Framelets and Graph Neural Networks

Yu Guang Wang

SJTU INS & Math
yuguang.wang@sjtu.edu.cn

SJTU Math Salon 2021

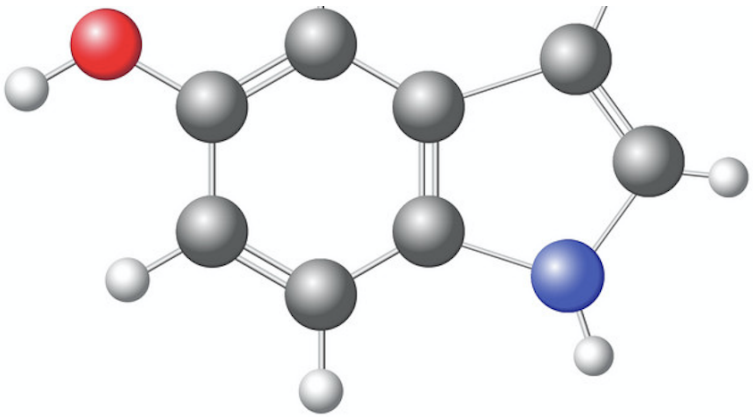
上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

What are Graph Neural Networks?

Graph neural networks (GNNs) are deep nets which take graph-structure data as input.

- The structure of a typical GNN is similar as CNN, which has multiple neural layers, each of which contains a number of neuronal nodes.
- The input of GNN is graph-structured data which usually consists of adjacency matrix (representing edges) plus features on vertices.

Inputs of GNN and CNN



Graph data for GNN

nodes, edges

size varies

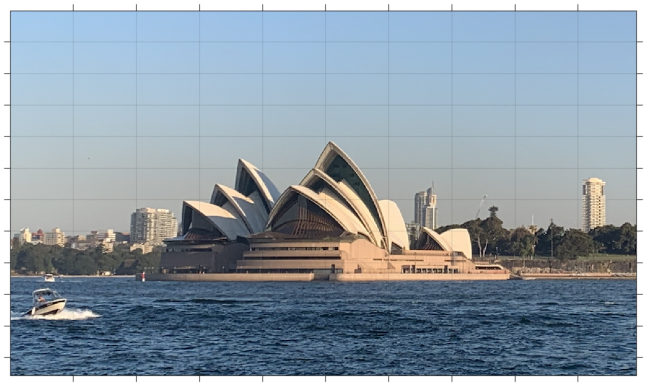
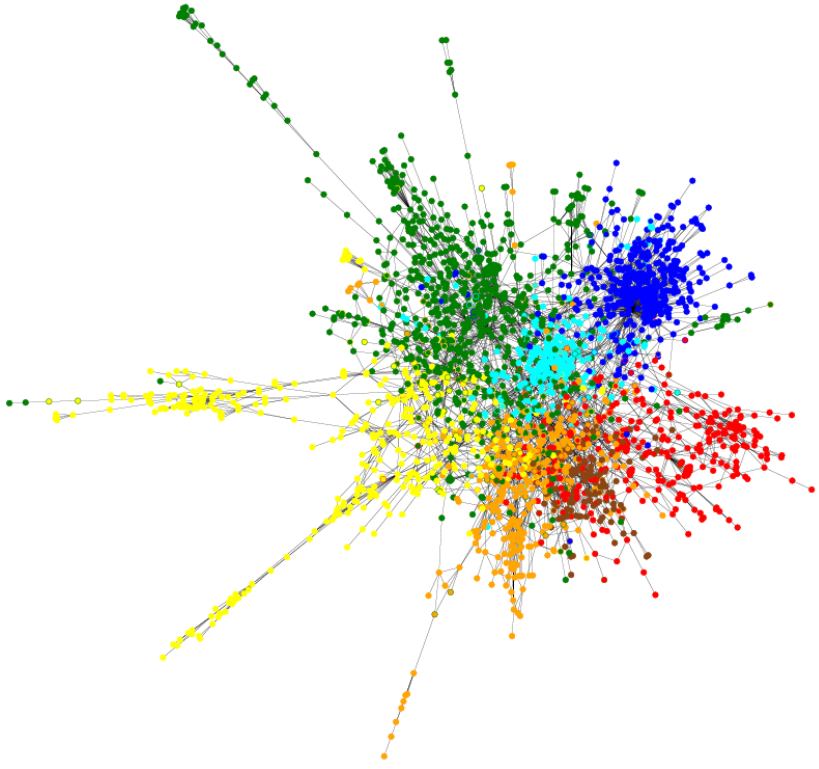


Image for CNN

pixels

regular grid

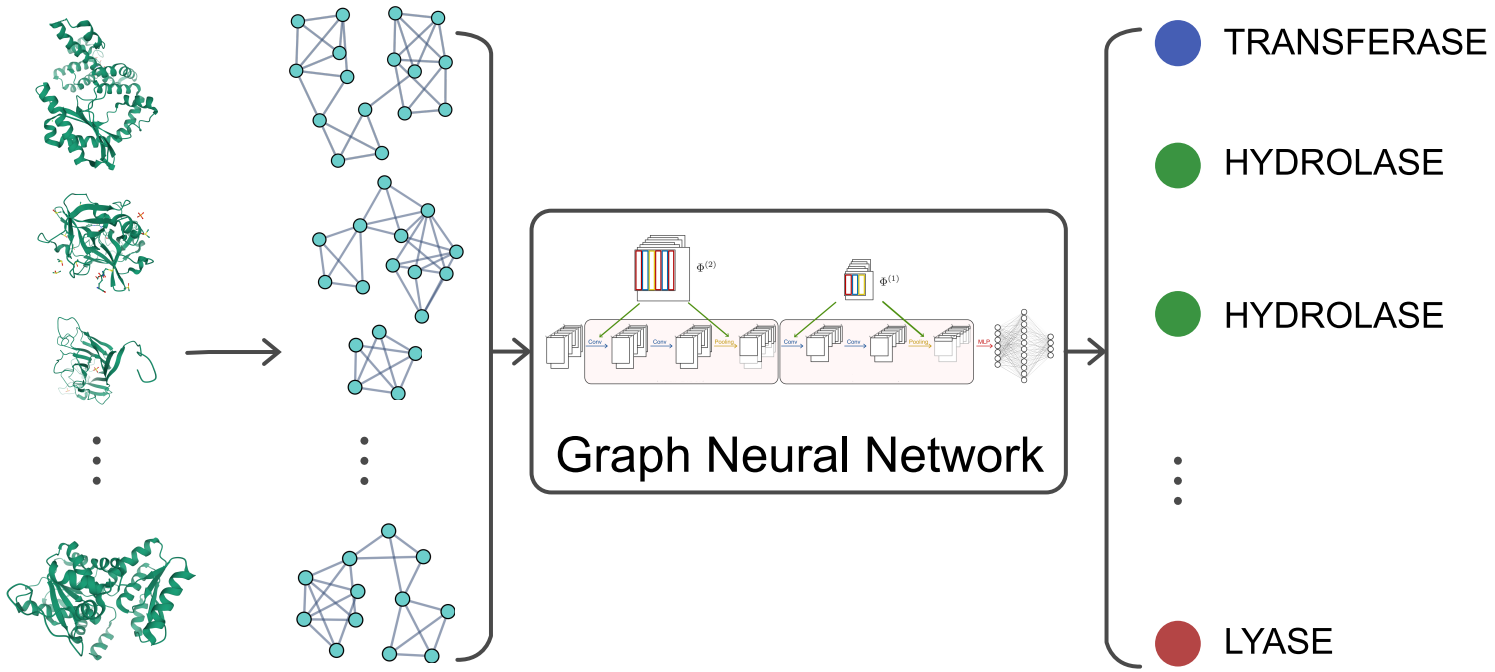
What problems can Graph Neural Networks solve?



Node property prediction (node classification, drug repositioning (knowledge graph), recommender systems e.g. e-commerce)

- The input is one graph.
- The GNN uses the known labeled nodes and the graph edges to infer the missing labels.

