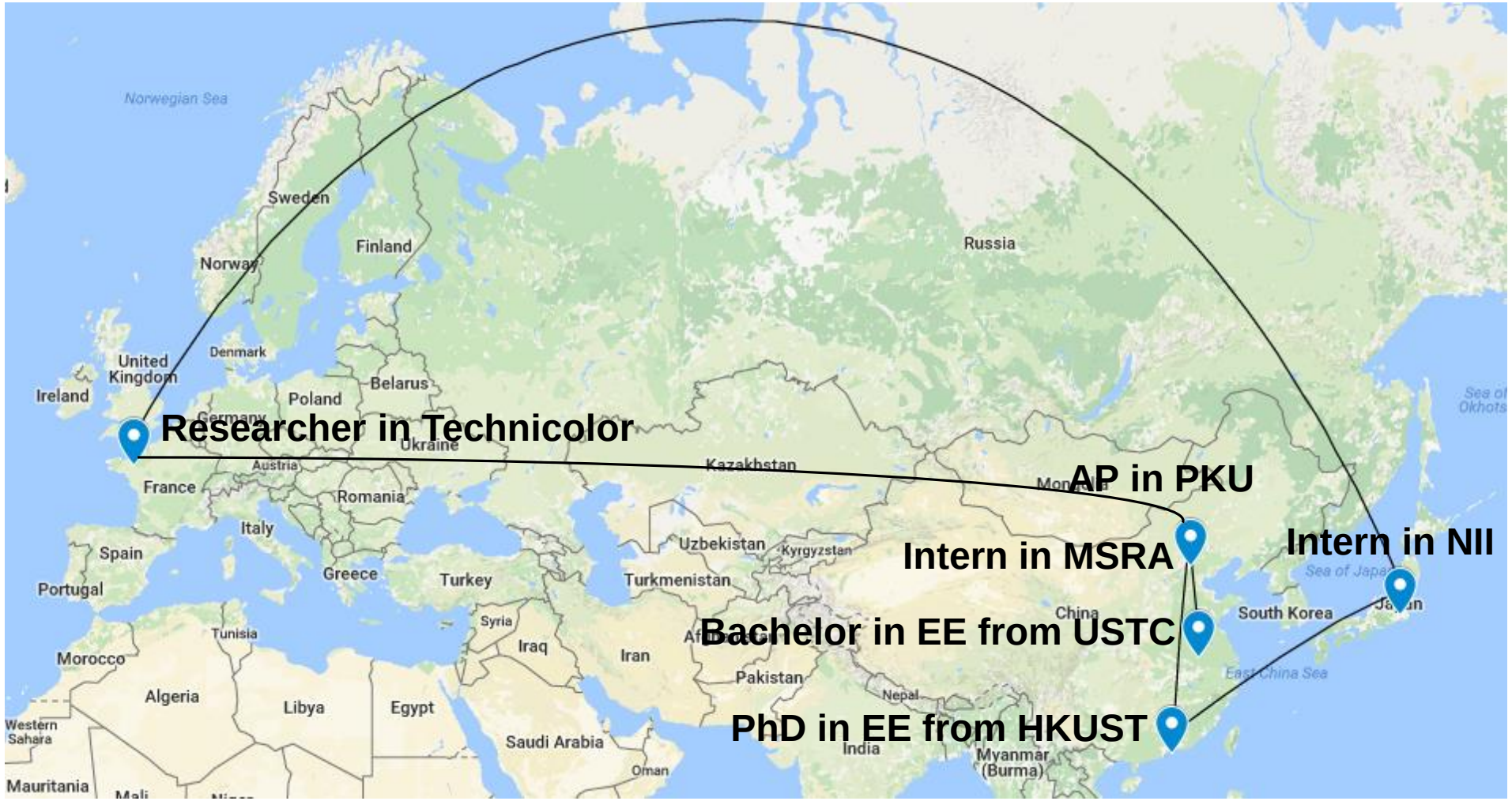
北京大学

Graph Convolutional Neural Networks: from perspective of Graph Signal Processing

Wei Hu

2019.7

About Me


Researcher in Technicolor
AP in PKU
Intern in MSRA
Intern in NII
Bachelor in EE from USTC
PhD in EE from HKUST
Norwegian Sea
Sweden
Finland
Norway
Russia
United Kingdom
Ireland
Denmark
Belarus
Poland
Germany
Ukraine
Kazakhstan
France
Austria
Romania
Italy
Spain
Portugal
Greece
Turkey
Uzbekistan
Kyrgyzstan
Turkmenistan
China
South Korea
Sea of Japan
Syria
Iraq
Iran
Tunisia
Morocco
Algeria
Libya
Egypt
Pakistan
Nepal
India
Myanmar (Burma)
Saudi Arabia
Oman
Western Sahara
Mauritania
East China Sea

Acknowledgement

Main Collaborators:

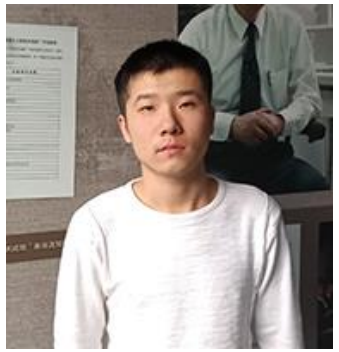
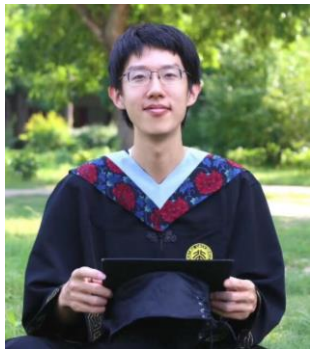
- Gene Cheung (York University, Canada)
- Antonio Ortega (USC, USA)
- Chia-Wen Lin (National Tsing Hua University, Taiwan)
- Xin Li (WVU, USA)
- Xiang Gao, Gusi Te (PKU, China)

Partial contents are courtesy of

Bronstein's tutorial in Graph Signal Processing Workshop, 2018





Outline

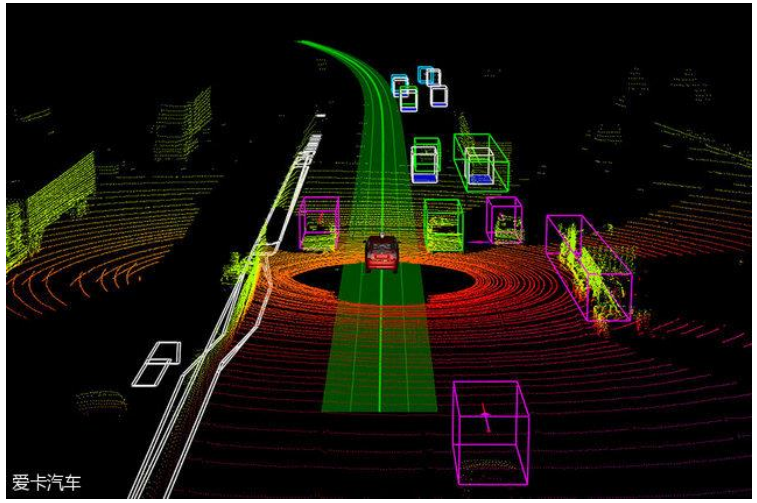
- Introduction to Graph Convolutional Neural Networks (Graph CNN)
- Background in Graph Signal Processing for analysis on graphs
- Spectral Graph CNNs
- Spatial Graph CNNs
- Applications, Challenges, Open problems

Motivation

Graphs are flexible mathematical structures modeling pair-wise relations between data entities.

(a) Brain network

(b) Social network

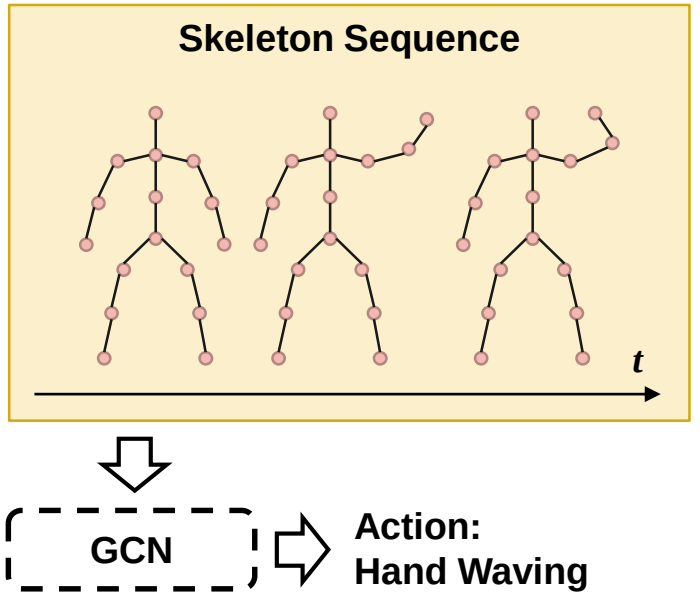


(c) 3D point clouds

Deep Learning on Graphs?

Application example: graph-wise classification

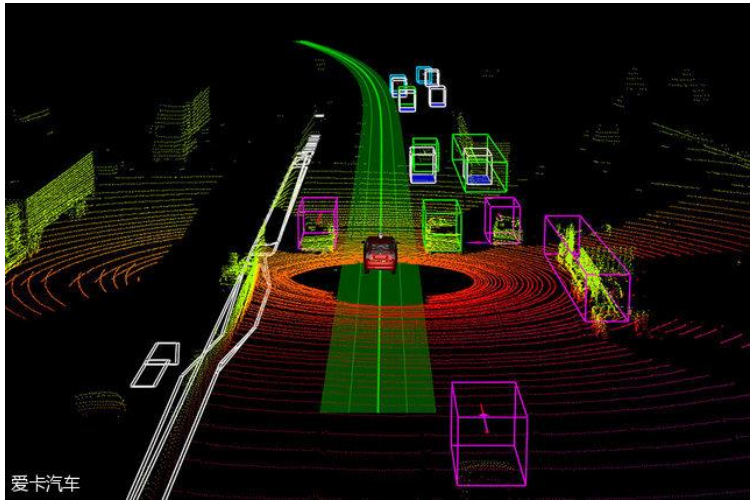
Point cloud classification



Action recognition

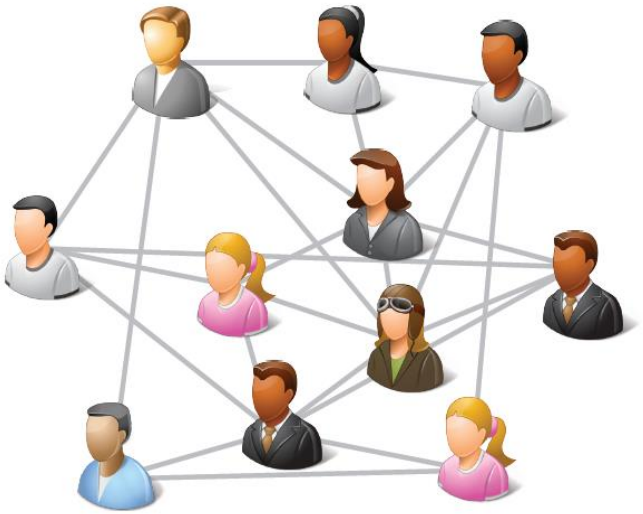
Application example: vertex-wise classification

Social network classification

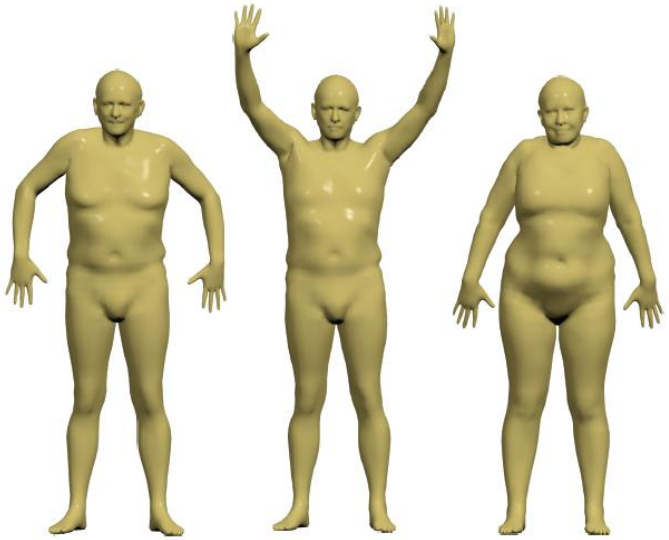


Detection in
autonomous driving

Fixed v.s. different domain



Social network
(fixed graph)



