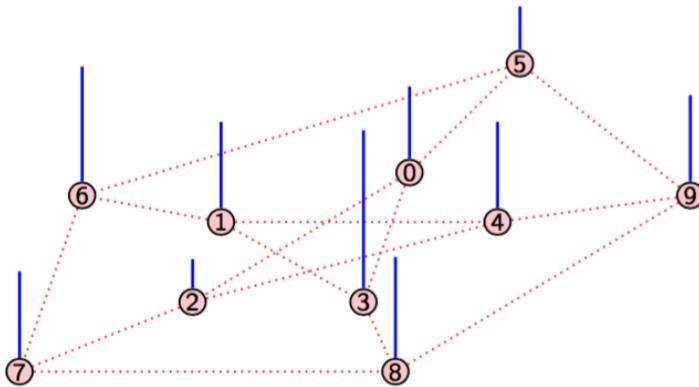
$$\mathcal{L} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2}$$

Signals on a Graph can be as Interesting as the Graph Itself

Graph signal: a function $f : \mathcal{V} \rightarrow \mathbb{R}$ that assigns real values to each vertex of the graph

■ Examples:

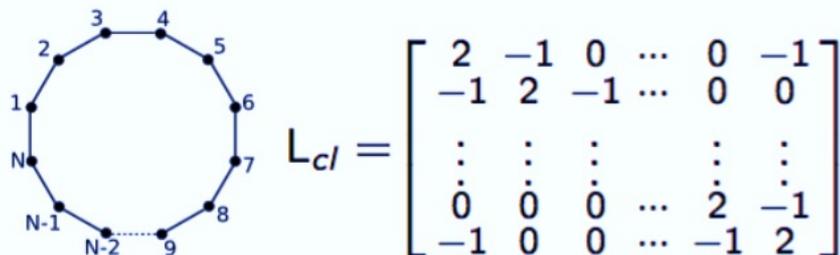
- Brain networks and fMRI signals
- Gene regulatory networks and gene expression levels
- Social networks and information cascades



Analyze characteristics of the signal by exploiting the graph structure

Connection to Fourier Analysis – Eigenvectors as Frequencies

THE “CLASSICAL” GRAPH :



All classical Fourier modes are the eigenvectors of L_{cl}

ANY GRAPH :



By analogy, any graph's Fourier modes are the eigenvectors of its Laplacian matrix L .

Graph Fourier Transform

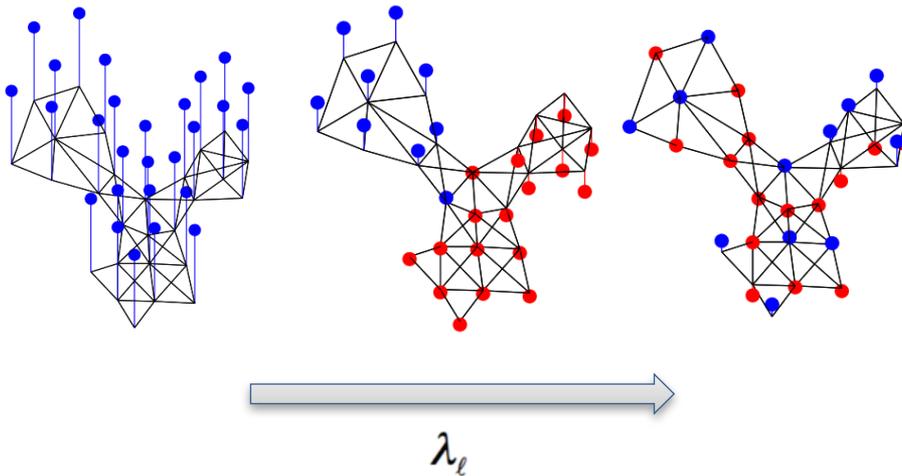
- Using the eigenvectors of the Laplacian, we can define