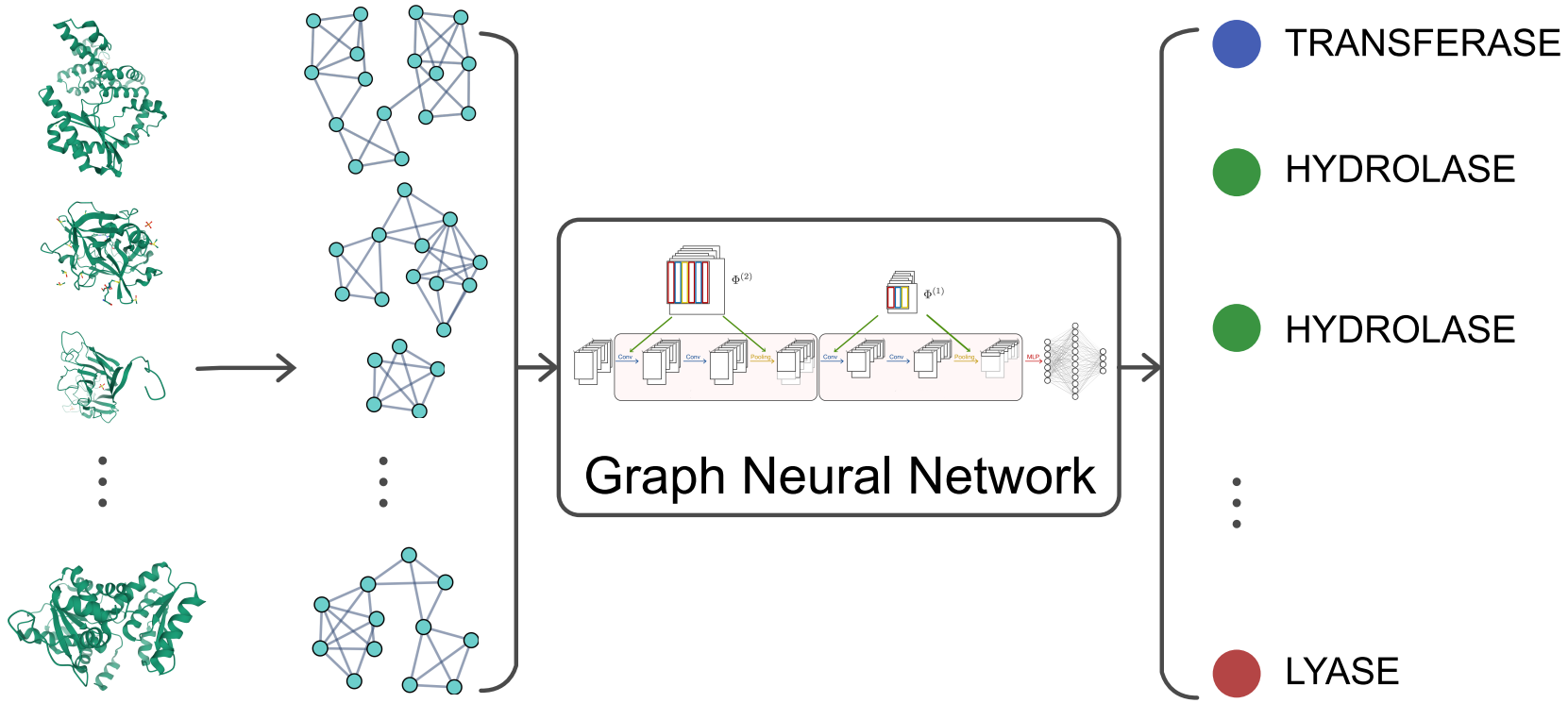
What problems can Graph Neural Networks solve? (Continued)



Graph property prediction (graph classification/regression, graph generation, 3D object recognition e.g. LiDAR in self-driving)

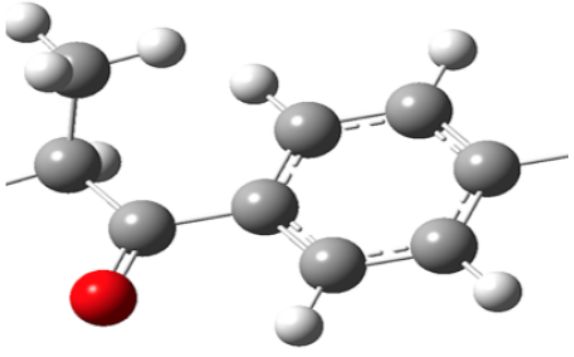
- The input is a set of graphs. Each graph sample has a Y-label.
- The graph samples with known Y-labels and features on the graphs are used in training for GNN. The trained GNN model can be used to infer the missing labels.

Graph Classification for Enzymes



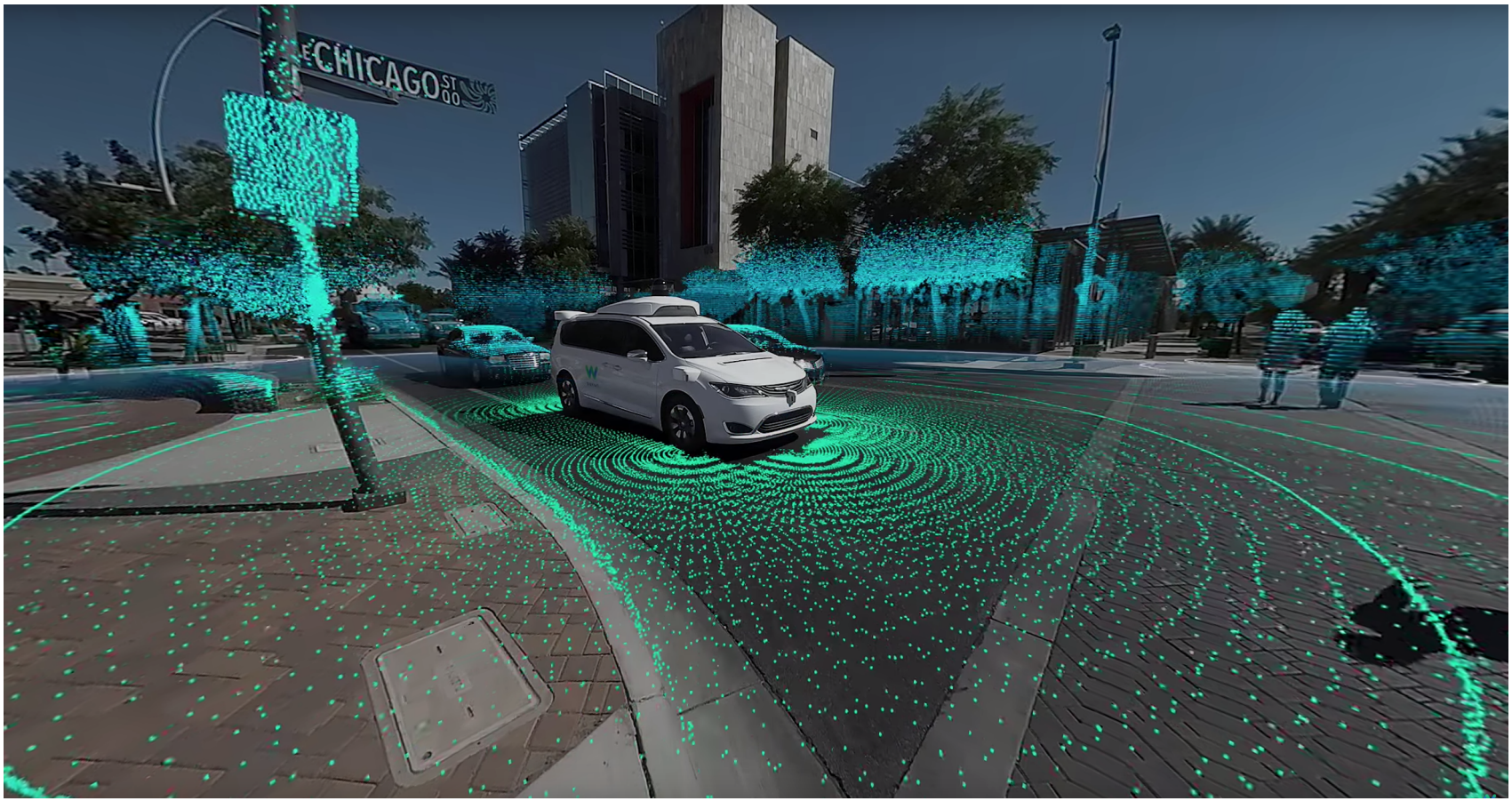
GNNs for labeling enzymes. A set of graphs is extracted as the input data set, where each graph represents a specific protein tertiary structure. The task is to assign each enzyme instance to one of the given EC top-level classes correctly. The GNN's role is to find universally applicable rules to label the graphs by learning the topological and feature information of the input. The structure of enzymes is retrieved from the Protein Data Bank.