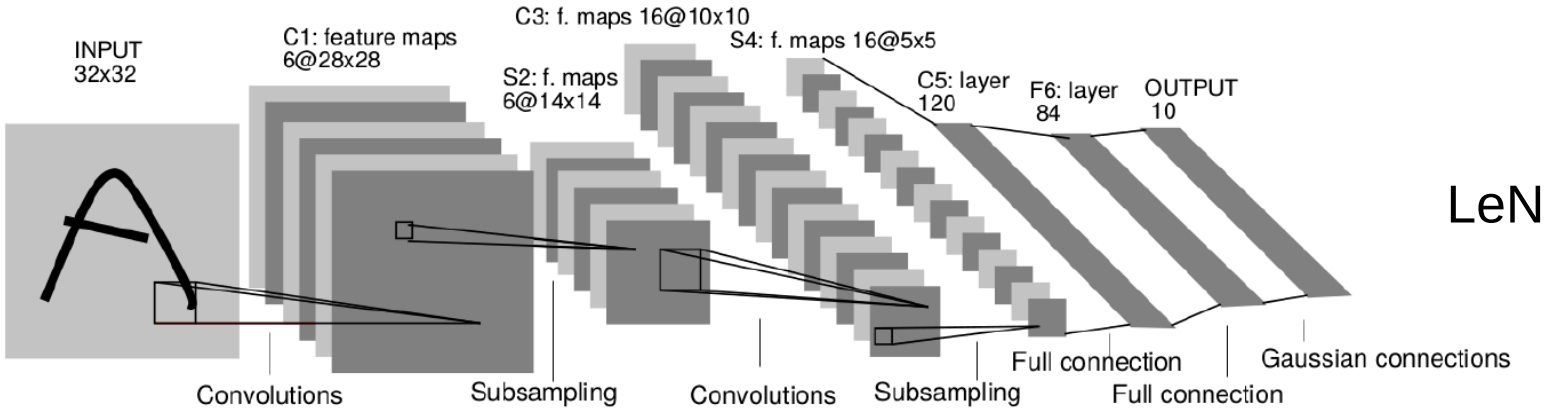
3D shapes
(different graphs)

Known v.s. unknown domain

Given graph

Learned graph

Deep neural networks: key ingredients and properties

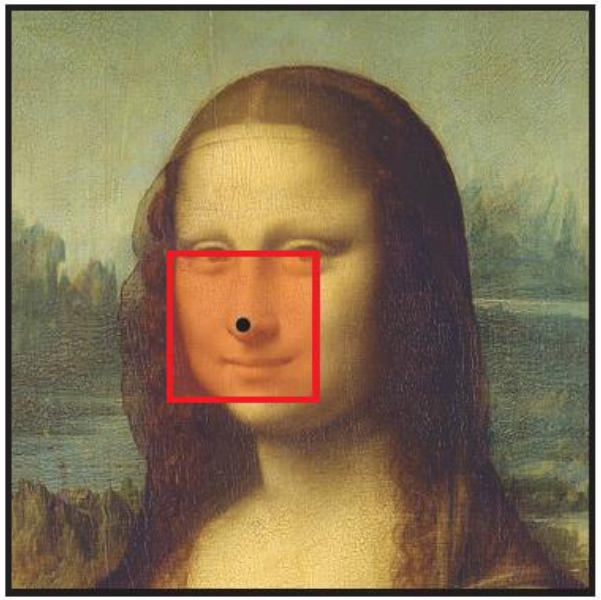


LeNet-5 CNN architecture

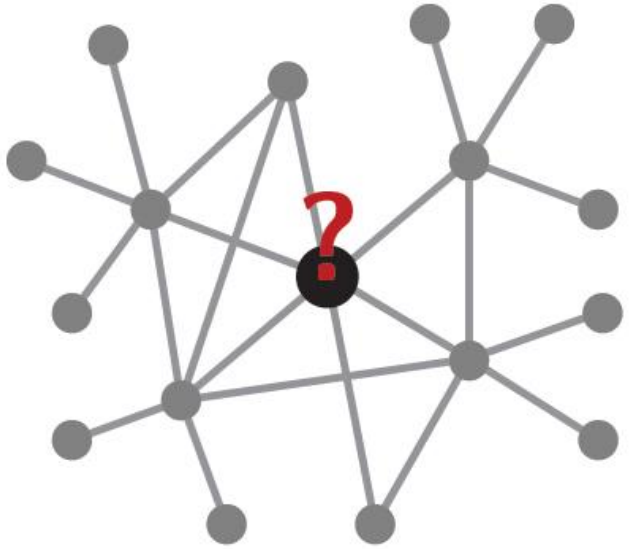
- ☺ Convolutional filters **(Translation invariance+Self-similarity)**
- ☺ Multiple layers **(Compositionality)**
- ☺ Filters localized in space **(Locality)**
- ☺ $\mathcal{O}(1)$ parameters per filter (independent of input image size n)
- ☺ $\mathcal{O}(n)$ complexity per layer (filtering done in the spatial domain)

LeCun et al. 1989

CNN on graphs?



Image



Graph

Outline

- Introduction to Graph Convolutional Neural Networks (Graph CNN)
- Background in Graph Signal Processing for analysis on graphs
- Spectral Graph CNNs
- Spatial Graph CNNs
- Applications, Challenges, Open problems

Graph Signal Processing in a Nutshell

- Graph: fixed or learned from data
- Signal: set of scalars / vectors associated to graph vertices
- Define notions: frequency, sampling, transforms, etc.
- Use these to solve problems such as compression, reconstruction, frequency analysis, spectral clustering, etc.

Graph Signal Processing in a Nutshell

- Overview papers: [1-3]
- Many research contributions (3 GSP workshops)
- Toolbox: GSPbox, GraSP, PyGSP
- Three textbooks coming up

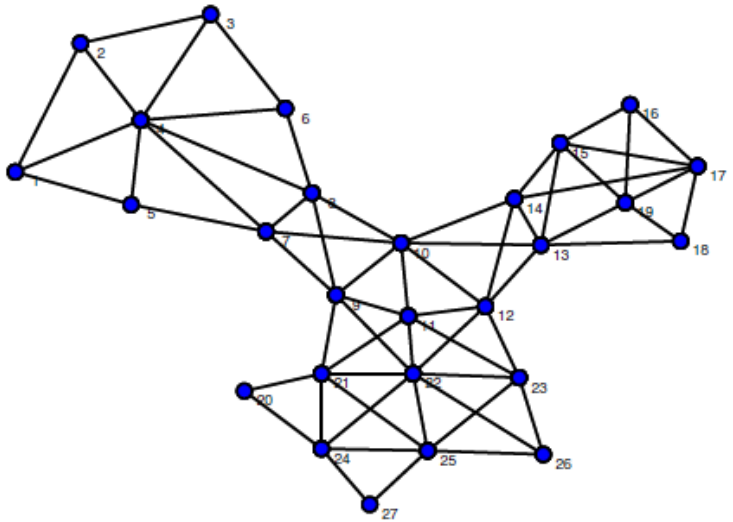
[1] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,” *IEEE Signal Processing Magazine*, vol. 30, pp. 83–98, May 2013.

[2] A. Sandryhaila and J.M. Moura, “Discrete signal processing on graphs,” *IEEE transactions on signal processing*, 61(7), pp.1644-1656, 2013.

[3] A. Ortega, P. Frossard, J. Kovačević, J.M. Moura, and P. Vandergheynst, “Graph signal processing: Overview, challenges, and applications,” Proceedings of the IEEE, 106(5), pp.808-828, 2018.

Basic Definitions

- Graph: vertices (nodes) connected via some edges (links)
- Graph Signal: set of scalar/vector values defined on the vertices.



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

Vertex Set $\mathcal{V} = \{v_1, v_2, ...\}$

Edge Set $\mathbf{E} = \{(v_1, v_2), (v_1, v_3), ...\}$

Weighted edges w, sets of weights a_{ij}

Graph Signal $\mathbf{x} = \{x_1, x_2, ...\}$

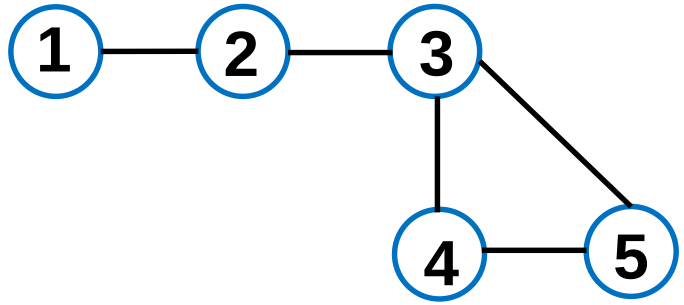
Neighborhood, h-hop

$\mathcal{N}_h(i) = \{j \in \mathcal{V} : hop\_dist(i, j) \leq h\}$

Allow to define locality on the graph

Algebraic representations

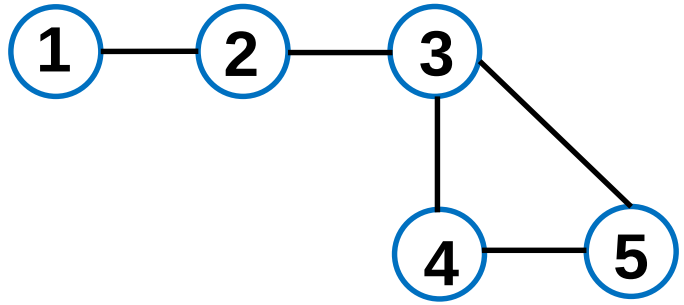
- Adjacency matrix: $\mathbf{A}$
  - $a_{i,j}$: edge weight for the edge (v_i, v_j)
  - Describe the similarity / correlation between nodes
  - Undirected graph: $a_{i,j} = a_{j,i}$



Algebraic representations

- Adjacency matrix: $\mathbf{A}$
  - $a_{i,j}$: edge weight for the edge (v_i, v_j)
  - Describe the similarity / correlation between nodes
  - Undirected graph: $a_{i,j} = a_{j,i}$
- Degree matrix: $\mathbf{D}$

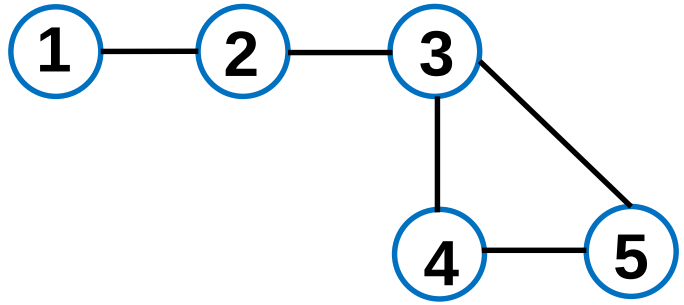
$$d_{i,i} = \sum_{j=1}^{N} a_{i,j}$$



Algebraic representations

- Combinatorial Graph Laplacian matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

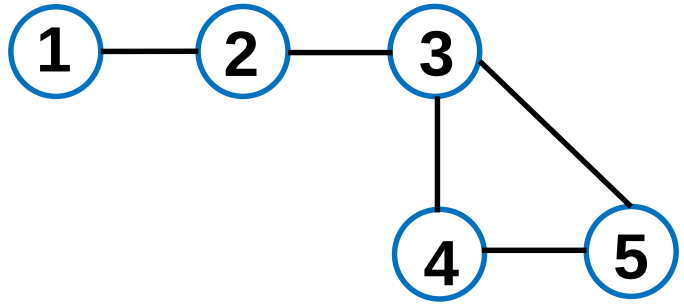


Algebraic representations

- Combinatorial Graph Laplacian matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

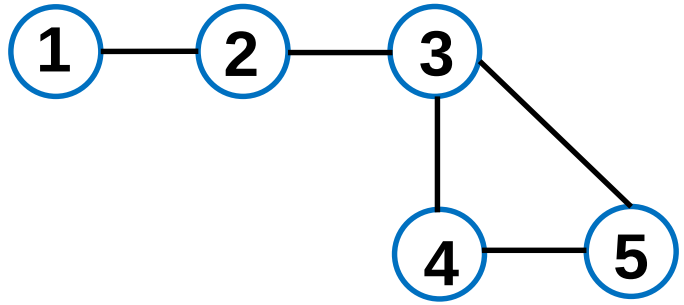
  - $\mathbf{L}$ is symmetric



Algebraic representations

- Combinatorial Graph Laplacian matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$



  - $\mathbf{L}$ is symmetric

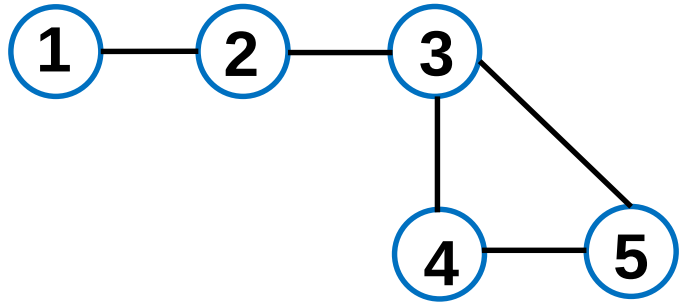
  - When operating $\mathbf{L}$ on a graph signal $\mathbf{x}$, it captures the variation in the signal

$$(\mathbf{L}\mathbf{x})(i) = \sum_{j \in \mathcal{N}_i} a_{i,j}(x_i - x_j)$$

Algebraic representations

- Combinatorial Graph Laplacian matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$