GFT

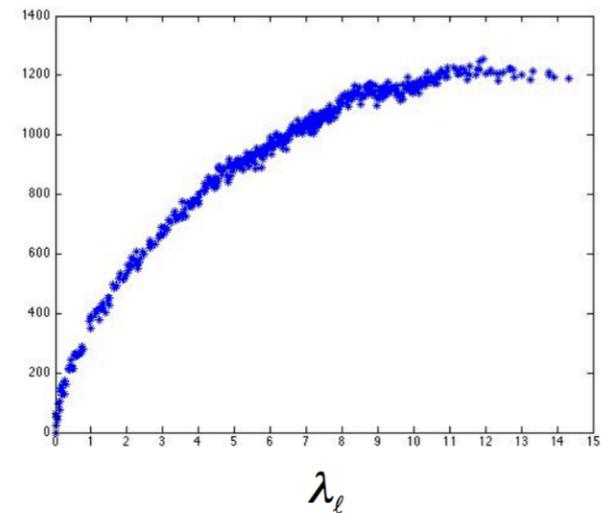
$$\hat{f}(\lambda_\ell) := \langle \mathbf{f}, \mathbf{u}_\ell \rangle = \sum_{i=1}^N f(i) u_\ell^*(i)$$

IGFT

$$f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\lambda_\ell) u_\ell(i)$$



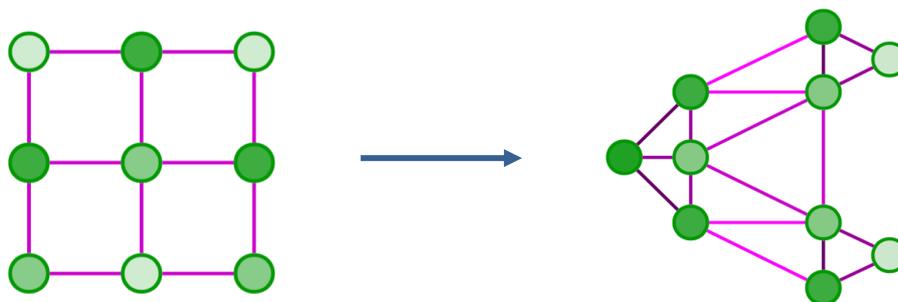
Number of zero crossings



Graph Filtering is the Key to Designing Graph Convolutional Neural Networks

Graphs vs Euclidean grids

- ▶ Irregular sampling.
- ▶ Weighted edges.



A **graph filter** $H : \mathbb{R}^N \rightarrow \mathbb{R}^N$ is a **map** between **graph signals**