Quantum Chemistry Graph Regression



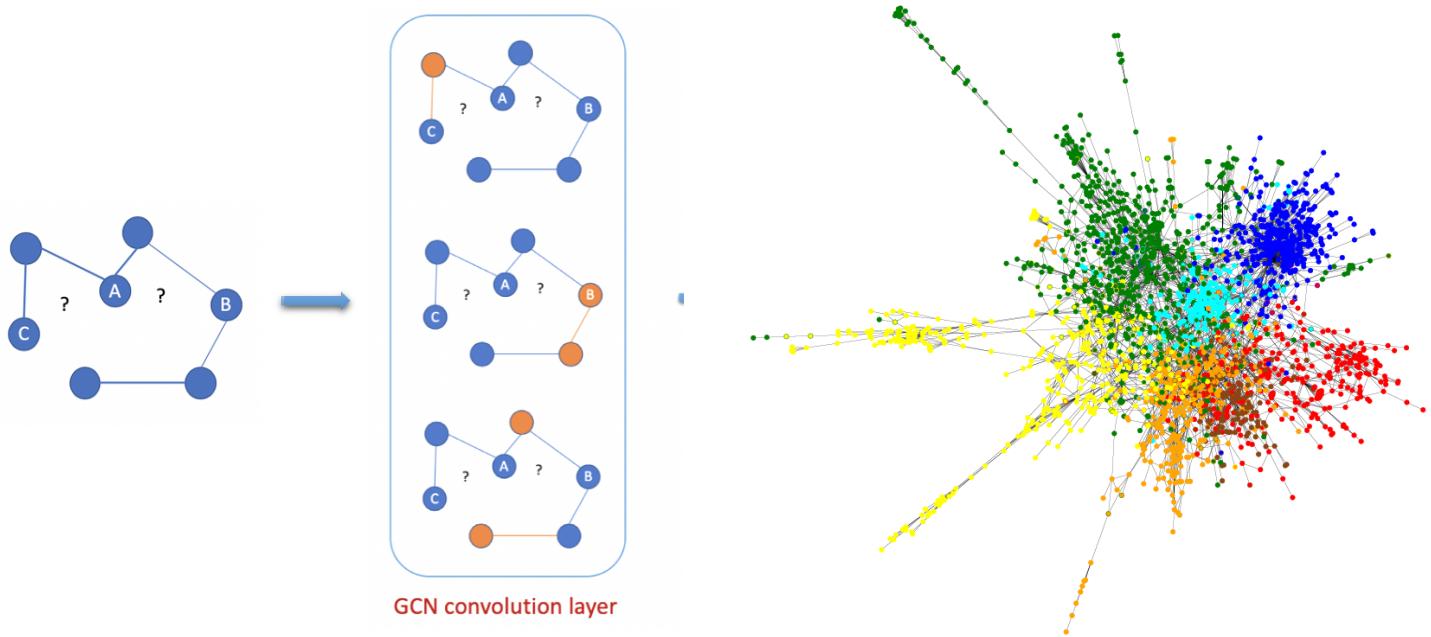
- QM7 is a collection of $7,165$ molecules, train/test $= 4/1$.
- Each molecule contains ≤ 23 atoms (including C, O, N, S), atoms are connected by bonds, molecular structure varies (e.g. double/triple bonds, cycles, carboxy, cyanide ...).
- Molecule is a graph, atoms are nodes, bonds are edges and Coulomb energy as weights, then Coulomb energy matrix is adjacency matrix.
- Task: to predict atomization energy of molecule given the molecular structure.

3D Object Recognition for Self-driving LiDAR



Source: Waymo

What makes Graph Neural Networks function?



Spectral graph convolution $g \star f = U\big((U^T g) \odot (U^T f)\big)$

- Motivated by traditional Fourier convolution
- Have similar role as convolution in CNN
- Can preserve the structural information of graph data input.
- More general form of neural message passing *Gilmer et al. (ICML 2017).*

Framelet convolution

For network filter $\boldsymbol{\theta}$ and input graph feature $X \in \mathbb{R}^{N\times d}$ of the graph $\mathcal{G}$ with N nodes, we define

$$\boldsymbol{\theta} \star X = \mathrm{ReLU}\left(\mathcal{V}\left(\mathrm{diag}(\boldsymbol{\theta})(\mathcal{W}X')\right)\right),\ X' = XW.$$

- $\mathcal{W}X'$ is the framelet coefficient matrix for the transformed X'
- $\mathcal{W}$ is a sequence of $nJ+1$ transform matrices (each of size $N \times N$) for low-pass and high-passes
- The size of the vector $\boldsymbol{\theta}$ is $(nJ+1)N$ which matches the total number of the framelet coefficients for each feature.
- The network filter $\boldsymbol{\theta}$ lies in the frequency domain, each component of which is multiplied to the corresponding row of $\mathcal{W}X'$.
- The matrix W is a trainable weight matrix with dimension $d \times d'$.

Graph framelets

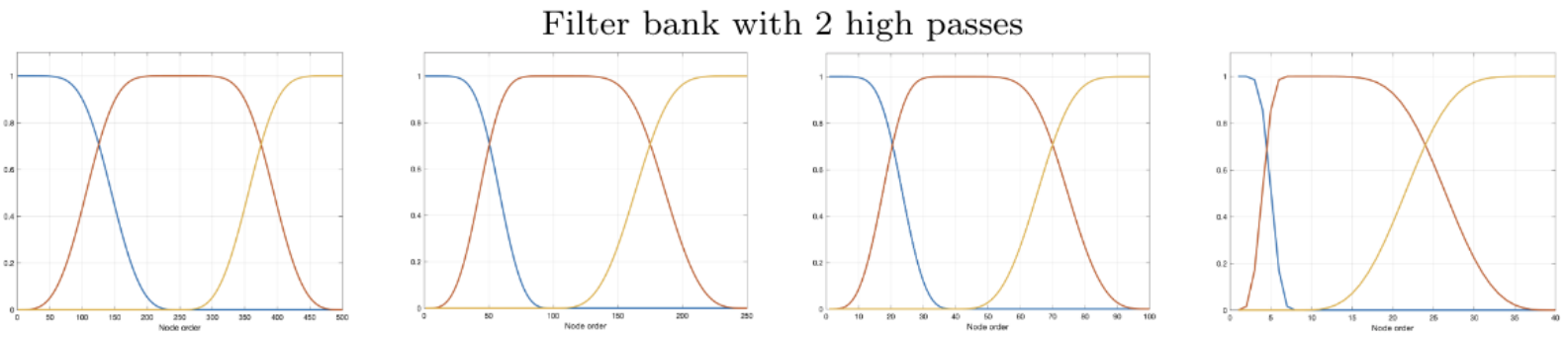
Suppose $\{(\lambda_\ell, \boldsymbol{u}_\ell)\}_{j=1}^N$ are the eigenvalue and eigenvector pairs for $\mathcal{L}$ of graph $\mathcal{G}$ with N nodes. The undecimated framelets at scale level $j = 1, \ldots, J$ for graph $\mathcal{G}$, for $n = 1, \ldots, r$, by