  - $\mathbf{L}$ is symmetric

  - When operating $\mathbf{L}$ on a graph signal $\mathbf{x}$, it captures the variation in the signal

$$(\mathbf{Lx})(i) = \sum_{j \in \mathcal{N}_i} a_{i,j}(x_i - x_j)$$

  - Total variation

$$\mathbf{x}^T\mathbf{Lx} = \sum_{i \sim j} a_{i,j}(x_i - x_j)^2$$

Graph-signal smoothness prior
--- an important regularizer

Graph Fourier Transform

- The graph Laplacian $\mathbf{L} \in \mathcal{R}^{N \times N}$ is real and symmetric: $\mathbf{L}\psi_l = \lambda_l \psi_l$
  - a set of real eigenvalues $\{\lambda_l\}_{l=0}^{N-1}$
  - a complete set of orthonormal eigenvectors $\{\psi_l\}_{l=0}^{N-1}$

- The eigenvectors $\{\psi_l\}_{l=0}^{N-1}$ define the GFT basis:

$$\mathbf{\Phi} = \begin{bmatrix} | & & | \\ \psi_0 & \cdots & \psi_{N-1} \\ | & & | \end{bmatrix}$$

Graph Fourier Transform

- For any signal $\mathbf{x} \in \mathcal{R}^N$ residing on the nodes of $\mathcal{G}$, its GFT $\hat{\mathbf{x}} \in \mathcal{R}^N$ is defined as

$$\hat{\mathbf{x}}(l) = < \psi_l, \mathbf{x} >, l = 0, 1, ..., N-1 \qquad (\hat{\mathbf{x}} = \mathbf{\Phi}^T \mathbf{x})$$

GFT coefficients　GFT basis　graph signal

Graph Fourier Transform

- For any signal $\mathbf{x} \in \mathcal{R}^N$ residing on the nodes of $\mathcal{G}$, its GFT $\hat{\mathbf{x}} \in \mathcal{R}^N$ is defined as

$$\hat{\mathbf{x}}(l) =< \psi_l, \mathbf{x} >, l = 0, 1, ..., N-1 \qquad (\hat{\mathbf{x}} = \mathbf{\Phi}^T \mathbf{x})$$

GFT coefficients　GFT basis　graph signal

- The inverse GFT follows as

$$\mathbf{x}(n) = \sum_{l=0}^{N-1} \hat{\mathbf{x}}(l)\psi_l(n), n = 0, 1, ..., N-1 \qquad (\mathbf{x} = \mathbf{\Phi}\hat{\mathbf{x}})$$

Why Graph Fourier Transform

- Offer compact transform domain representation

$\Longrightarrow$ GFT $\alpha_1 = \begin{bmatrix} 237 & 0 & 0 & 0 \\ 163 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ DCT $\alpha_2 = \begin{bmatrix} 285 & -29 & -5 & -4 \\ 16 & 1 & -16 & -4 \\ -5 & 3 & 5 & -7 \\ -1 & -4 & 1 & 9 \end{bmatrix}$

- Reason: the graph adaptively captures the correlation in the graph signal

- $\approx$ KLT for a family of statistical models

Wei Hu, Gene Cheung, Antonio Ortega, Oscar C. Au, "Multi-resolution Graph Fourier Transform for Compression of Piecewise Smooth Images," *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 419-433, January 2015.

Outline

- Introduction to Graph Convolutional Neural Networks (Graph CNN)
- Background in Graph Signal Processing for analysis on graphs
- Spectral Graph CNNs
- Spatial Graph CNNs
- Applications, Challenges, Open problems

Convolution

Euclidean

Spatial domain

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(x')g(x-x')dx'$$

Spectral domain

$$\widehat{(f \star g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$$

'Convolution Theorem'

Non-Euclidean

?

?

Convolution

Euclidean

Spatial domain

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(x')g(x-x')dx'$$

Spectral domain

$$\widehat{(f \star g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$$

'Convolution Theorem'

Non-Euclidean

?

$$\widehat{\mathbf{f} \star \mathbf{g}} = (\mathbf{\Phi}^\top \mathbf{g}) \circ (\mathbf{\Phi}^\top \mathbf{f})$$

Principle of Spectral Graph CNNs

- According to the Convolution Theorem, the graph convolution of signal $\mathbf{f}$ and filter $\mathbf{g}$ is

$$\mathbf{g} *_{\mathcal{G}} \mathbf{f} = \Phi(\Phi^{\top}\mathbf{g} \odot \Phi^{\top}\mathbf{f})$$

Φ is the matrix of eigenvectors

$\odot$ denotes the Hadamard product

Principle of Spectral Graph CNNs

- According to the Convolution Theorem, the graph convolution of signal $\mathbf{f}$ and filter $\mathbf{g}$ is

$$\mathbf{g} *_{\mathcal{G}} \mathbf{f} = \Phi(\Phi^{\top}\mathbf{g} \odot \Phi^{\top}\mathbf{f})$$

- Let $\mathbf{g}_{\theta} = \mathrm{diag}(\Phi^{\top}\mathbf{g})$, the graph convolution is simplified as

$$\mathbf{g}_{\theta} *_{\mathcal{G}} \mathbf{f} = \Phi\mathbf{g}_{\theta}\Phi^{\top}\mathbf{f}$$

Principle of Spectral Graph CNNs

- According to the Convolution Theorem, the graph convolution of signal $\mathbf{f}$ and filter $\mathbf{g}$ is

$$\mathbf{g} *_{\mathcal{G}} \mathbf{f} = \Phi(\Phi^{\top}\mathbf{g} \odot \Phi^{\top}\mathbf{f})$$

- Let $\mathbf{g}_{\theta} = \mathrm{diag}(\Phi^{\top}\mathbf{g})$, the graph convolution is simplified as

$$\mathbf{g}_{\theta} *_{\mathcal{G}} \mathbf{f} = \Phi\mathbf{g}_{\theta}\Phi^{\top}\mathbf{f}$$

The key difference in spectral Graph CNNs is the choice of filter $\mathbf{g}_{\theta}$.

Spectral Graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

$\mathbf{g}_\theta = \mathbf{W}$ is a learnable diagonal matrix.

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014.

Spectral graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Spectral graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

☹ Filters are basis-dependent ⇒ does not generalize across graphs

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Spectral graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

☹ Filters are basis-dependent ⇒ does not generalize across graphs

☹ Only undirected graphs (symmetric Laplacian matrix required for orthogonal eigendecomposition)

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Spectral graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

☹ Filters are basis-dependent ⇒ does not generalize across graphs

☹ Only undirected graphs (symmetric Laplacian matrix required for orthogonal eigendecomposition)

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Spectral graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

☹ Filters are basis-dependent ⇒ does not generalize across graphs

☹ Only undirected graphs (symmetric Laplacian matrix required for orthogonal eigendecomposition)

☹ $\mathcal{O}(n)$ parameters per layer

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Spectral graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

☹ Filters are basis-dependent ⇒ does not generalize across graphs

☹ Only undirected graphs (symmetric Laplacian matrix required for orthogonal eigendecomposition)

☹ $\mathcal{O}(n)$ parameters per layer

☹ $\mathcal{O}(n^2)$ computation of forward / inverse Fourier transforms $\mathbf{\Phi}^\top$, $\mathbf{\Phi}$

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Spectral graph CNNs --- Spectral Networks

- The convolution operation is defined in the graph Fourier domain as

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi \mathbf{W} \Phi^\top \mathbf{f}$$

☹ Filters are basis-dependent ⇒ does not generalize across graphs

☹ Only undirected graphs (symmetric Laplacian matrix required for orthogonal eigendecomposition)

☹ $\mathcal{O}(n)$ parameters per layer

☹ $\mathcal{O}(n^2)$ computation of forward / inverse Fourier transforms $\mathbf{\Phi}^\top$, $\mathbf{\Phi}$

☹ No guarantee of spatial localization of filters

J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, “Spectral networks and locally connected networks on graphs,” in *Proceedings of the 3rd International Conference on Learning Representations*, 2014.

Spectral graph CNNs --- Spectral Networks

- Localization in space = smoothness in frequency domain

- Parametrize the filter using a smooth spectral transfer function $\Theta(\lambda)$

$$\Phi\Theta(\mathbf{\Lambda})\Phi^{\top}\mathbf{f} = \Phi \begin{bmatrix} \Theta(\lambda_1) & & \\ & \ddots & \\ & & \Theta(\lambda_n) \end{bmatrix} \Phi^{\top}\mathbf{f}$$

M. Henaff, J. Bruna, and Y. LeCun, “Deep convolutional networks on graph-structured data,” *arXiv preprint arXiv:1506.05163*, 2015.

Spectral graph CNNs --- ChebNet

- Represent spectral transfer function as a polynomial of order K

$$\Theta(\boldsymbol{\Lambda}) = \sum_{k=0}^{K-1} \theta_k \Lambda^k$$

where $\theta_0, \ldots, \theta_{K-1}$ are learnable parameters

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- Represent spectral transfer function as a polynomial of order K

$$\Theta(\mathbf{\Lambda}) = \sum_{k=0}^{K-1} \theta_k \Lambda^k$$

- Using Chebyshev polynomial to approximate above formulation

$$\Theta(\mathbf{\Lambda}) = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{\Lambda}})$$

where $T_0(x) = 1, \quad T_1(x) = x, \quad T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x).$

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- The graph convolution operation becomes

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi\Theta(\mathbf{\Lambda})\Phi^\top\mathbf{f} \approx \Phi \sum_{k=0}^{K-1} \theta_k T_k\left(\tilde{\mathbf{\Lambda}}\right)\Phi^\top\mathbf{f}$$

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- The graph convolution operation becomes

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} = \Phi\Theta(\mathbf{\Lambda})\Phi^\top\mathbf{f} \approx \Phi\sum_{k=0}^{K-1}\theta_k T_k\left(\tilde{\mathbf{\Lambda}}\right)\Phi^\top\mathbf{f}$$

$$= \sum_{k=0}^{K-1}\theta_k T_k\left(\tilde{\mathbf{L}}\right)\mathbf{f}$$

where $\tilde{\mathbf{L}} = \frac{2}{\lambda_{\max}}\mathbf{L} - \boldsymbol{I}_n$, $\lambda_{\max}$ denotes the largest eigenvalue of $\mathbf{L}$, which makes eigenvalues lie in $[-1,1]$.

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- The graph convolution operation of ChebNet

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})\,\mathbf{f}$$

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- The graph convolution operation of ChebNet

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})\,\mathbf{f}$$

😐 Can be applied to directed graphs (spectral interpretation is lost)

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- The graph convolution operation of ChebNet

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})\,\mathbf{f}$$

😐 Can be applied to directed graphs (spectral interpretation is lost)

☺ $\mathcal{O}(1)$ parameters per layer

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- The graph convolution operation of ChebNet

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})\,\mathbf{f}$$

😐 Can be applied to directed graphs (spectral interpretation is lost)

☺ $\mathcal{O}(1)$ parameters per layer

☺ Filters have guaranteed K-hop support

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Spectral graph CNNs --- ChebNet

- The graph convolution operation of ChebNet

$$\mathbf{g}_\theta *_\mathcal{G} \mathbf{f} \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})\,\mathbf{f}$$

😐 Can be applied to directed graphs (spectral interpretation is lost)

☺ $\mathcal{O}(1)$ parameters per layer

☺ Filters have guaranteed K-hop support

☺ No explicit computation of $\mathbf{\Phi}^\top$, $\mathbf{\Phi} \Rightarrow \mathcal{O}(n)$ computational complexity

D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129-150, 2011

M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *NIPS*, pp. 3844–3852, 2016.