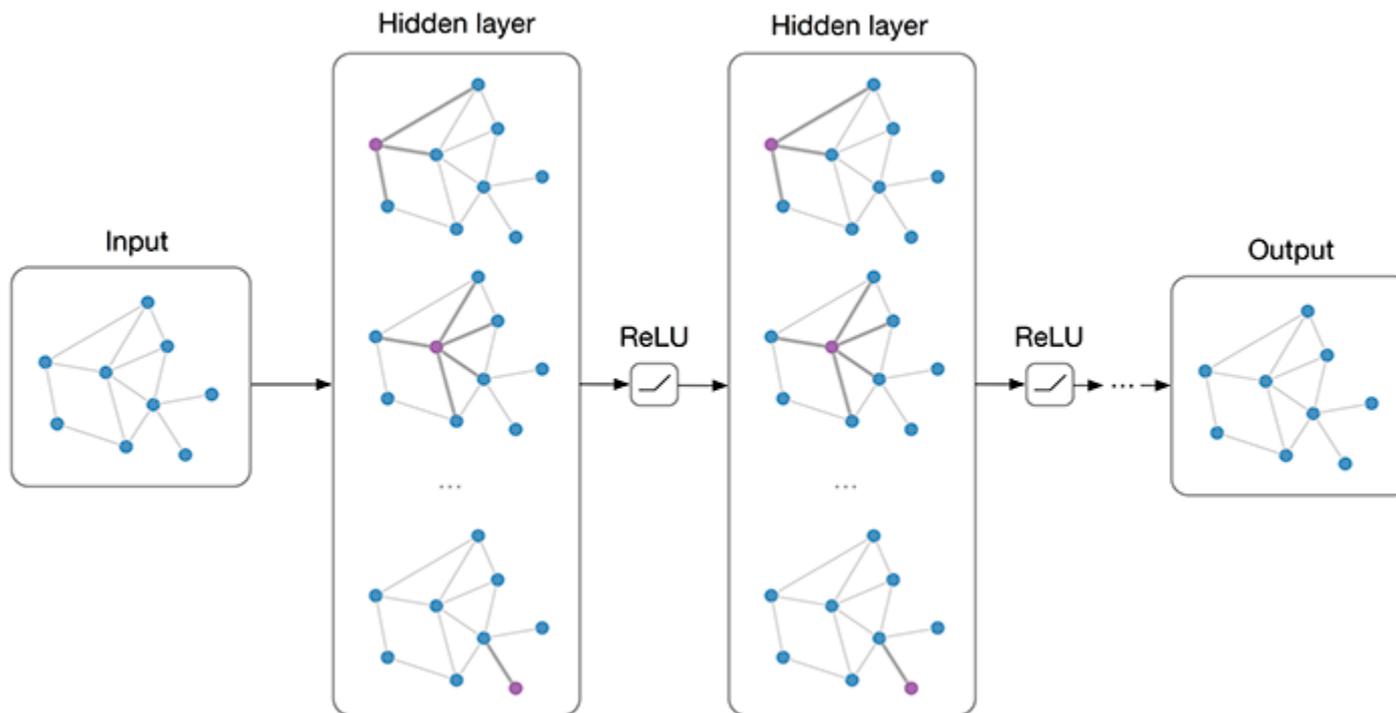
Focus on linear filters

\Rightarrow map represented by an $N \times N$ matrix



$$\hat{f}_{out}(\lambda_\ell) = \hat{f}_{in}(\lambda_\ell) \hat{h}(\lambda_\ell)$$

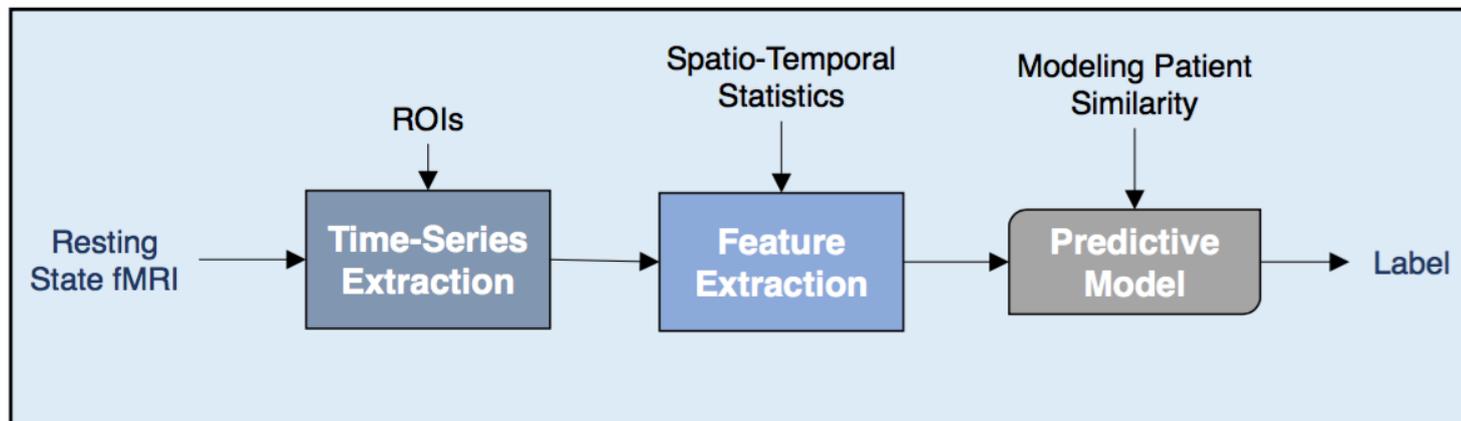
Graph Convolutional Networks Learn a Sequence of Graph Filters