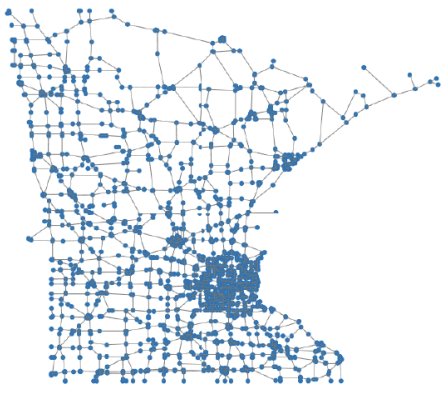
$$\boldsymbol{\varphi}_{j,p}(v) = \sum_{\ell=1}^{N} \widehat{\alpha}\left(\frac{\lambda_\ell}{2^j}\right) \overline{\boldsymbol{u}_\ell(p)} \boldsymbol{u}_\ell(v)$$

$$\boldsymbol{\psi}_{j,p}^n(v) = \sum_{\ell=1}^{N} \widehat{\beta^{(n)}}\left(\frac{\lambda_\ell}{2^j}\right) \overline{\boldsymbol{u}_\ell(p)} \boldsymbol{u}_\ell(v)$$

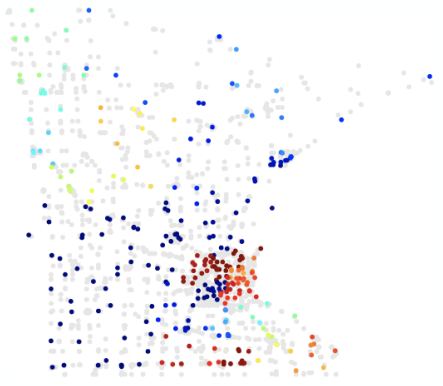
- $\boldsymbol{\varphi}_{j,p}$ or $\boldsymbol{\psi}_{j,p}^r$ is the low-pass or high-pass framelet translated at node p.
- The low-pass and high-pass framelet coefficients for a signal f on graph $\mathcal{G}$ are $v_{j,p}$ and $w_{j,p}^r$, which are the projections $\langle \boldsymbol{\varphi}_{j,p}, f \rangle$ and $\langle \boldsymbol{\psi}_{j,p}^r, f \rangle$ of the graph signal onto framelets at scale j and node p.
- The dilation factor is 2^j with the dilation (base) 2.

Framelet transforms via filter bank (Minnesota traffic network)

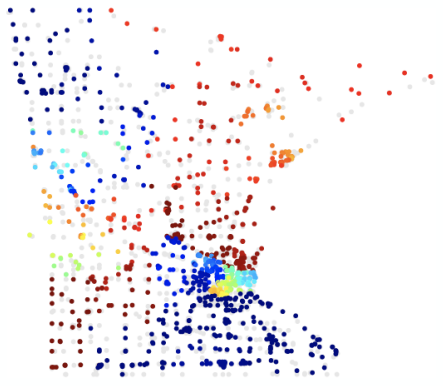




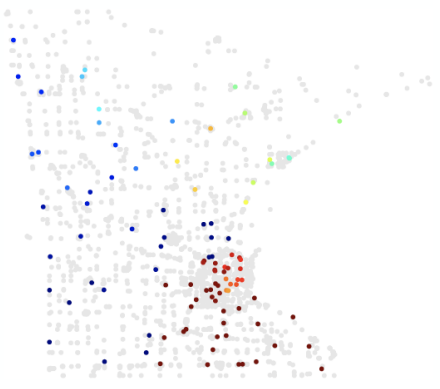
(a) Original Network



(c) $\hat{\mathbf{w}}_0^1$

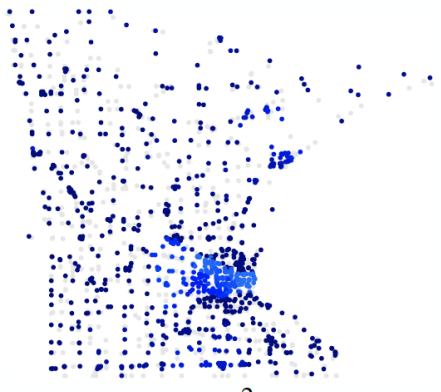


(e) $\hat{\mathbf{w}}_1^1$

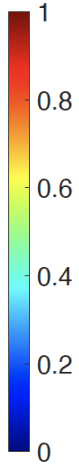


(b) $\hat{\mathbf{v}}_0$

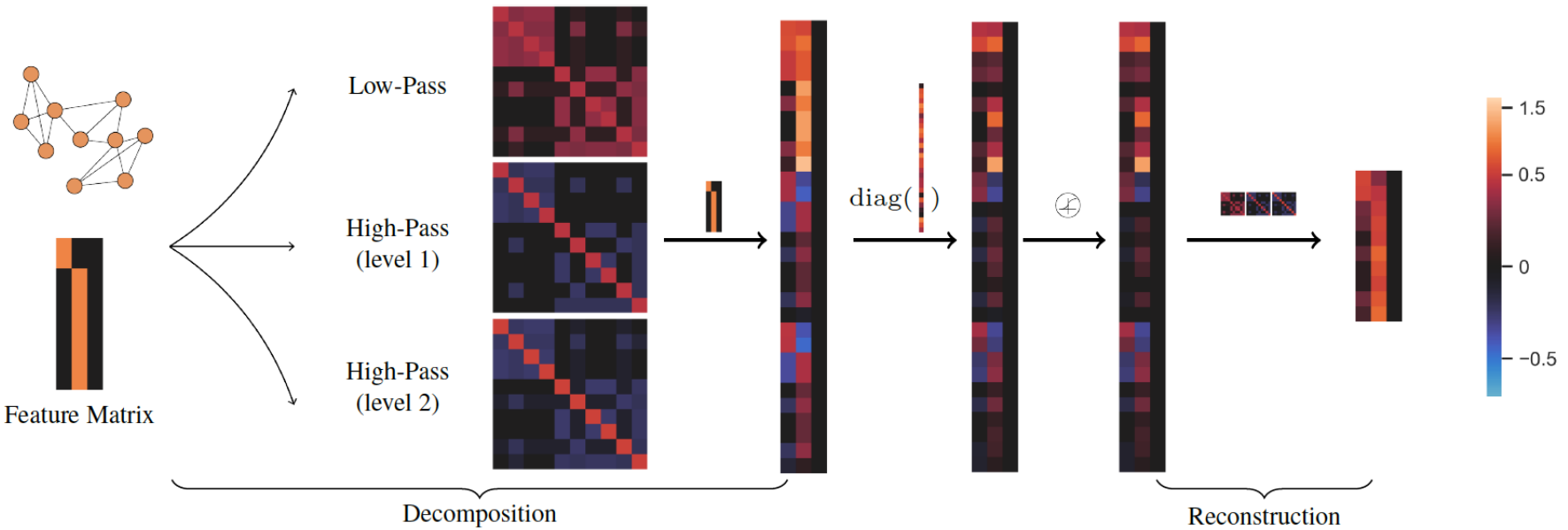
(d) $\hat{\mathbf{w}}_0^2$



(f) $\hat{\mathbf{w}}_1^2$



Shrinkage activation



$$\boldsymbol{\theta} \star X = \mathcal{V}\left(\text{Shrinkage}\left(\text{diag}(\boldsymbol{\theta})(\mathcal{W}X')\right)\right), \; X' = XW.$$

- Different from ReLU, the shrinkage activation works in the framelet domain when the shrinkage thresholds for the high-pass coefficients in the framelet domain.