Example: citation networks

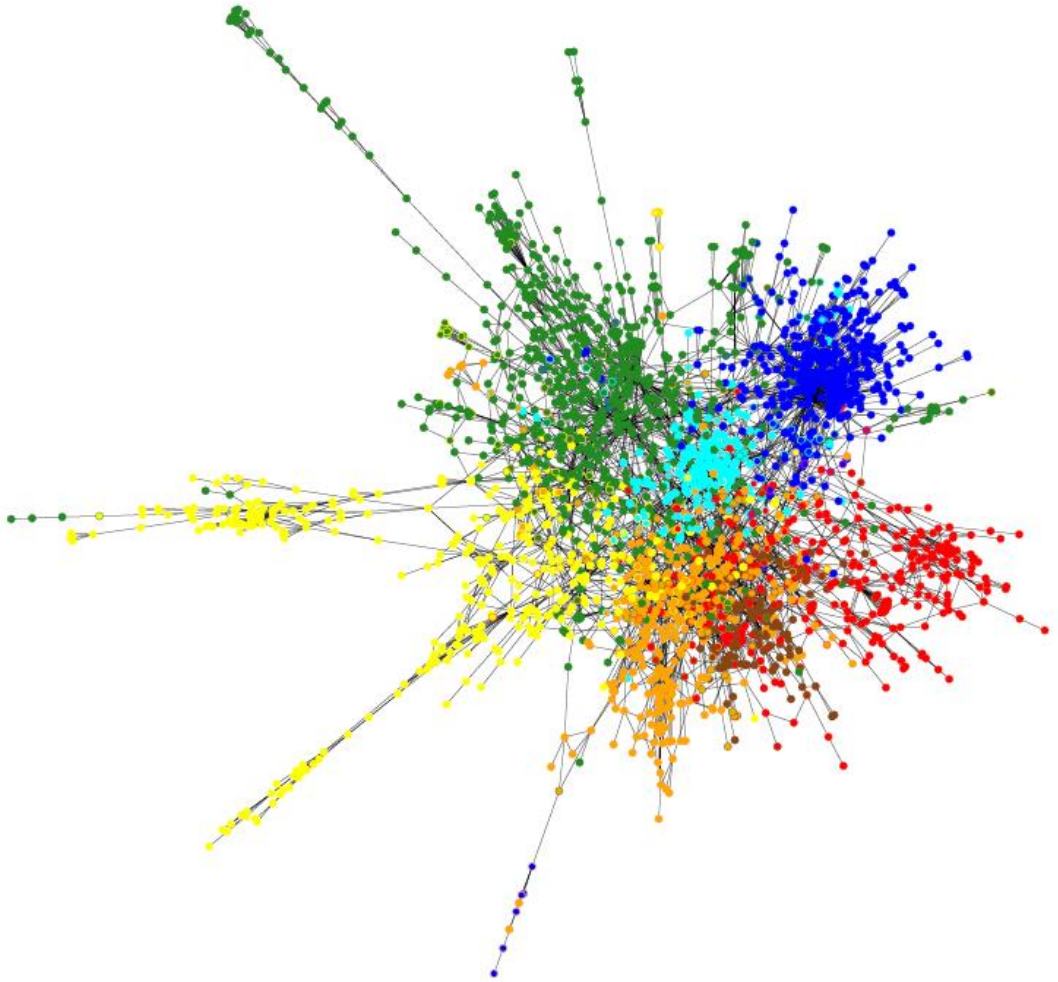


Figure: Monti, Boscaini, Masci, Rodola, Svoboda, B 2017; data: Sen et al. 2008

Example: citation networks

Method	Cora[1]	PubMed[2]
Manifold Regularization[3]	59.5%	70.7%
Semidefinite Embedding[4]	59.0%	71.1%
Label Propagation[5]	68.0%	63.0%
DeepWalk[6]	67.2%	65.3%
Planetoid[7]	75.7%	77.2%
Spectral graph CNN[8]	**81.6%**	**78.7%**

Classification accuracy of different methods on citation network datasets

Data: [1,2]Sen et al. 2008; methods: [3]Belkin et al. 2006; [4]Weston et al. 2012; [5]Zhu et al. 2003; [6]Perozzi et al. 2014; [7]Yang et al. 2016; [8]Kipf, Welling 2016 (simplification of ChebNet)

Outline

- Introduction to Graph Convolutional Neural Networks (Graph CNN)
- Background in Graph Signal Processing for analysis on graphs
- Spectral Graph CNNs
- Spatial Graph CNNs
- Applications, Challenges, Open problems

Convolution

Euclidean

Spatial domain

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(x')g(x-x')dx'$$

Spectral domain

$$\widehat{(f \star g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$$

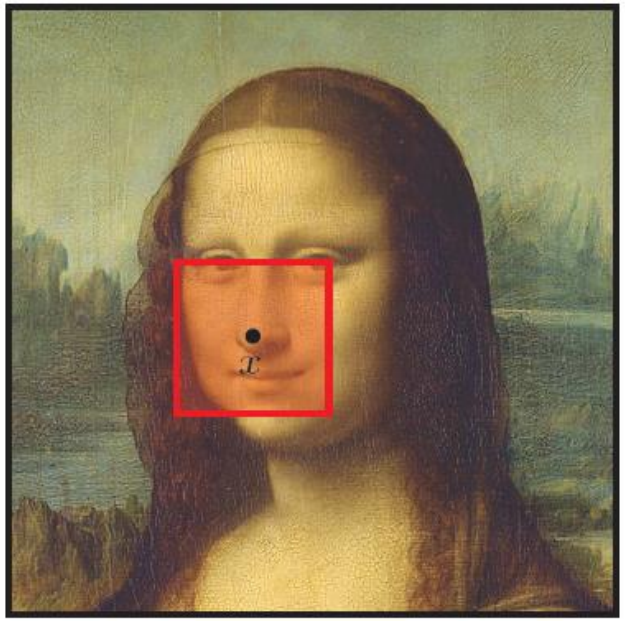
'Convolution Theorem'

Non-Euclidean

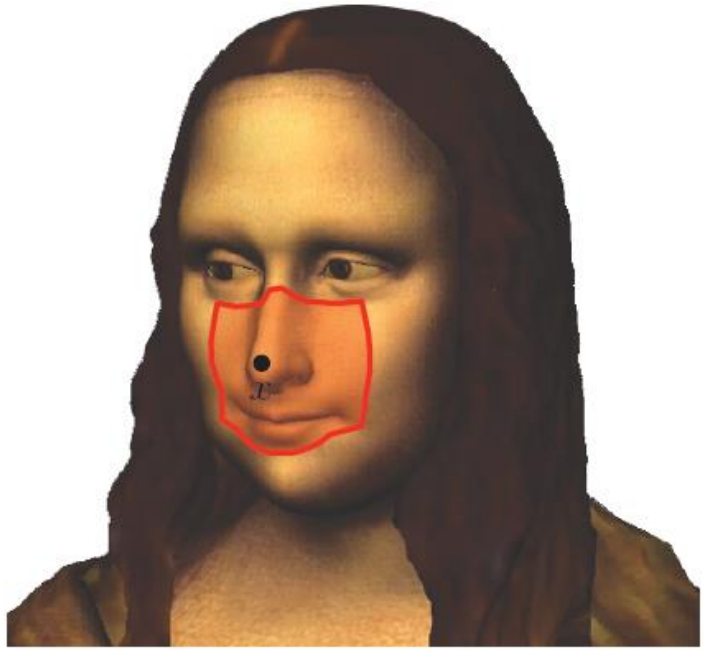
?

$$\widehat{\mathbf{f} \star \mathbf{g}} = (\mathbf{\Phi}^\top \mathbf{g}) \circ (\mathbf{\Phi}^\top \mathbf{f})$$

Spatial convolution



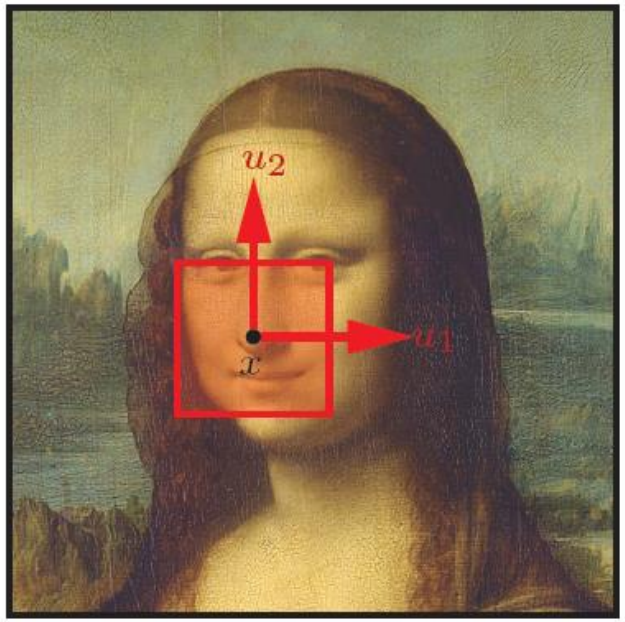
Euclidean



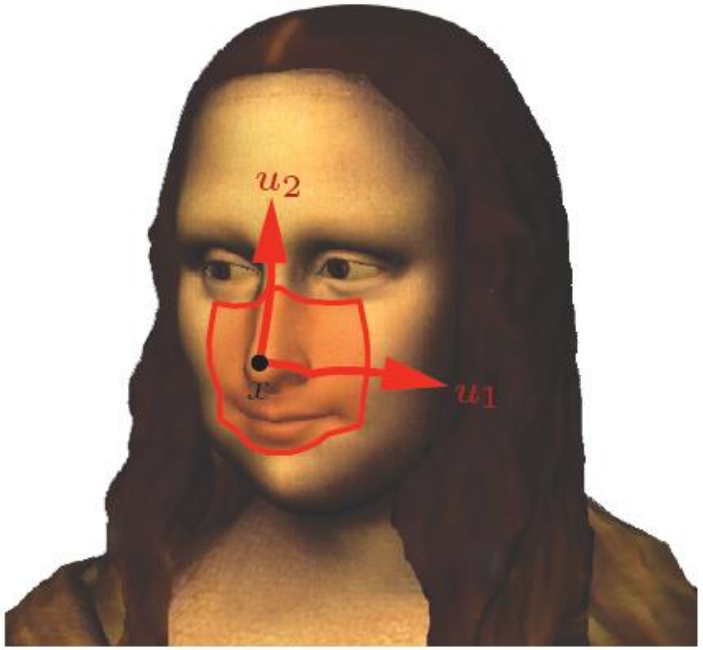
Non-Euclidean

Boscaini, Masci, B, Vandergheynst 2015

Spatial convolution



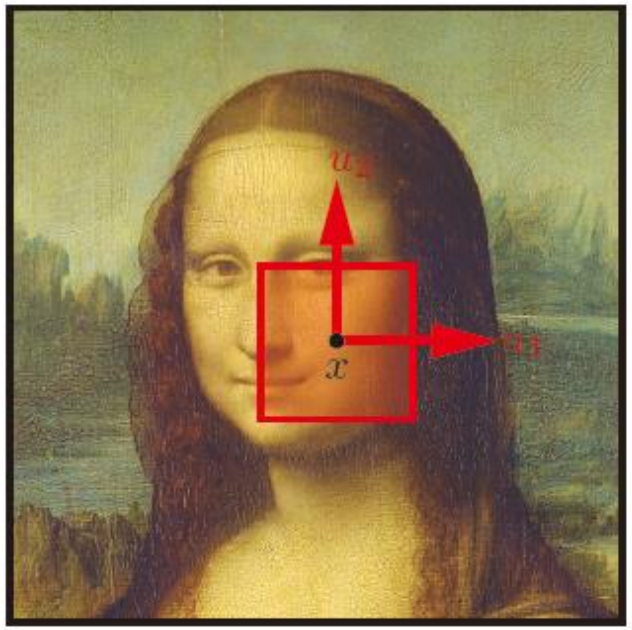
Euclidean



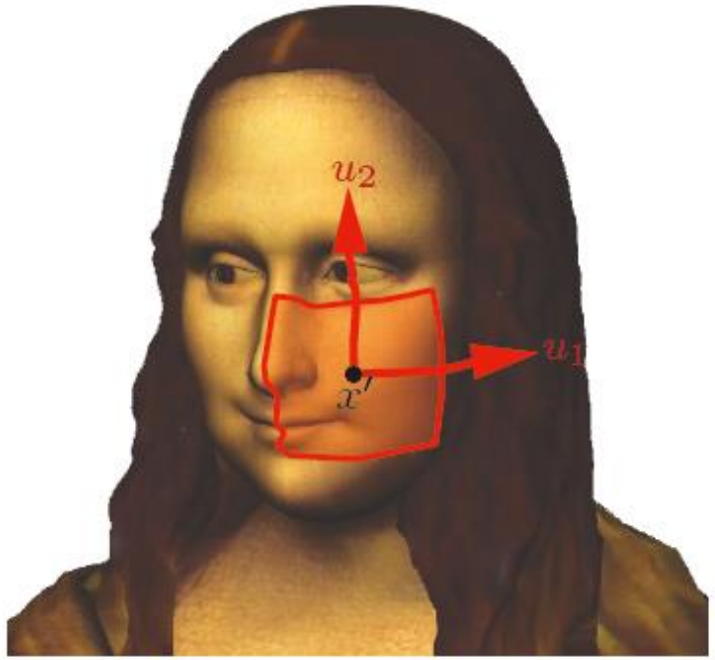
Non-Euclidean

Boscaini, Masci, B, Vandergheynst 2015

Spatial convolution



Euclidean



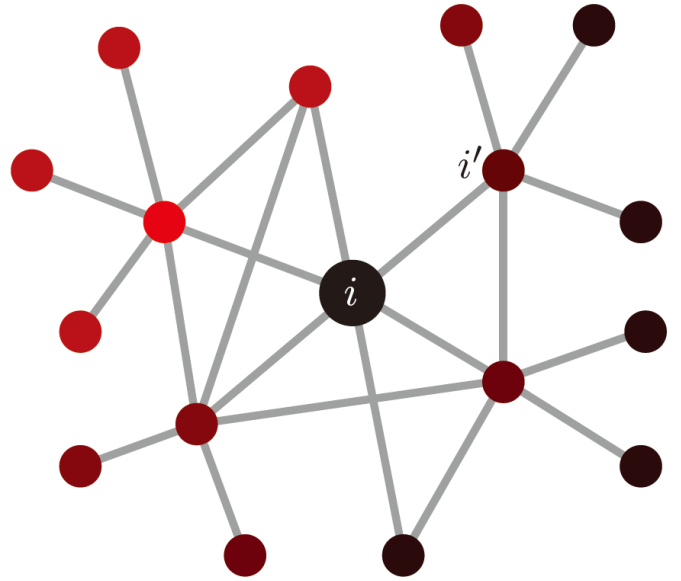
Non-Euclidean

Boscaini, Masci, B, Vandergheynst 2015

Spatial convolution on graphs

- Local feature $\boldsymbol{u}_{i,i'}$, e.g. vertex degree, geodesic distance, ...
- Set of weighting functions

$$w_1(\boldsymbol{u}), \dots, w_j(\boldsymbol{u})$$



Spatial convolution

$$(f \star g)_i = \sum_{j=1}^{J} g_j \sum_{i'=1}^{N} w_j(\boldsymbol{u}_{i,i'}) f_{i'}$$

where $g_1, \dots, g_J$ are the spatial filter coefficient, $\sum_{i'=1}^{N} w_j(\boldsymbol{u}_{i,i'})$ are patch operators