Chebyshev Approximation

$$g_{\hat{\theta}} \star \mathbf{x} = \hat{\theta}_0 \mathbf{x} - \hat{\theta}_1 \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2} \mathbf{x}.$$

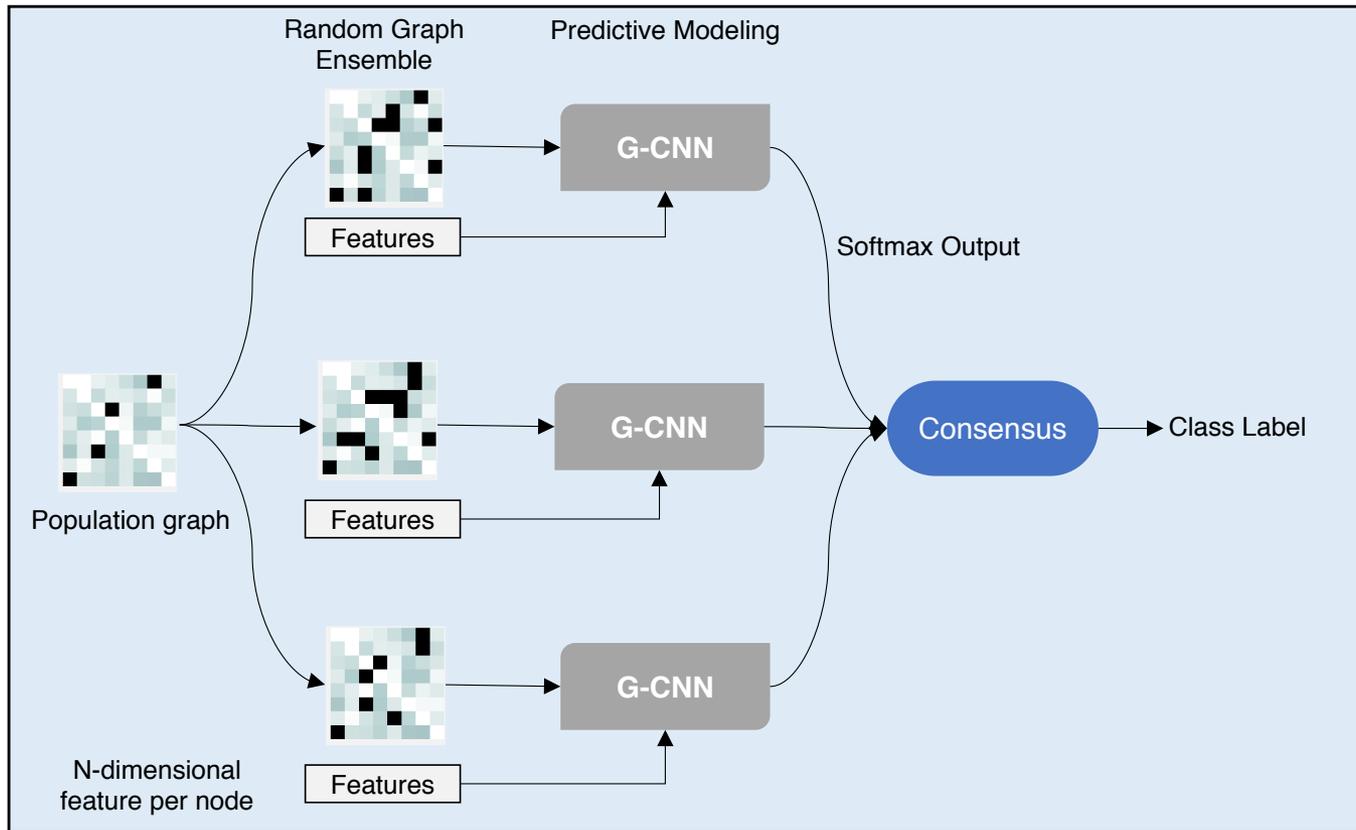
Problem: Using Graph CNN for Autism Spectrum Disorder Classification