Shrinkage for signal compression

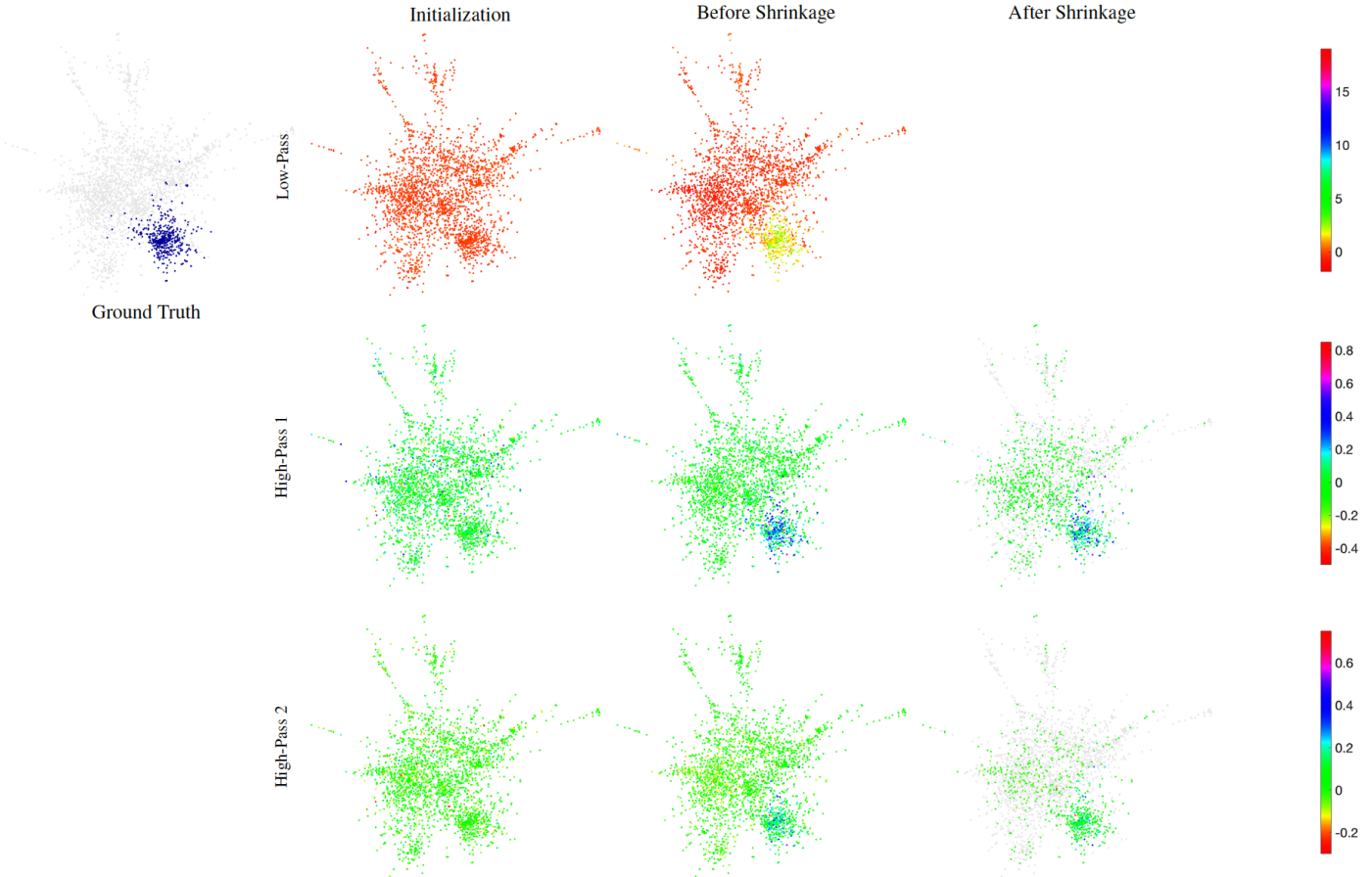
The high-pass coefficients in the frequency domain can be cut off by shrinkage thresholding. For example, the *soft-thresholding* *Donoho (1994), Donoho (1995), Tibshirani (1996).* defines

$$\text{Shrinakge}(\text{x}) = \text{sgn}(\text{x})(|\text{x}| - \lambda)_+ \quad \forall \text{x} \in \mathbb{R},$$

where λ is the threshold value.

- Any x with its absolute value less than λ shall return to zero.
- Applying the above soft-thresholding to the shrinkage activation in framelet convolution only influences small high-pass framelet coefficients.
- We also consider the scale-dependent selection threshold with demarcation point *Donoho (1995).* : $\lambda = \sigma\sqrt{2\log(N)}/\sqrt{N}$ for N coefficients.
- The hyperparameter σ is an analogue to the noise level of the wavelet denoising model. Here we let σ be associated with the magnitude order of the coefficients so it reflects the scale of the framelet representation.

Framelet convoution on Cora



- Coefficients at initialization, after 2 convolutional layers before shrinkage activation, and after shrinkage activation. (Comp Ratio 47.7%)
- Horizontal comparison indicates that high-pass coefficients compressed a critical part of coefficients after shrinkage (in green).
- Vertical comparison shows high-pass coefficients usually have more distinctive values and concentrated on the detailed information compared to low-pass.

Node classification benchmark

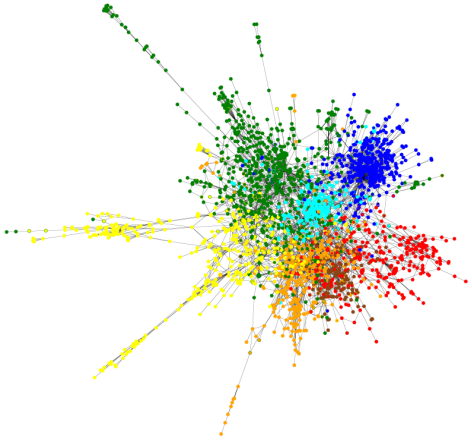
The node classification tasks for GNN are conducted on Cora, Citeseer and Pubmed, which are three benchmark citation networks. Moreover, we employ ogbn-arxiv from open graph benchmark OGB to illustrate the power of our framelet convolution on large-scale graph-structured data.

Table 5. Summary of the datasets for node classification tasks.

	Cora	**Citeseer**	**Pubmed**	**ogbn-arxiv**
# Nodes	2, 708	3, 327	19, 717	169, 343
# Edges	5, 429	4, 732	44, 338	1, 166, 243
# Features	1, 433	3, 703	500	128
# Classes	7	6	3	40
# Training Nodes	140	120	60	90, 941
# Validation Nodes	500	500	500	29, 799
# Test Nodes	1, 000	1, 000	1, 000	48, 603
Label Rate	0.052	0.036	0.003	0.537

- The UFGConv-R and UFGConv-S are compared against **MLP**, **DeepWalk**, **Chebyshev**, **GCN**, **Spectral CNN**, **GWNN**, **MPNN**, **GraphSAGE**, **LanczosNet**, **DCNN**, **GAT**.

Framelet convolution for Node classifcation

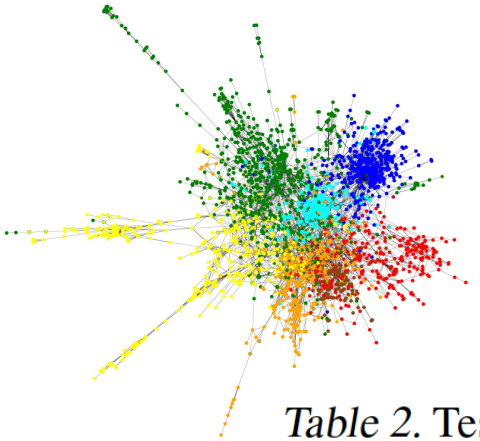


We report the accuracy score in percentage with the top-3 highlighted in the following Table 2. For UFGConv-S, we also report the compression ratio for shrinkage (in green). The UFGConv-R method achieves the highest prediction accuracy among all baseline models. The learned UFGConv-S with threshold level $\sigma = 1$ trims up to 50% information but still obtains the top-3 rank on two tasks.

Method	Cora	Citeseer	Pubmed
MLP	55.1	46.5	71.4
DeepWalk	67.2	43.2	65.3
Spectral	73.3	58.9	73.9
Chebyshev	81.2	69.8	74.4
GCN	81.5	70.3	79.0
GWNN	82.8	**71.7**±0.6	**79.1**
GAT	**83.0**±0.7	**72.5**±0.7	79.0±0.3
MPNN	78.0±1.1	64.0±1.9	75.6±1.0
GraphSAGE	74.5±0.8	67.2±1.0	76.8±0.6
LanczosNet	79.5±1.8	66.2±1.9	78.3±0.3
DCNN	79.7±0.8	69.4±1.3	76.8±0.8
UFGConv-S	**83.0**±0.5	71.0	**79.4**±0.4
(Compression)	(**47.7**)	(**39.0**)	(**27.7**)
UFGConv-R	**83.6**±0.6	**72.7**±0.6	**79.9**±0.1

† The top three are highlighted by **First**, **Second**, **Third**.