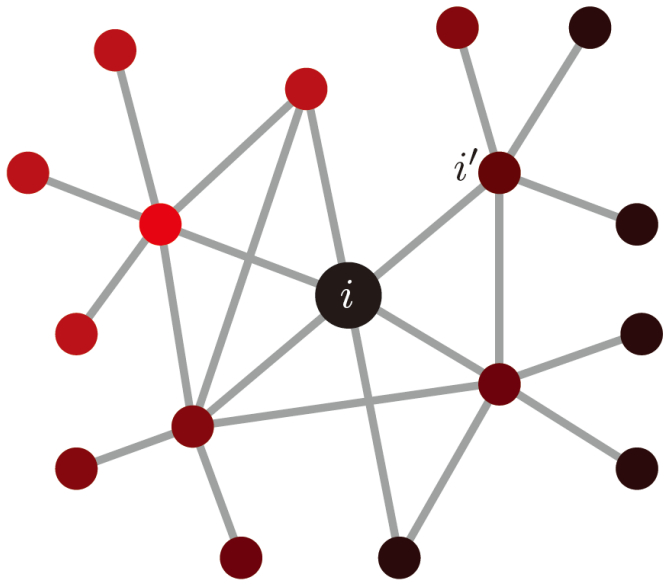
Monti, Boscaini, Rodola, Svoboda, B 2017

Learnable patch operator

- Local feature $\boldsymbol{u}_{i,i'}$, e.g. vertex degree, geodesic distance,...
- Gaussian weighting functions

$$w_{\mu,\Sigma}(\boldsymbol{u}) = \exp\left(-\frac{1}{2}(\boldsymbol{u}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{u}-\boldsymbol{\mu})\right)$$

with learnable covariance $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$



Spatial convolution

$$(f \star g)_i(x) = \sum_{j=1}^{J} g_j \sum_{i'=1}^{N} w_{\mu_j,\Sigma_j}(\boldsymbol{u}_{i,i'}) f_{i'}$$

where $g_1, \ldots, g_J$ are the spatial filter coefficients and $\mu_1, \ldots, \mu_J$ and $\Sigma_1, \ldots, \Sigma_J$ are patch operator parameters

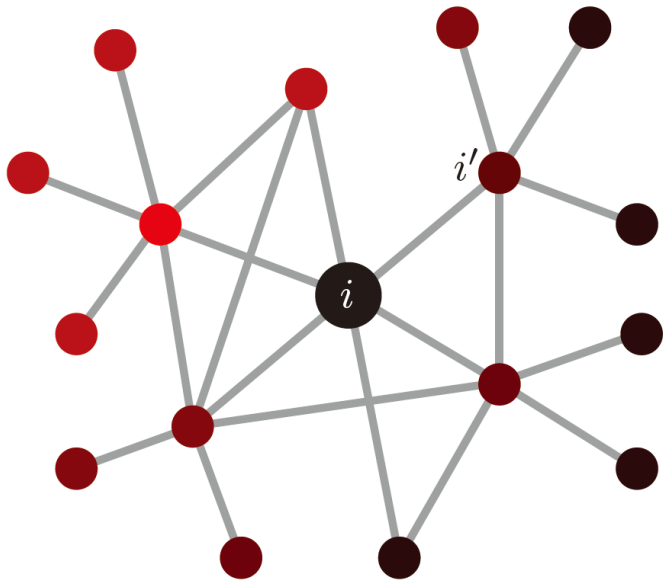
Monti, Boscaini, Rodola, Svoboda, B 2017

Mixture Model Network (MoNet)

- Local feature $\boldsymbol{u}_{i,i'}$, e.g. vertex degree, geodesic distance,...
- Gaussian weighting functions

$$w_{\mu,\Sigma}(\boldsymbol{u}) = \exp\left(-\frac{1}{2}(\boldsymbol{u}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{u}-\boldsymbol{\mu})\right)$$

with learnable covariance $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$



Spatial convolution

$$(f \star g)_i(x) = \sum_{i'=1}^{N}\sum_{j=1}^{J} g_j w_{\mu_j,\Sigma_j}(\boldsymbol{u}_{i,i'}) f_{i'}$$

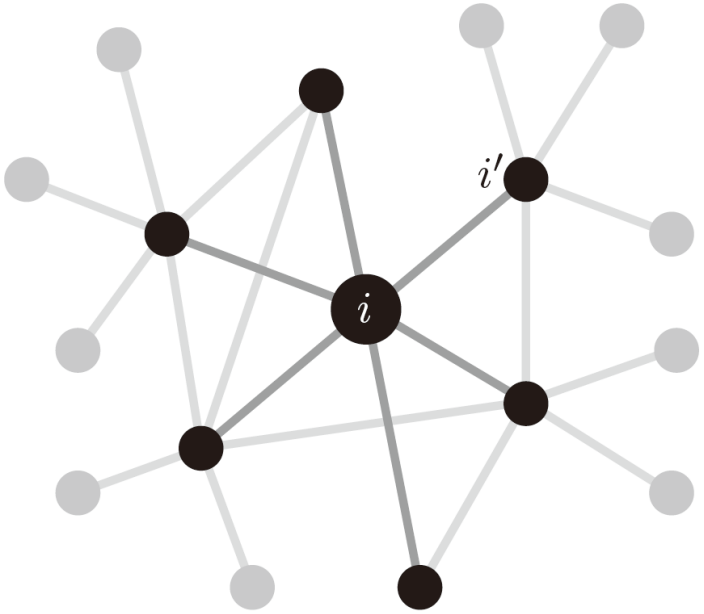
where $g_1, \ldots, g_J$ are the spatial filter coefficients and $\mu_1, \ldots, \mu_J$ and $\Sigma_1, \ldots, \Sigma_J$ are patch operator parameters

Monti, Boscaini, Rodola, Svoboda, B 2017

Edge convolution (DGCNN)

- Permutation-invariant aggregation operator $\square$ (e.g. Σ or max) on the neighborhood of i
- Edge feature function $h_\Theta(\cdot,\cdot)$ parametrized by Θ
- Edge convolution

$$f_i' = \square_{i':(i,i')\in\mathcal{E}} h_\Theta(f_i, f_i')$$



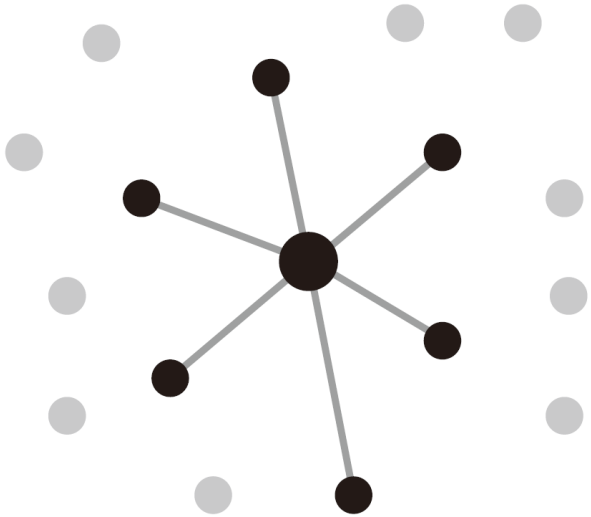
Method	Aggregation $\square$	Edge feature h
PointNet[1]	–	$h(f_i, f_{i'}) = h(f_i)$
MoNet[2]	$\sum$	$h(f_i, f_{i'}) = \sum_{j=1}^{J} g_j w_{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j}(\mathbf{u}_{ii'}) f_{i'}$

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas, “PointNet: Deep learning on point sets for 3d classification and segmentation,” *IEEE Conference on Computer Vision and Pattern Recognition(CVPR), Honolulu, HI, USA, July, 2017*

Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic Graph CNN for Learning on Point Clouds," *arXiv preprint arXiv:1801.07829*, 2018.

Edge convolution (DGCNN)

- Construct k-NN graph in feature space and update it after each layer

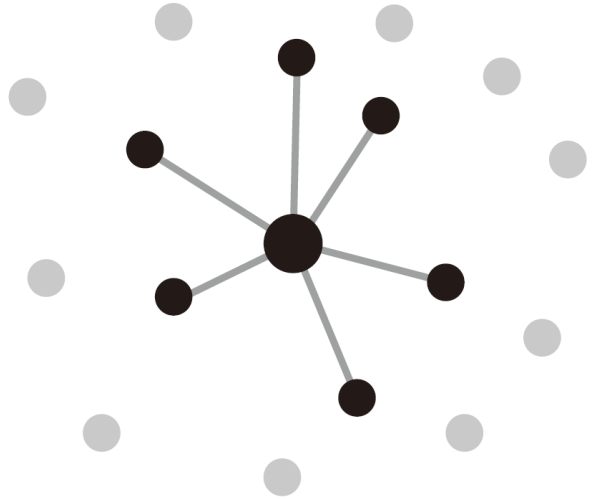


Layer l

Features $\mathbf{x}_1^{(l)}, \ldots, \mathbf{x}_n^{(l)} \in \mathbb{R}^{d_l}$

k-NN graph $\mathcal{G}^{(l)}$

$h^{(l)} : \mathbb{R}^{d_l} \times \mathbb{R}^{d_l} \to \mathbb{R}^{d_{l+1}}$



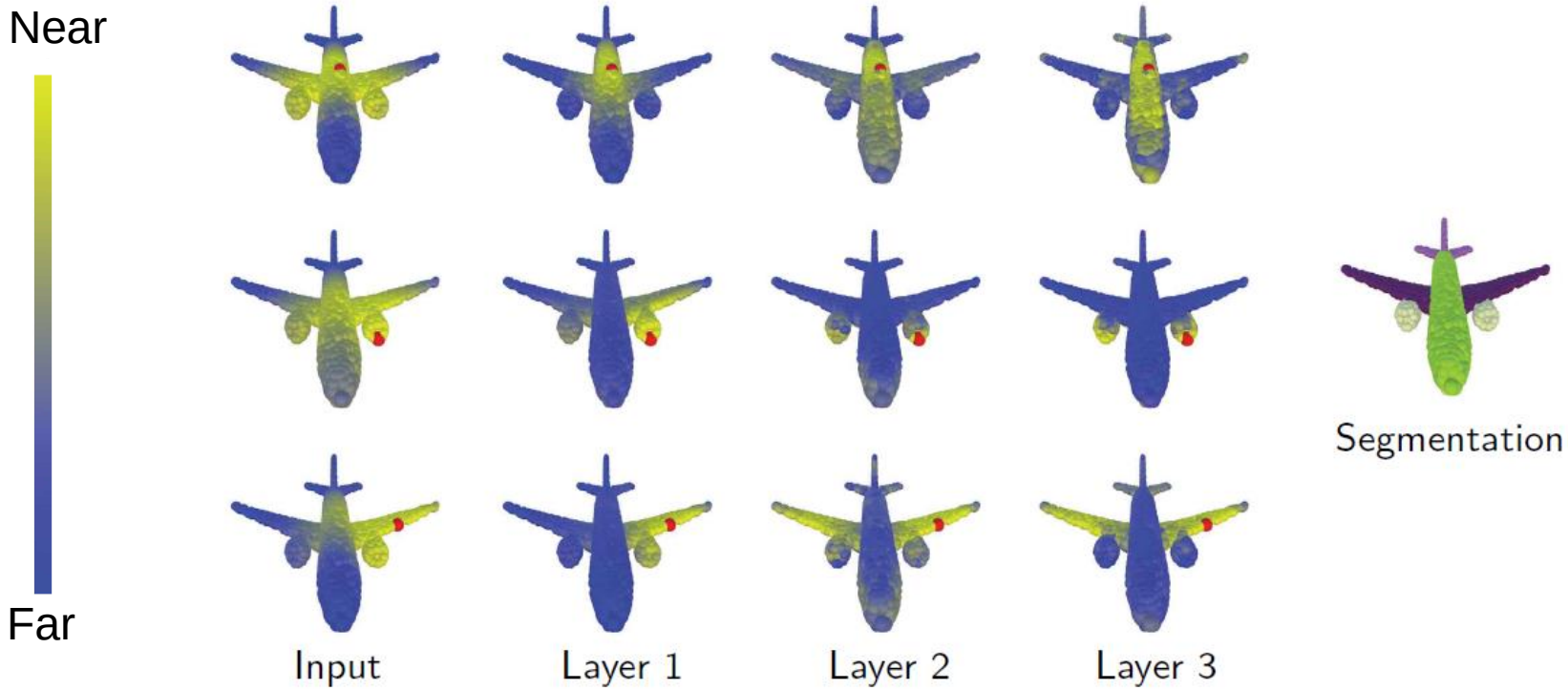
Layer $l+1$

Features $\mathbf{x}_1^{(l+1)}, \ldots, \mathbf{x}_n^{(l+1)} \in \mathbb{R}^{d_{l+1}}$