Neuropathology studies map variations in brain functionality to clinical measures

Population Graph Design	Signal Construction
Non-imaging features (e.g. gender/site) or combination of imaging and non-imaging features	Statistics from imaging features (e.g. fMRI)

Fine-Tuning Graph CNNs is Highly Challenging due to their Sensitivity to Graph Design



We adopt a bootstrapping approach that performs dropout in the input layer of the network

Results: Autism Brain Imaging Data Exchange Initiative

- 872 patients from 20 different sites
- Signal Construction: Upper-Triangular part of covariance matrix + PCA

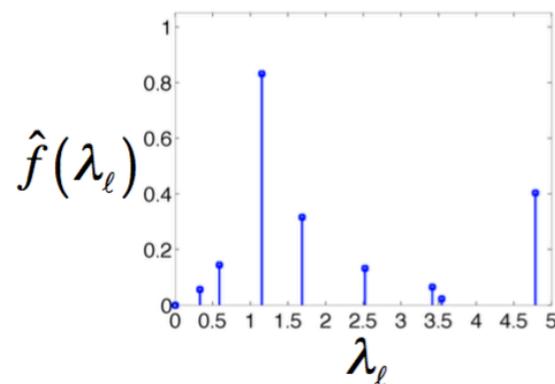
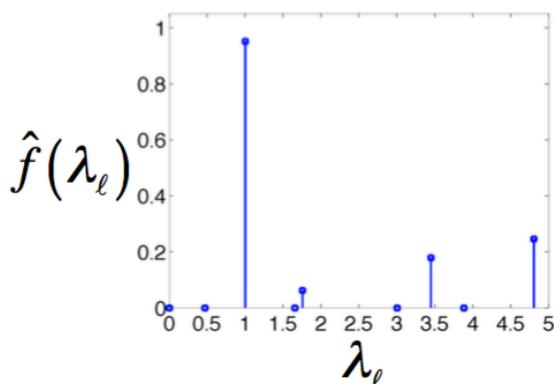
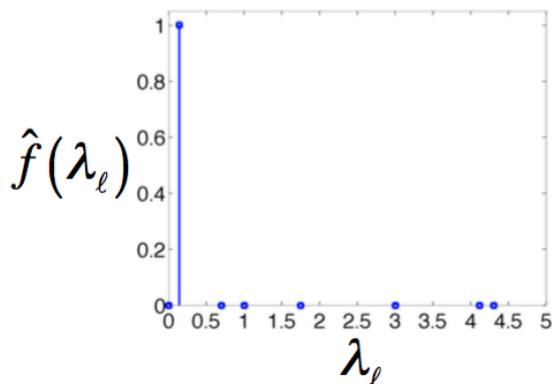
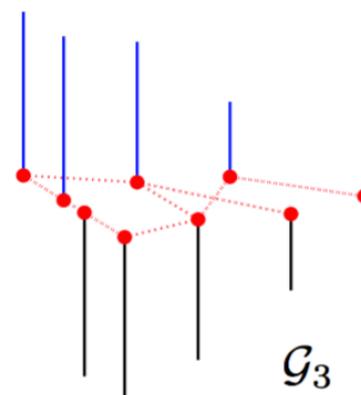
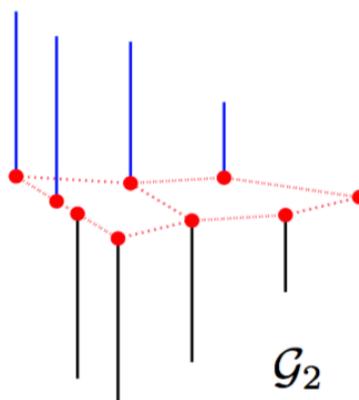
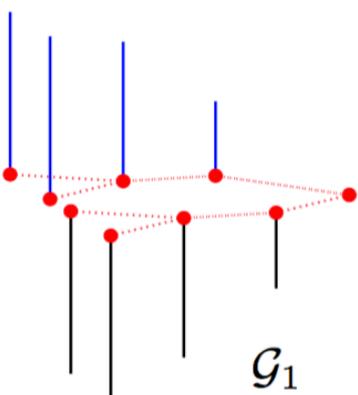
Baselines	
Linear SVM	64.71
Kernel SVM	65.72