Framelet convolution for ogbn-arXiv (169K nodes, 1.1m edges)



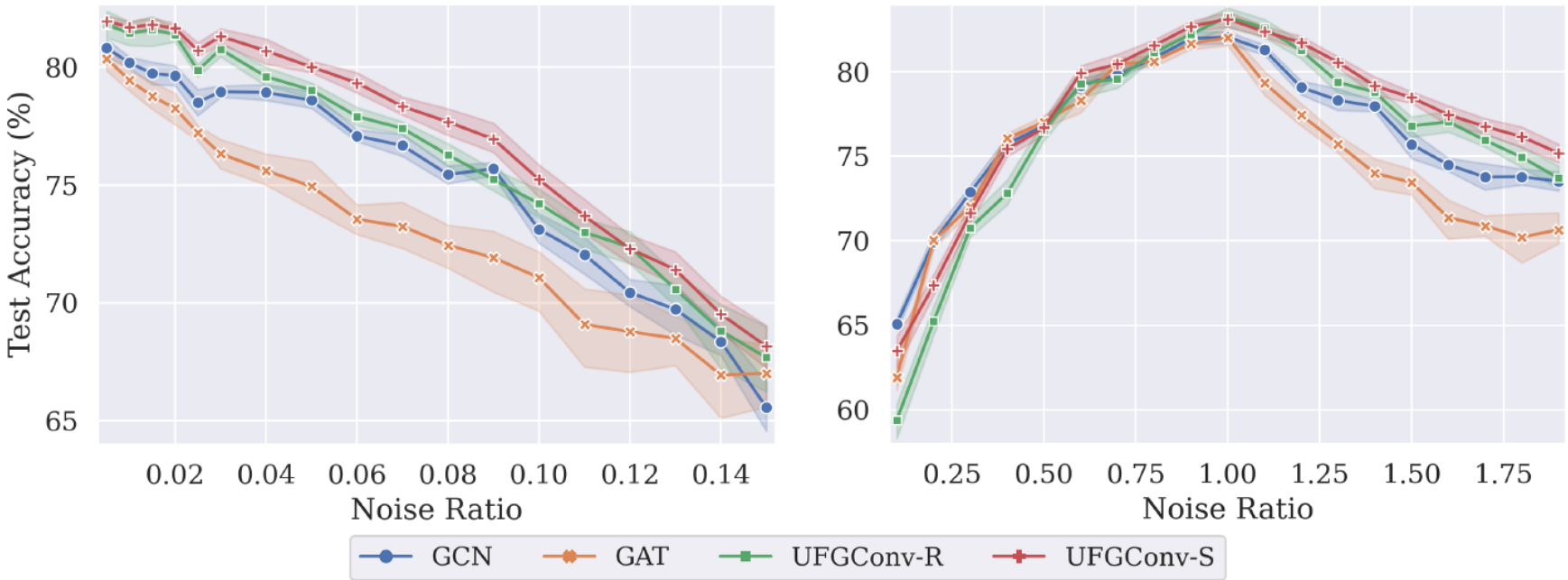
Framelet convolution achieves similar outstanding performance as reported in Table 2 for the ogbn-arxiv dataset, where the UFGConv-R ranks first with a moderate number of parameters, and the UFGConv-S with threshold $\sigma = 1$ achieves a comparably excellent accuracy using 64.2% information.

Table 2. Test accuracy (in percentage) for **ogbn-arxiv** with standard deviation after ±. The compression ratio for UFGConv-S with shrinkage threshold level $\sigma = 1$ is **64.2%**.

Method	Test Acc.	Val. Acc.	#Params
MLP	55.50±0.23	57.65±0.12	110,120
Node2vec	70.07±0.13	71.29±0.13	21,818,792
GraphZoom	71.18±0.18	72.20±0.07	8,963,624
P&L + C&S	71.26±0.01	73.00±0.01	5,160
GraphSAGE	71.49±0.27	72.77±0.17	218,664
GCN	71.74±0.29	73.00±0.17	142,888
DeeperGCN	71.92±0.17	72.62±0.14	491,176
SIGN	**71.95**±0.11	73.23±0.06	3,566,128
GaAN	**71.97**±0.18	–	1,471,506
UFGConv-S	70.04±0.22	71.04±0.11	1,633,183
UFGConv-R	**71.97**±0.12	73.21±0.05	1,633,183

† The top three are highlighted by **First**, **Second**, **Third**.

Robustness of framelet convolution under noise

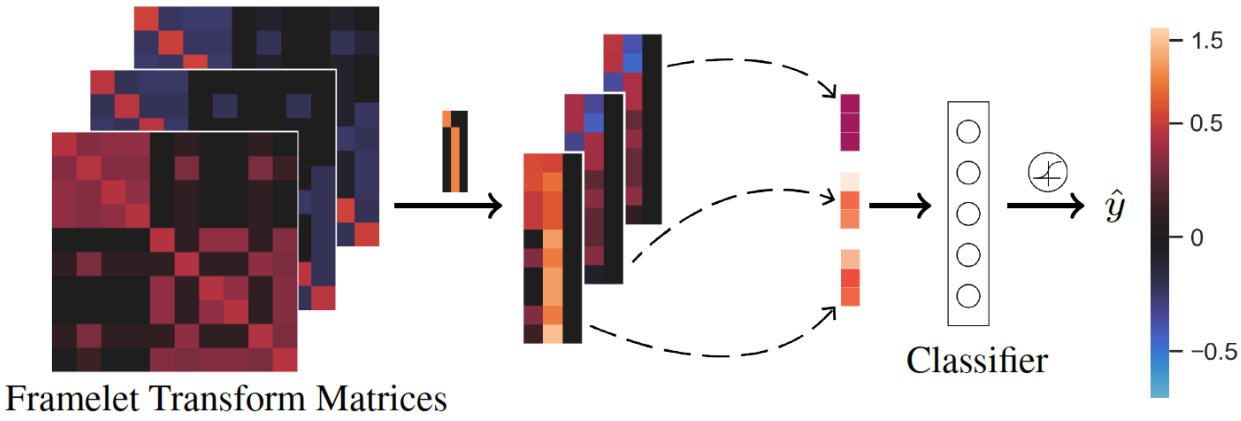


The left and right pictures show node and structure perturbation analysis on **Cora** respectively. We randomly change node feature or edge weight from 0/1 to its opposite 1/0. Figure reports the test accuracy of UFGCONV, GCN and GAT. The x-axis is the distortion ratio (or SNR). As the SNR increases, UFGCONV behaves well ahead of GCN and GAT while with higher test accuracy and smaller variance. The slightly lower performance of UFGCONV only occurs when misinformation dominates the graph structure. It illustrates the effectiveness of shrinkage framelet convolution node and structure denoising.

How GNNs handle input graphs with varying number of nodes and connectivity structures?

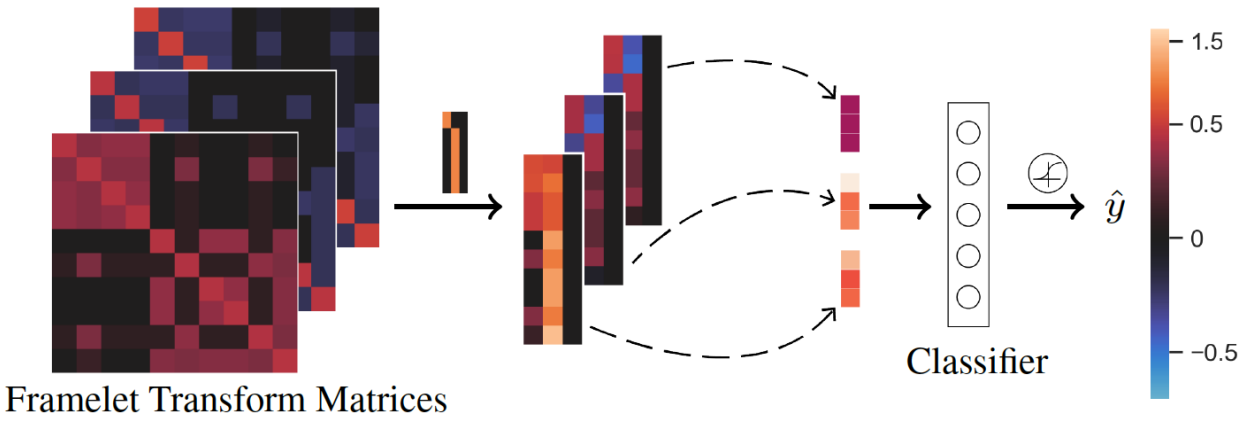
- One way is to use graph pooling. It is a computational strategy to reduce the number of graph nodes while preserve as much as geometric information of the original input graph data.
- By pooling, one has a unified graph-level representation for graph-structured data when the size and topology of an individual graph are changing.