k-NN graph $\mathcal{G}^{(l+1)}$

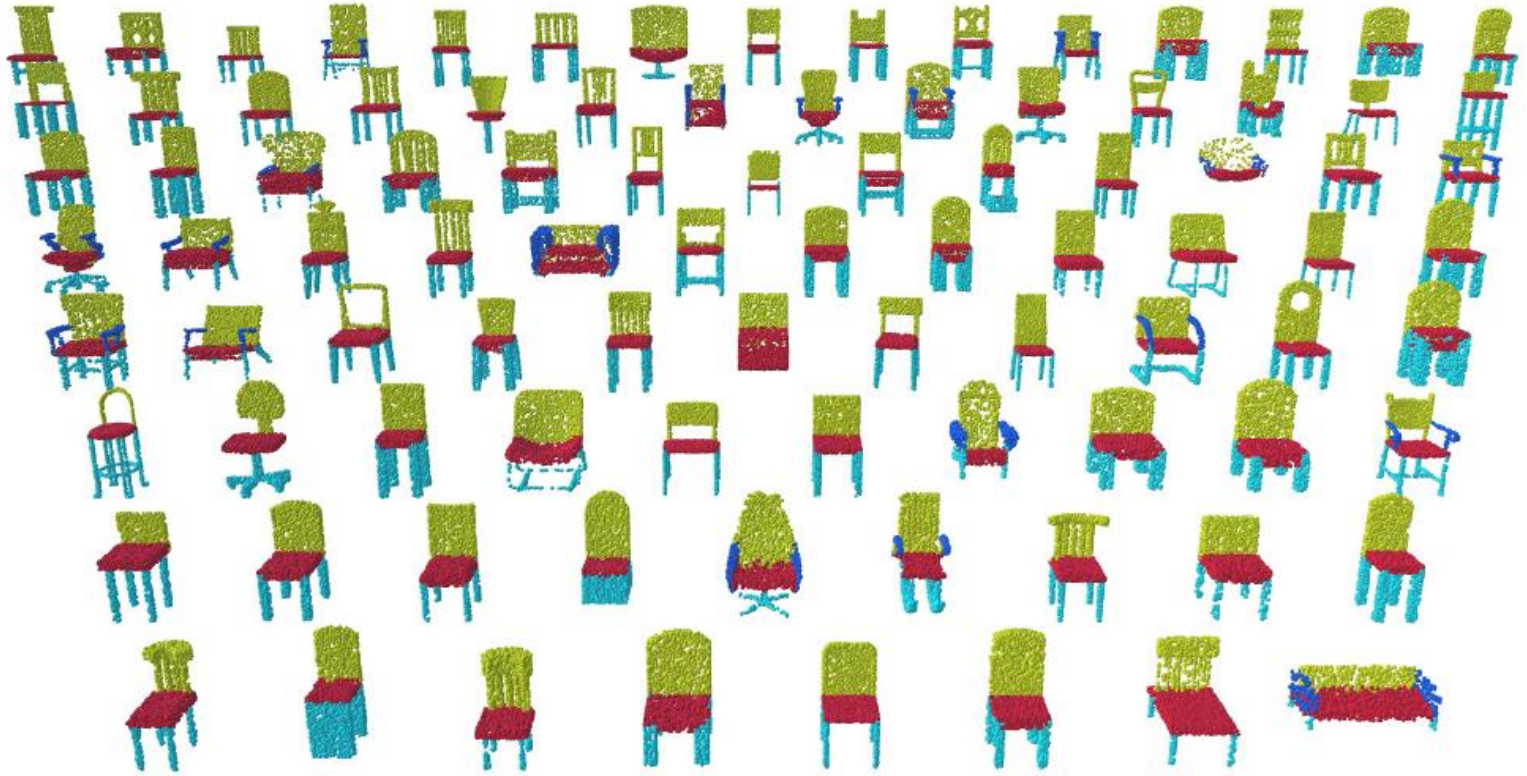
$h^{(l+1)} : \mathbb{R}^{d_{l+1}} \times \mathbb{R}^{d_{l+1}} \to \mathbb{R}^{d_{l+2}}$

Learning semantic features



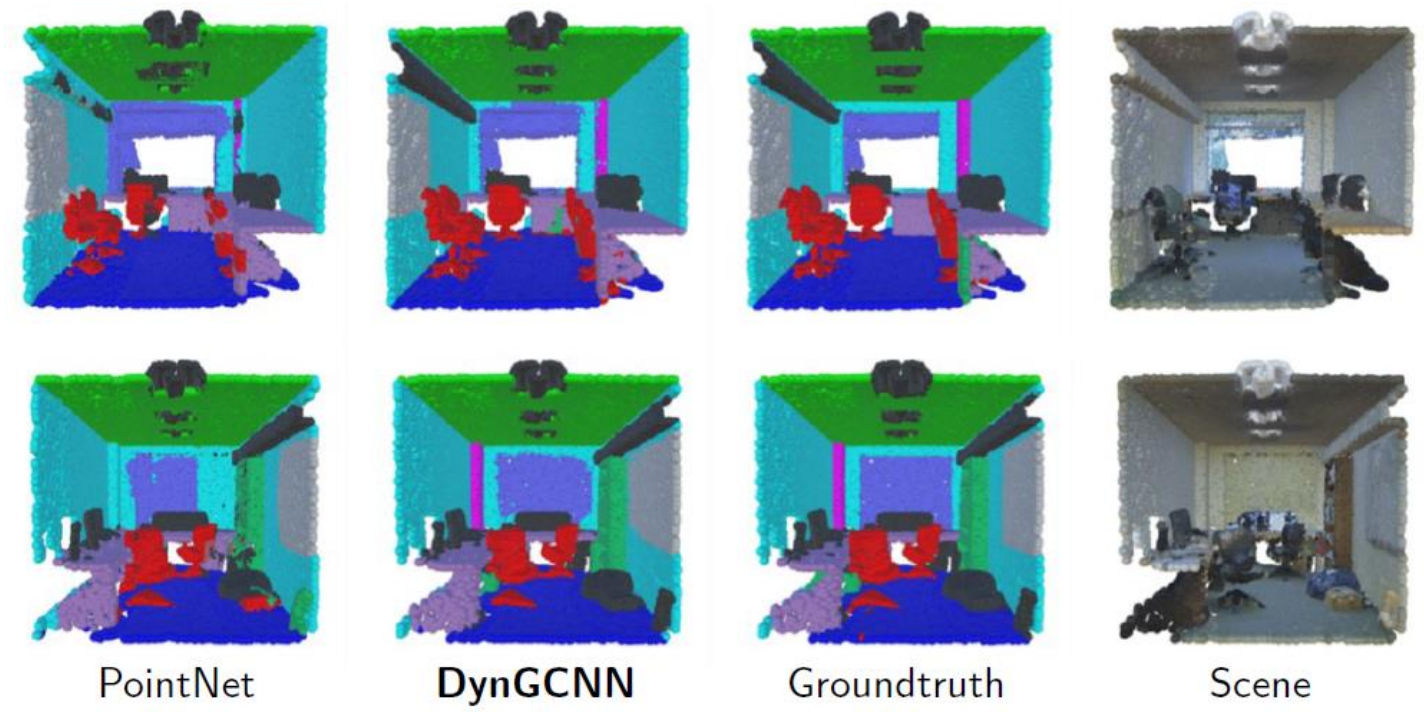
Left: Distance from red point in the feature space of different DGCNN layers
Right: semantic segmentation results

Semantic segmentation



Point cloud semantic segmentation using DGCNN

Semantic segmentation



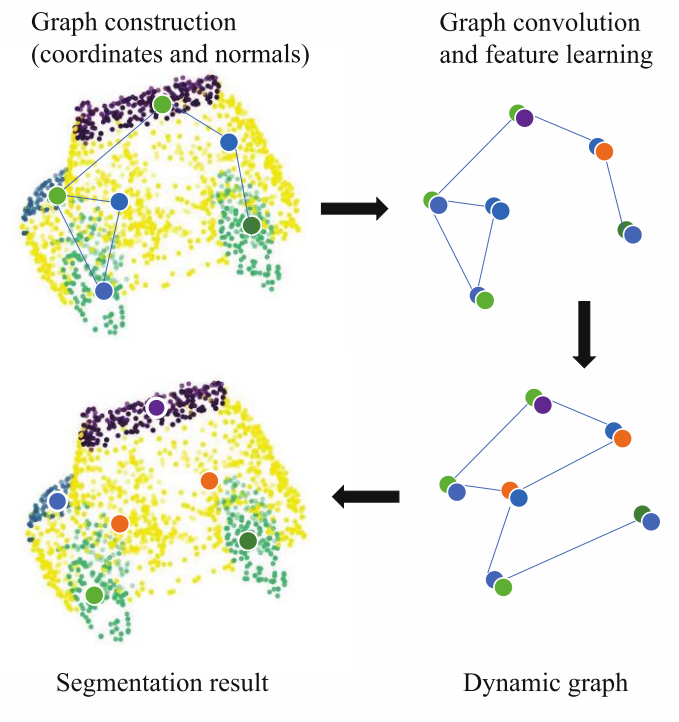
Results of semantic segmentation of point cloud+RGB data using different architectures

Methods: Qi et al. 2017 (PointNet); Wang, Sun, Liu, Sarma, B, Solomon 2018 (DynGCNN); data: Armeni et al. 2016 (S3DIS)

Outline

- Introduction to Graph Convolutional Neural Networks (Graph CNN)
- Background in Graph Signal Processing for analysis on graphs
- Spectral Graph CNNs
- Spatial Graph CNNs
- Applications, Challenges, Open problems

Robust Point cloud segmentation



Framework

- Real-word data often suffer from noise, missing data…
- Integrate graph-signal smoothness prior to the loss

$$\text{Loss} = -\sum_{i=1}^{N} y_i \log(y_i') + \sum_{i\sim j} a_{i,j} \left\| y_i - y_j \right\|_2^2$$

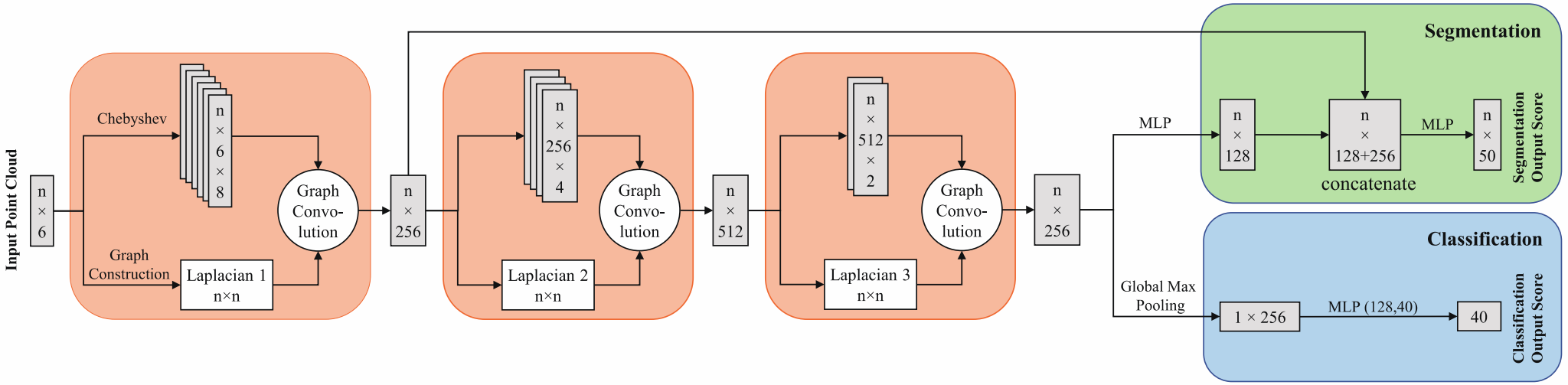
output labels　　g. t. labels　　edge weight

-- enforce the features of adjacent vertices to be more similar.

G. Te, W. Hu, A. Zheng, and Z. Guo, "RGCNN: Regularized graph CNN for point cloud segmentation," *ACM Multimedia Conference*, Seoul, Korea, October 2018.