Ensemble G-CNN	
(Gender, Site)	67.7
Linear Kernel + (Gender, Site)	67.7
Graph Kernel + (Gender, Site)	68.28

Bootstrapped Graph CNNs improve the prediction accuracy on the challenging ABIDE dataset

Problem: Neighborhood Graph Construction is Sensitive to Sample Density and Noise

- Same signal defined on different graphs can lead to completely different results.



Idea 1: Graph Auto-Encoders can Recover the Optimal Embeddings for Data

- Auto-encoders aim to minimize the reconstruction loss

$$Loss(\theta_1, \theta_2) = \sum_{i=1}^N l(x_i, g(f(x_i; \theta_1); \theta_2))$$

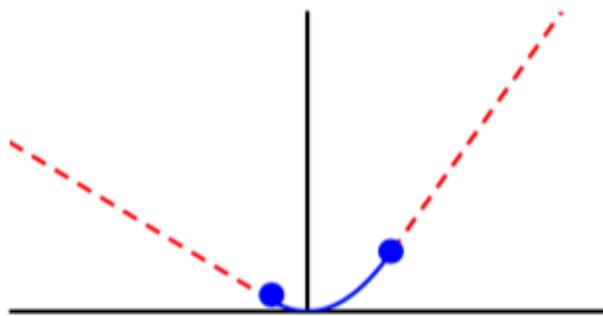
- The “reconstruction nature” of spectral clustering can be viewed as an auto-encoder

 **Laplacian Matrix**

$$\arg \min_{\tilde{\mathbf{W}}} \|\mathbf{W} - \tilde{\mathbf{W}}\|_F = \mathbf{U}\tilde{\Sigma}\mathbf{V}^T, \text{ s.t. } \text{rank}(\tilde{\mathbf{W}}) = k$$

- Using L2 loss – Hidden features will converge to the smallest eigenvectors of the Laplacian matrix

Idea 2: Spectral Embedding with Quantile Loss Enables Robust Sparsification of Graphs



Piecewise Linear Quadratic Loss typically used to explore heterogeneous datasets

$$\rho_{\tau}(r) = \begin{cases} \tau|r| - \frac{\kappa\tau^2}{2} & \text{if } r < -\tau\kappa, \\ \frac{1}{2\kappa}r^2 & \text{if } r \in [-\kappa\tau, (1-\tau)\kappa], \\ (1-\tau)|r| - \frac{\kappa(1-\tau)^2}{2}, & \text{if } r > (1-\tau)\kappa. \end{cases}$$

$$\min_{\mathbf{L}, \mathbf{R}} \rho_{\tau}(\mathbf{W} - \mathbf{L}\mathbf{R}^T)$$

Low-rank decomposition of the similarity matrix

Idea 2: Spectral Embedding with Quantile Loss Enables Robust Sparsification of Graphs

When the quantile parameter is large:

- Positive residuals will be penalized lesser, i.e. the approximation will underestimate the similarity function

When the quantile parameter is small:

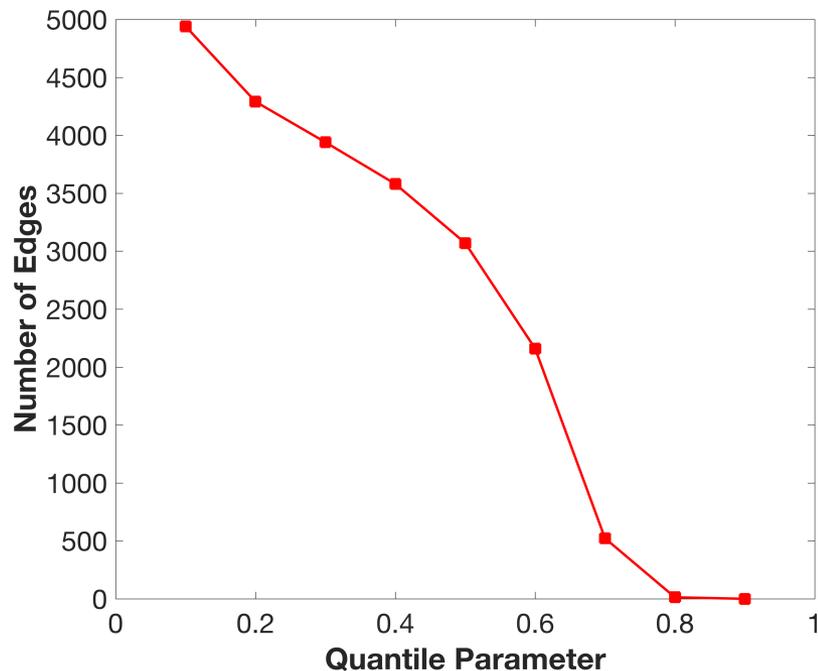
- Returns a dense graph with overestimated edge strengths

$$\min_{\mathbf{L}, \mathbf{R}} \rho_{\tau}(\mathbf{W} - \mathbf{L}\mathbf{R}^T)$$

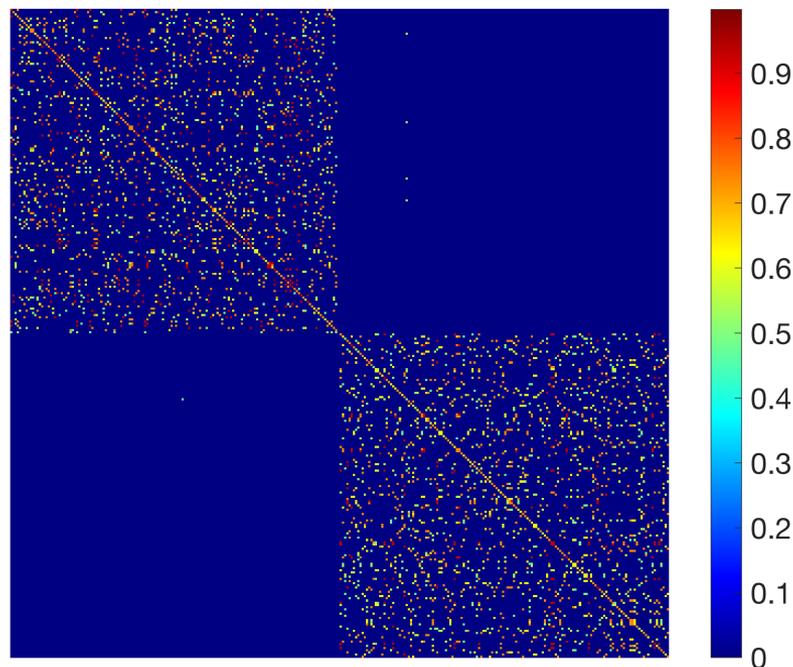
Low-rank decomposition of the similarity matrix

Conditional Quantiles of the Similarity Function Reveal the Importance of Edges

Rate of Decay



Edge Probabilities



$$p(e_{ij}) = \max \left(\delta, 1 - \frac{\beta_{\hat{\tau}}}{\beta_{0.1}} \right)$$