Framelet pooling



- We use 2-level framelet system. Given a graph with feature matrix $X \in \mathbb{R}^{N\times d}$, we can obtain a set of framelet coefficients $\mathcal{W}_{r,j}f$ including one low pass $\mathcal{W}_{0,2}f$ at level 2 and two high passes $\mathcal{W}_{1,1}f$ and $\mathcal{W}_{1,2}f$ at levels 1 and 2, respectively.
- Each scale-wise framelet coefficient is an $N\times d$ real-valued matrix, and its ith feature column $(\mathcal{W}_{r,j}X)_i$ for $i = 1,\ldots,d$ would be aggregated by sum, or sum of squares of the elements.
- The two aggregation methods correspond to two *framelet pooling* strategies as shown in the picture. The calculation compresses the $N \times d$ coefficients to a d-dimensional vector, and the pooled output from the three framelet coefficients results in $3D$ vectors.

Energy conservation in framelet pooling



- The framelet pooling benefits the network training by employing the information from multi-scales, when all scales in the framelet representation of the graph signal are taken into account. Depending on how we aggregate the framelet coefficients, we distinguish the strategies by UFGPOOL-SUM or UFGPOOL-SPECTRUM.
- The latter aggregates the nodes by the wavelet (power) spectrum (i.e., the sum of absolute squares of framelet coefficients over nodes, $\sum_{p\in V} |v_{j,p}|^2$ and $\sum_{p\in V} |w^r_{j,p}|^2$). The total information of the graph signal X^{in} is then well-conserved after the pooling. The sum of wavelet power spectrum is equal to the total energy of the signal: $\|X^{\text{pooled}}\| = \|X^{\text{in}}\|$.

Framelet pooling for graph classification

We select six benchmarks to test the proposed pooling strategies, including four graph classification tasks with moderate sample sizes, one regression task, and one large-scale classification task. First five tasks use TUDataset benchmarks, including D&D, PROTEINS to categorize proteins into enzyme and non-enzyme structures; NCI1 to identify chemical compounds that block lung cancer cells; Mutagenicity to recognize mutagenic molecular compounds for potentially marketable drug; and QM7 to predict atomization energy value of molecules. The dataset ogbg-molhiv is for large-scale molecule classification.

Table 6. Summary of the datasets for the graph property prediction tasks.

Datasets	**PROTEINS**	**Mutagenicity**	**D&D**	**NCI1**	**ogbg-molhiv**	**QM7**
# Graphs	1,113	4,337	1,178	4,110	41,127	7,165
Min # Nodes	4	4	30	3	2	4
Max # Nodes	620	417	5,748	111	222	23
Avg # Nodes	39	30	284	30	26	15
Avg # Edges	73	31	716	32	28	123
# Features	3	14	89	37	9	0
# Classes	2	2	2	2	2	1 (R)

- We compare our framelet pooling UFGPool-SUM and UFGPool-Spectrum with six baseline methods that are capable for global pooling to verify the effectiveness of the learned graph representation. The baselines include **TopKPool**, **AttentionPool**, **SAGPool**, and the classic **Sum**, **Mean** and **Max** pooling.

Framelet pooling for Graph classification

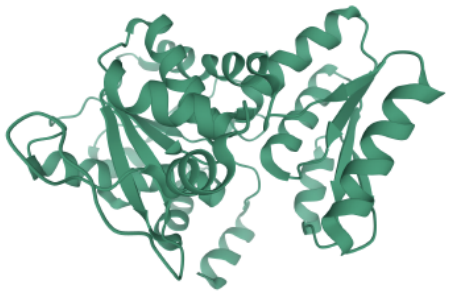


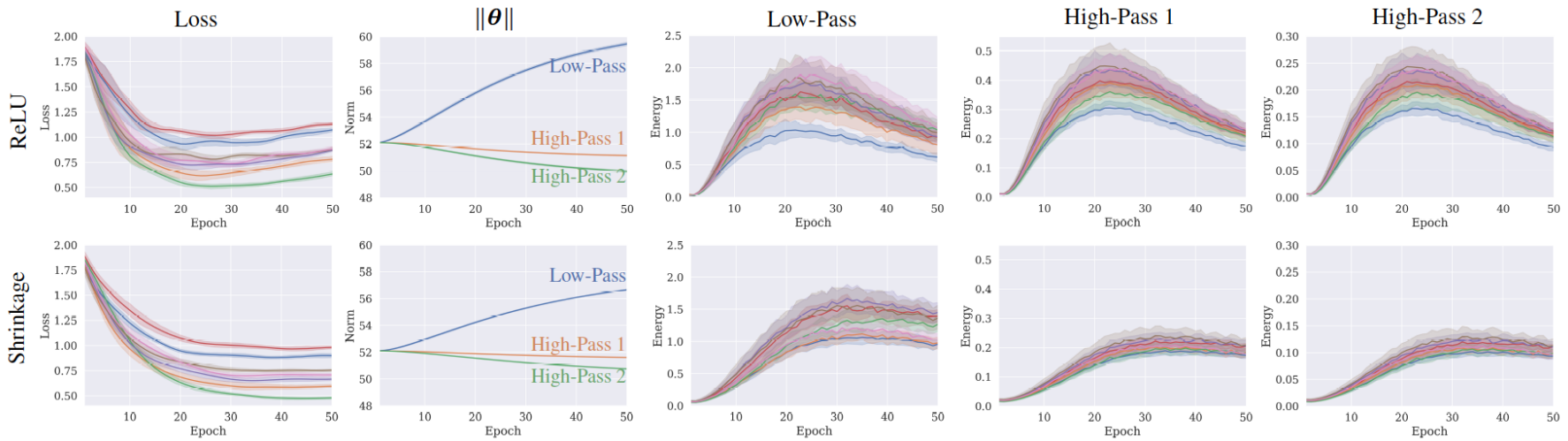
Table 3 shows our UFGPOOL methods outperform other methods on all datasets. Specifically, UFGPOOL-SUM achieves the top accuracy in four out of six datasets, and the second best accuracy in the other two, where the top performance is achieved by UFGPOOL-SPECTRUM. We also observe that UFGPOOL-SPECTRUM performs better on small molecules prediction: **Mutagenicity**, **QM7** and **ogbg-molhiv**. This precedence might come from encoding the multi-scale signal energy to the network where the framelet spectra capture the practically significant features of molecular data.

Table 3. Performance comparison for graph property prediction. **QM7** is a regression task in MSE; **ogbg-molhiv** is a classification task in ROC-AUC in percentage; others are for classification in test accuracy in percentage. The value after ± is standard deviation.