Robust Point cloud segmentation

- Update the graph Laplacian dynamically

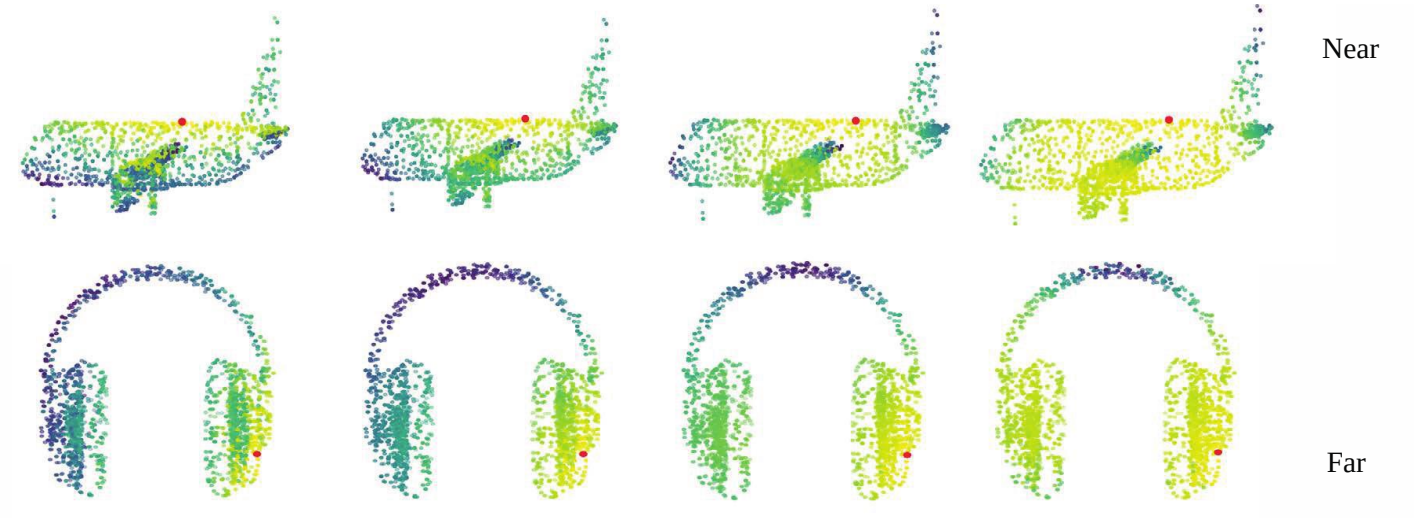


Network Structure

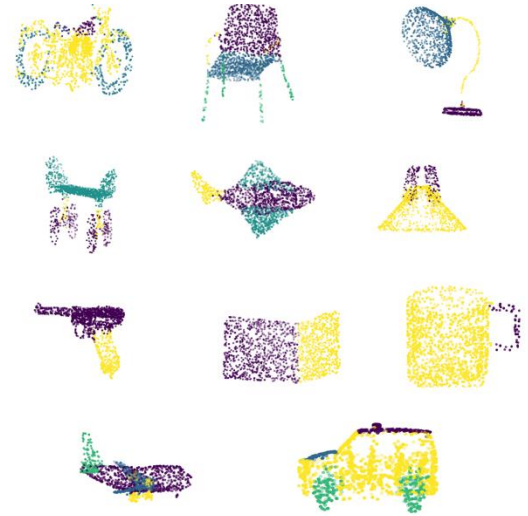
Robust Point cloud segmentation

Performance on ShapeNet

	mean	aero	bag	cap	car	chair	earphone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skateboard	table
ShapeNet	81.4	81.0	78.4	77.7	75.7	87.6	61.9	92.0	85.4	82.5	95.7	70.6	91.9	**85.9**	53.1	69.8	75.3
PointNet	83.7	**83.4**	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
PointNet++	**85.1**	82.4	79.0	87.7	**77.3**	**90.8**	71.8	91.0	85.9	83.7	95.3	**71.6**	94.1	81.3	58.7	76.4	**82.6**
SynSpecCNN	84.7	81.6	81.7	81.9	75.2	90.2	**74.9**	**93.0**	86.1	**84.7**	95.6	66.7	92.7	81.6	**60.6**	**82.9**	82.1
Ours	84.3	80.2	**82.8**	**92.6**	75.3	89.2	73.7	91.3	**88.4**	83.3	**96.0**	63.9	**95.7**	60.9	44.6	72.9	80.4

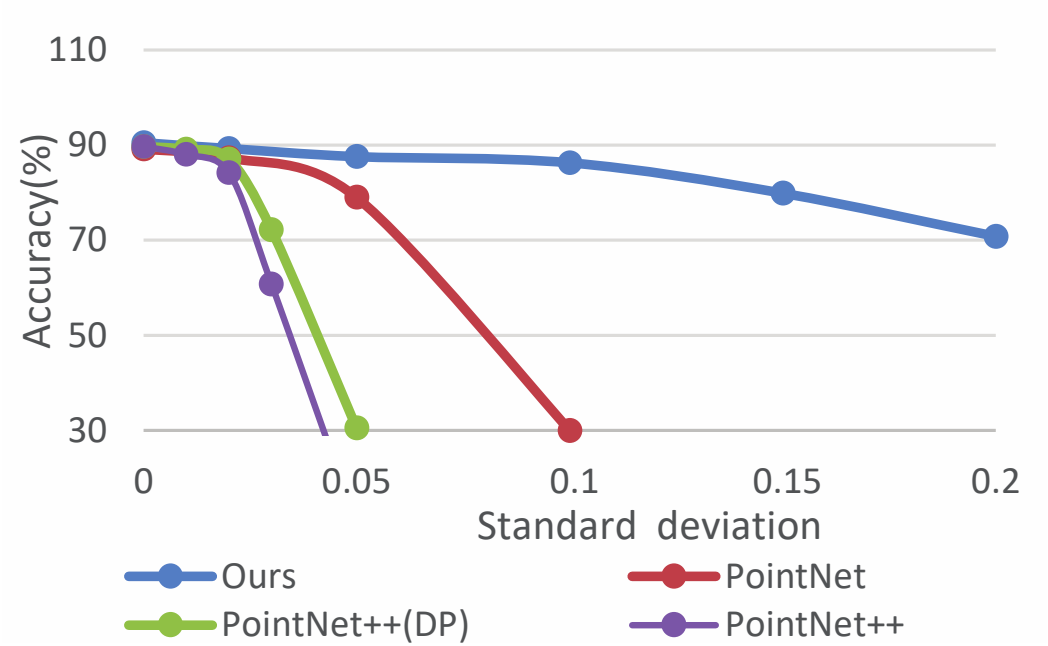


Visualization of feature space



Results

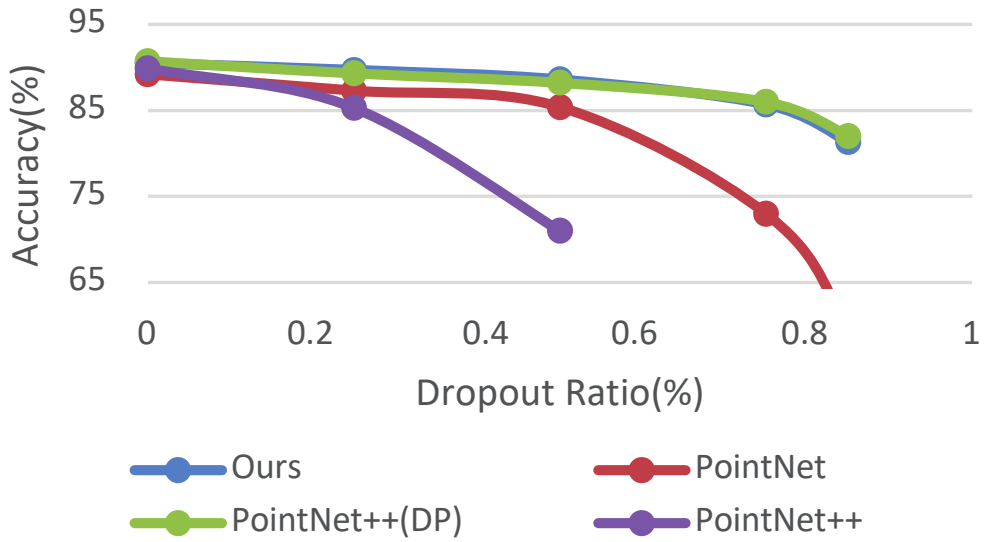
Robust Point cloud segmentation



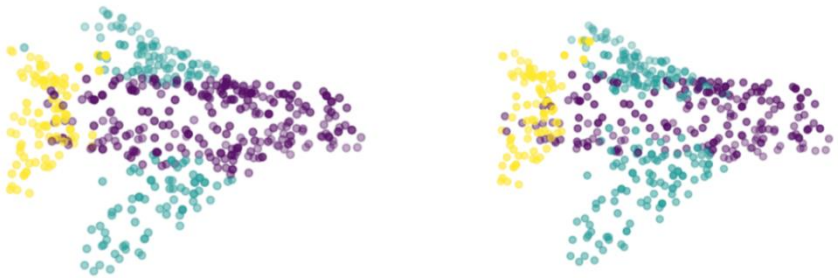
Accuracy with Gaussian noise

(a) Ground truth (b) Ours

Robust Point cloud segmentation



Accuracy with missing data



(a) Ground truth

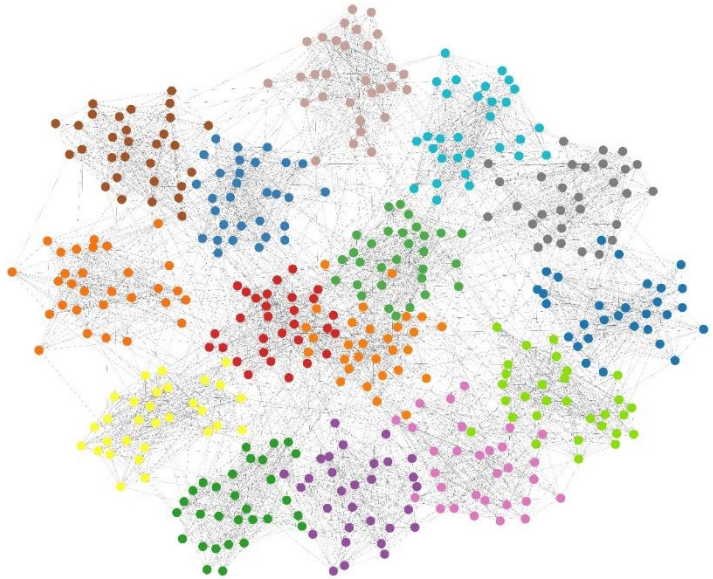
(b) Ours

Exploring graph learning for semi-supervised classification for social networks

- Pose a Maximum a Posteriori (MAP) estimation on the adjacency matrix

$$\widetilde{\boldsymbol{A}}_{\mathrm{MAP}}(x) = \underset{\widehat{\boldsymbol{A}}}{\operatorname{argmax}}\, f\left(x\middle|\widehat{\boldsymbol{A}}\right) g\left(\widehat{\boldsymbol{A}}\right)$$

  - $\widehat{\boldsymbol{A}}$ is the optimal graph adjacency matrix
  - x is the observed graph signal
  - $f\left(x\middle|\widehat{\boldsymbol{A}}\right)$ is the likelihood function
  - $g\left(\widehat{\boldsymbol{A}}\right)$ is the prior distribution



X. Gao, W. Hu, and Z. Guo, "Exploring Graph Learning for Semi-Supervised Classification Beyond Euclidean Data," *arXiv preprint arXiv:1904.10146*, 2019.

Exploring graph learning for semi-supervised classification for social networks

- Proposed likelihood function

$$f\left(x\middle|\widehat{\boldsymbol{A}}\right) = \beta \exp\left(-\lambda_0 x^{\top}\left(I - \widehat{\boldsymbol{A}}\right)x\right)$$

where β and λ_0 are parameters

Exploring graph learning for semi-supervised classification for social networks

- Proposed prior distribution

Properties of a valid adjacency matrix

$$g(\widehat{\boldsymbol{A}}) = \exp\left(-\lambda_1\left\|\widehat{\boldsymbol{A}}\right\|_1 - \lambda_2\left\|\widehat{\boldsymbol{A}}^{\top} - \widehat{\boldsymbol{A}}\right\|_F^2 - \lambda_3\left\|\widehat{\boldsymbol{A}}\mathbf{1} - \mathbf{1}\right\|_F^2 - \lambda_4\left|\mathrm{tr}\left(\widehat{\boldsymbol{A}}\right)\right|^2\right)$$

Sparsity　Symmetry　Normalized　Loopless

λ_i are parameters, $\mathrm{tr}(\cdot)$ and $\|\cdot\|_F$ denote the trace and the Frobenius norm of a matrix.