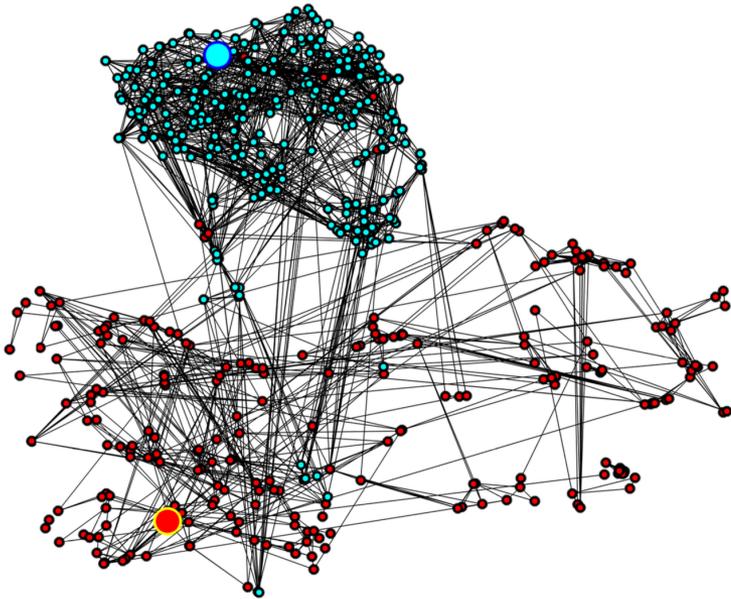
Minimum probability for any edge

Highest quantile at which edge persists

Number of edges at 0.1 quantile

Using the Locally-Scaled Graphs in Semi Supervised Label Propagation

- Given a few labeled examples and a neighborhood graph, propagate labels to all samples – Greedy random walk



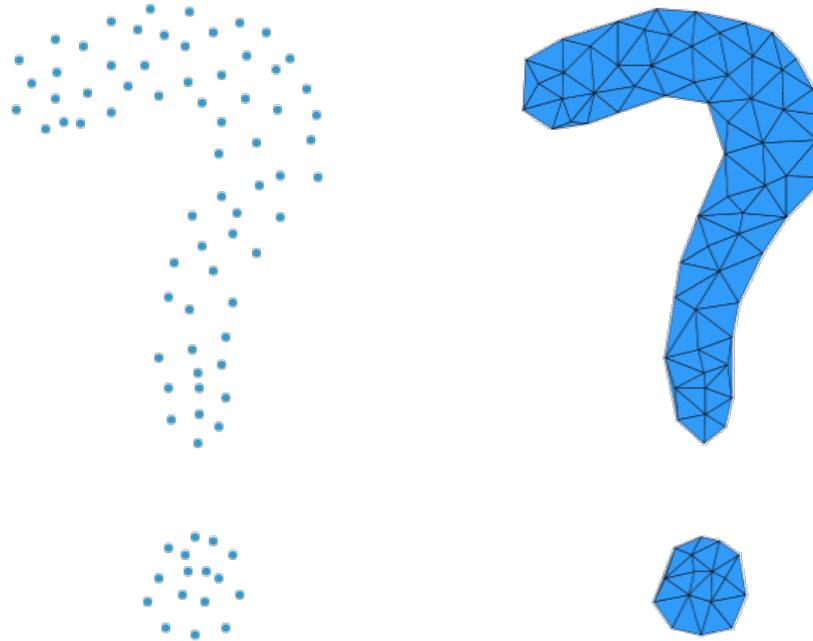
Dataset	Local Scaling	Path-Based	Proposed Approach
Blood Transfusion	76.1	77.1	84.9
Breast Cancer	82.7	84.9	90.5
Echocardiogram	70.1	73.2	88.49
Kidney Disease	66.8	67.5	71.9
SPECT Heart	70	68.5	86
Thoracic Surgery	65.3	66.2	75.8
<u>Arcene</u>	59	61.4	71.3

Extreme Case – 1 Labeled Node

Conclusions

- Graph signal processing provides a convenient framework for a wide variety of data analysis problems.
- Can be highly effective in fusing imaging and non-imaging features in a neural network setting
- Generalized deep networks for data on non-uniform grids – *the new kid on the block*
- Use of appropriate loss functions can lead to robust graph constructions
- Applies to problems in supervised, unsupervised, semi-supervised, transfer learning...

Questions?



Contact

Jayaraman J. Thiagarajan
Center for Applied Scientific Computing
Lawrence Livermore National Laboratory
Email: jjayaram@llnl.gov