Datasets	PROTEINS	Mutagenicity	D&D	NCI1	ogbg-molhiv	QM7
TOPKPOOL	73.48±3.57	79.84±2.46	74.87±4.12	75.11±3.45	78.14±0.62	175.41±3.16
ATTENTION	73.93±5.37	80.25±2.22	77.48±2.65	74.04±1.27	74.44±2.12	177.99±2.22
SAGPOOL	**75.89**±2.91	79.86±2.36	74.96±3.60	76.30±1.53	75.26±2.29	**41.93**±1.14
SUM	74.91±4.08	**80.69**±3.26	**78.91**±3.37	**76.96**±1.70	77.41±1.16	42.09±0.91
MAX	73.57±3.94	78.83±1.70	75.80±4.11	75.96±1.82	78.16±1.33	177.48±4.70
MEAN	73.13±3.18	80.37±2.44	76.89±2.23	73.70±2.55	**78.21**±0.90	177.49±4.69
UFGPOOL-SUM	**77.77**±2.60	**81.59**±1.40	**80.92**±1.68	**77.88**±1.24	**78.80**±0.56	**41.74**±0.84
UFGPOOL-SPECTRUM	**77.23**±2.40	**82.05**±1.28	**79.83**±1.88	**77.54**±2.24	**78.36**±0.77	**41.67**±0.95

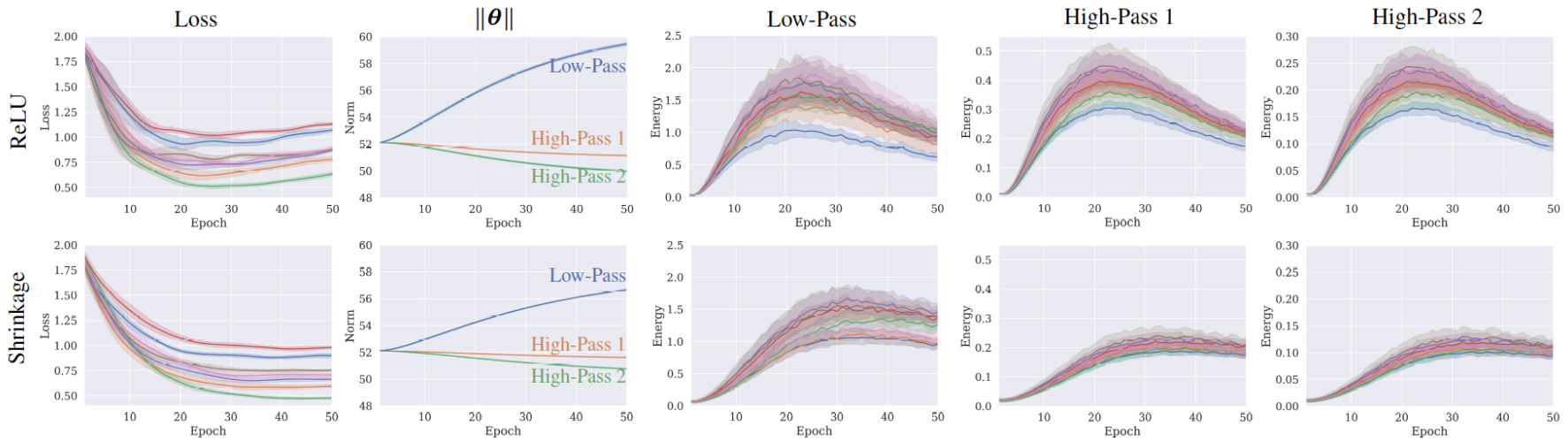
† The top three are highlighted by **First**, **Second**, **Third**.

Framelet spectrum, traning loss and network capacity



- Learning behavior of the final framelet convolutional layer of GNN with two UFG-CONV for **Cora**.
- The coefficients after shrinkage activation are proportional to the framelet power spectrum at the coefficient scale. We thus let the threshold level σ proportionate upon the framelet energy $\|\mathcal{W}_{r,j}X\|^2$ $(r > 0)$ for high-passes. For example, the framelet spectrum curves in training show a higher magnitude order of the low-pass (column 3) than those of high-passes (columns 4-5). This is because coefficients in high-passes reflect more detailed characteristics than in low-pass.
- Compared with the ReLU case (row 1), the shrinkage activation (row 2) filters out some high-pass coefficients in graph convolution, which results in much smaller framelet spectra for high-passes. In contrast, the low-pass shrinkage involves no cutoff, and the energy is less distinguishable from the ReLU case.

Framelet spectrum, traning loss and network capacity



- The training loss curve of each output feature (column 1) indicates that shrinkage allows for more stable training, with a monotonically decreasing loss.
- The splitting in low-pass and high-passes for loss suggests a more flexible and precise control of the training. It also opens the possibility of designing a new weighted loss taking account of framelet spectrum.

Discussion

- We explore the adaptation of graph framelets for graph neural networks, and link graph neural networks and signal processing. In many node-level or graph-level tasks, framelet convolutions can reduce both feature and structure noises.
- We introduce shrinkage activation that thresholds high-pass coefficients in framelet convolution, which strengthens the network denoising capability and simultaneously compresses graph signal at a remarkable rate.
- The framelet pooling outperforms baselines on a variety of graph property prediction tasks.

Thank you!

References

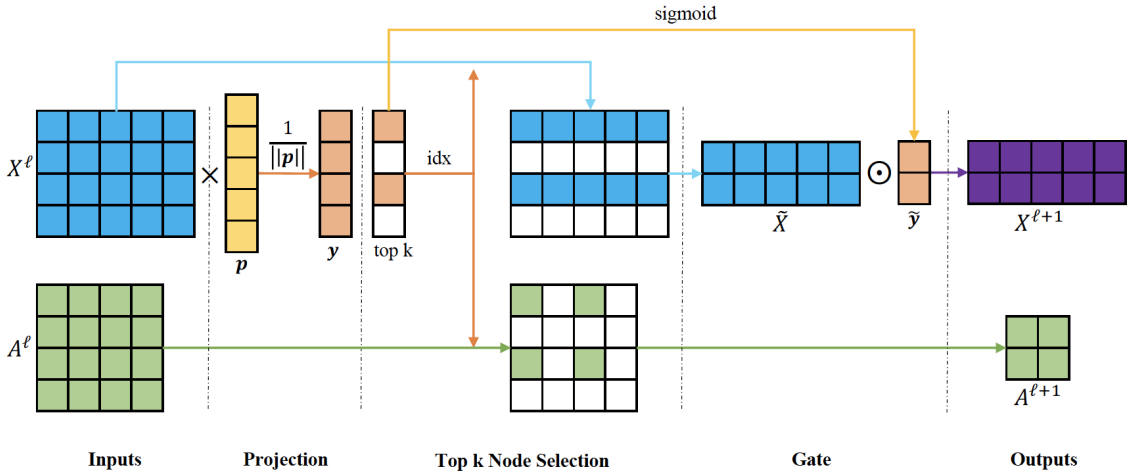
- How framelets Enhance Graph Neural Networks. *ICML 2021*
- Decimated Framelet System on Graphs and Fast $\mathcal{G}$-Framelet Transforms. *JMLR 2021*
- Tight Framelets and Fast Framelet Filter Bank Transforms on Manifolds. *ACHA 2020*
- Haar Graph Pooling. *ICML 2020*
- Fast Haar Transforms for Graph Neural Networks. *Neural Networks 2020*

Codes YuGuangWang@Github

Methods and Architecture in Comparison

- In the experiment, we normalize the label value by subtracting the mean and scaling the standard deviation (Std Dev) to 1. We then need to convert the predicted output to the original label domain (by re-scaling and adding the mean back).
- Following *Gilmer et al. (2017).* , we use mean squared error (MSE) as the loss for training and mean absolute error (MAE) as the evaluation metric for validation and test.
- Here, the splitting percentages for training, validation, and test are 80%, 10%, and 10%, respectively. We set the hidden dimension of the GCN layer as 64, the Adam for optimization with the learning rate 5.0e-4, and the maximal epoch 50 with no early stop. We do not use dropout as it would slightly lower the performance. For better comparison, we repeat all experiments ten times with different random seeds.
- We use the same GNN architecture to test HaarPool and SAGPool *Lee et al., (2019).* : one GCN layer, one graph pooling layer, plus one 3-layer MLP.
- We compare the performance (test MAE) of the GCN-HaarPool against the GCN-SAGPool and other methods including Random Forest **RF** *Breiman, (2001).* , Multitask Networks **Multitask** *Ramsundar et al., (2015).* , Kernel Ridge Regression **KRR** *Cortes & Vapnik, (1995).* , Graph Convolutional models **GC** *Altae-Tran et al., (2017).* .

Existing Graph Pooling Methods



TopKPooling *Gao & Ji 2019, Knyazev et al 2019, Cangea, Liò et al 2018.*

- Spatial Method
  - Global (Non-hierarchical) pooling:

    ChebNet, Set2Set, SortPool, MPNN
  - Hierarchical pooling:

    DiffPool, TopKPool, SAGPool, EdgePool
- Spectral Method
  - LaPool, EigenPool, SOPool