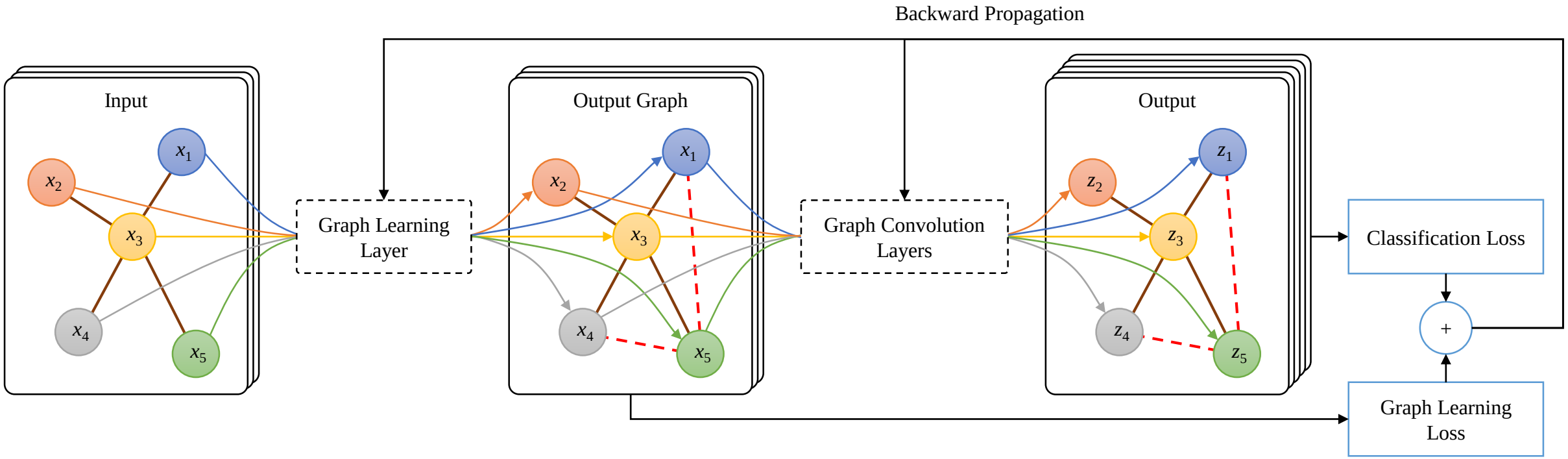
Exploring graph learning for semi-supervised classification for social networks

- Based on MAP estimation, propose graph learning loss function

$$\mathcal{L}_{\mathrm{GL}} = \mathcal{L}_{\mathrm{smooth}} + \mathcal{L}_{\mathrm{sparsity}} + \mathcal{L}_{\mathrm{properties}}$$

  - $\mathcal{L}_{\mathrm{smooth}} = \lambda_0 \left\| x^{\top}\left(I - \widehat{\boldsymbol{A}}_{\mathrm{out}}\right)x \right\|_2^2$
  - $\mathcal{L}_{\mathrm{sparsity}} = \lambda_1 \left\| \widehat{\boldsymbol{A}}_{\mathrm{out}} \right\|_1$
  - $\mathcal{L}_{\mathrm{proterties}} = \lambda_2 \left\| \widehat{\boldsymbol{A}}_{\mathrm{out}}^{\top} - \widehat{\boldsymbol{A}}_{\mathrm{out}} \right\|_2^2 - \lambda_3 \left\| \widehat{\boldsymbol{A}}_{\mathrm{out}}\mathbf{1} - \mathbf{1} \right\|_2^2 - \lambda_4 \left| \mathrm{tr}\left(\widehat{\boldsymbol{A}}_{\mathrm{out}}\right) \right|^2$

Exploring graph learning for semi-supervised classification for social networks



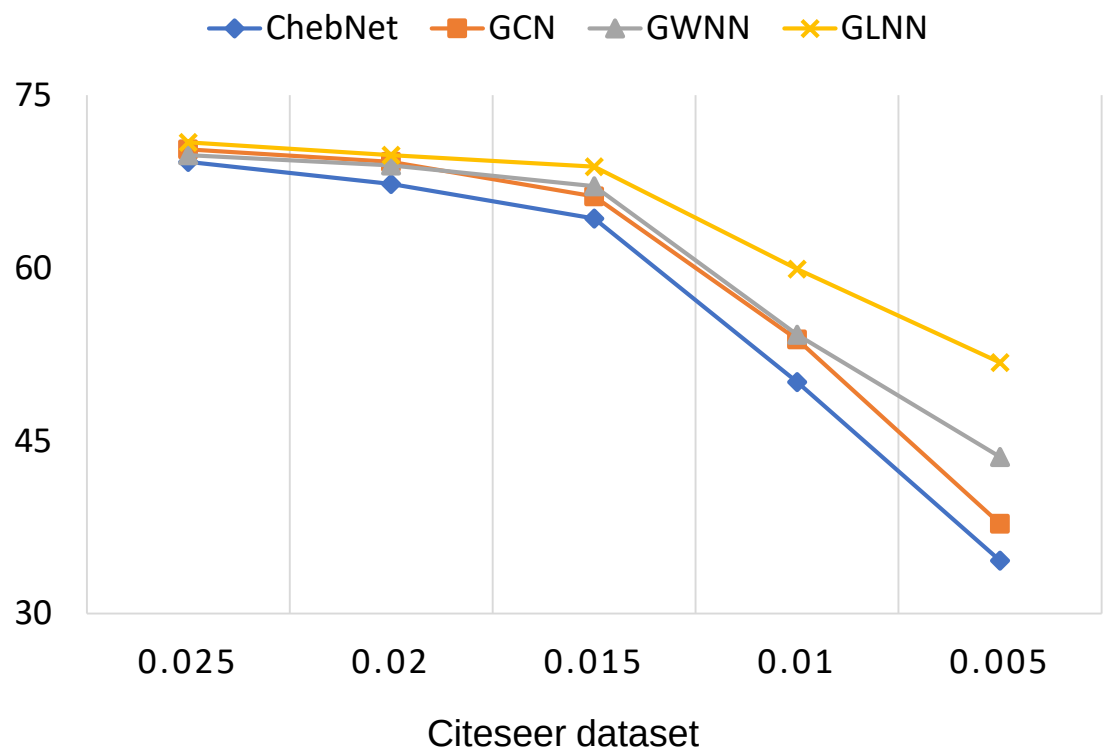
The architecture of the proposed GLNN for semi-supervised node classification.

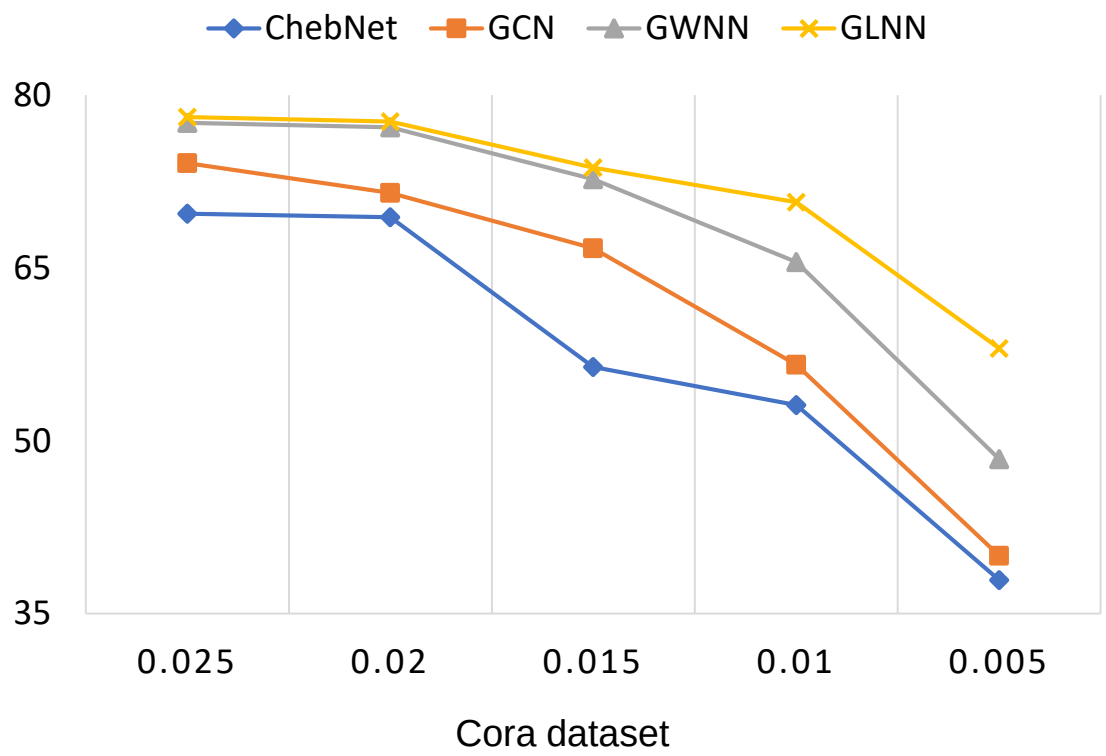
Exploring graph learning for semi-supervised classification for social networks

Results of node classification in terms of accuracy.

Method	Year	TerroristsRel	TerrorAttack	Citeseer	Cora	Pubmed*
Planetoid	2016	-	-	64.7	75.7	74.5
ChebNet	2016	60.1	62.5	69.6	81.2	71.6
MoNet	2017	59.2	62.4	-	81.7	73.5
LoNGAE	2018	63.5	66.0	71.8	79.0	74.7
GAT	2018	63.5	65.5	71.0	82.3	72.3
DGI	2019	52.7	61.5	71.8	82.3	74.6
GWNN	2019	65.5	65.0	71.7	82.8	72.4
GCN	2017	62.2	66.0	70.3	81.5	73.8
GLNN		**66.2**	**68.0**	**72.2**	**83.4**	**74.8**

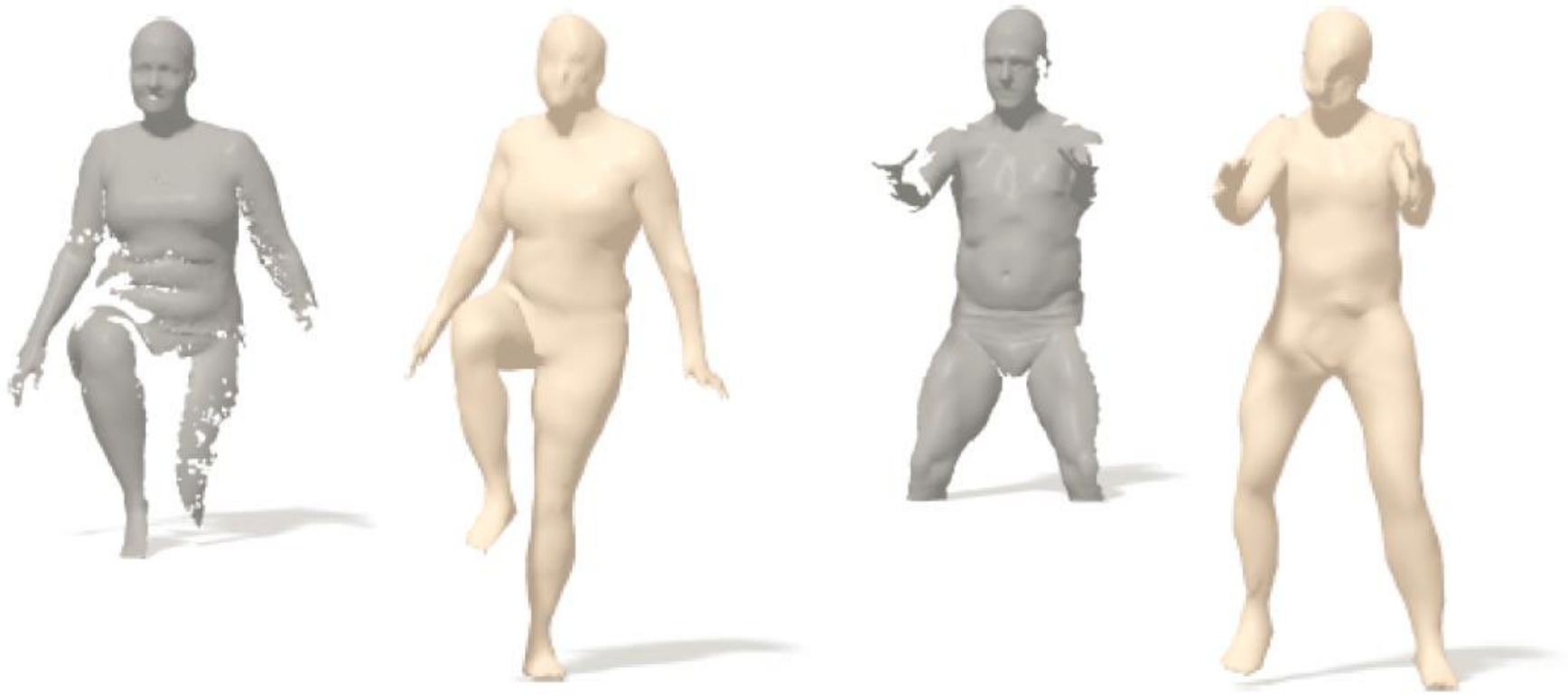
Exploring graph learning for semi-supervised classification for social networks





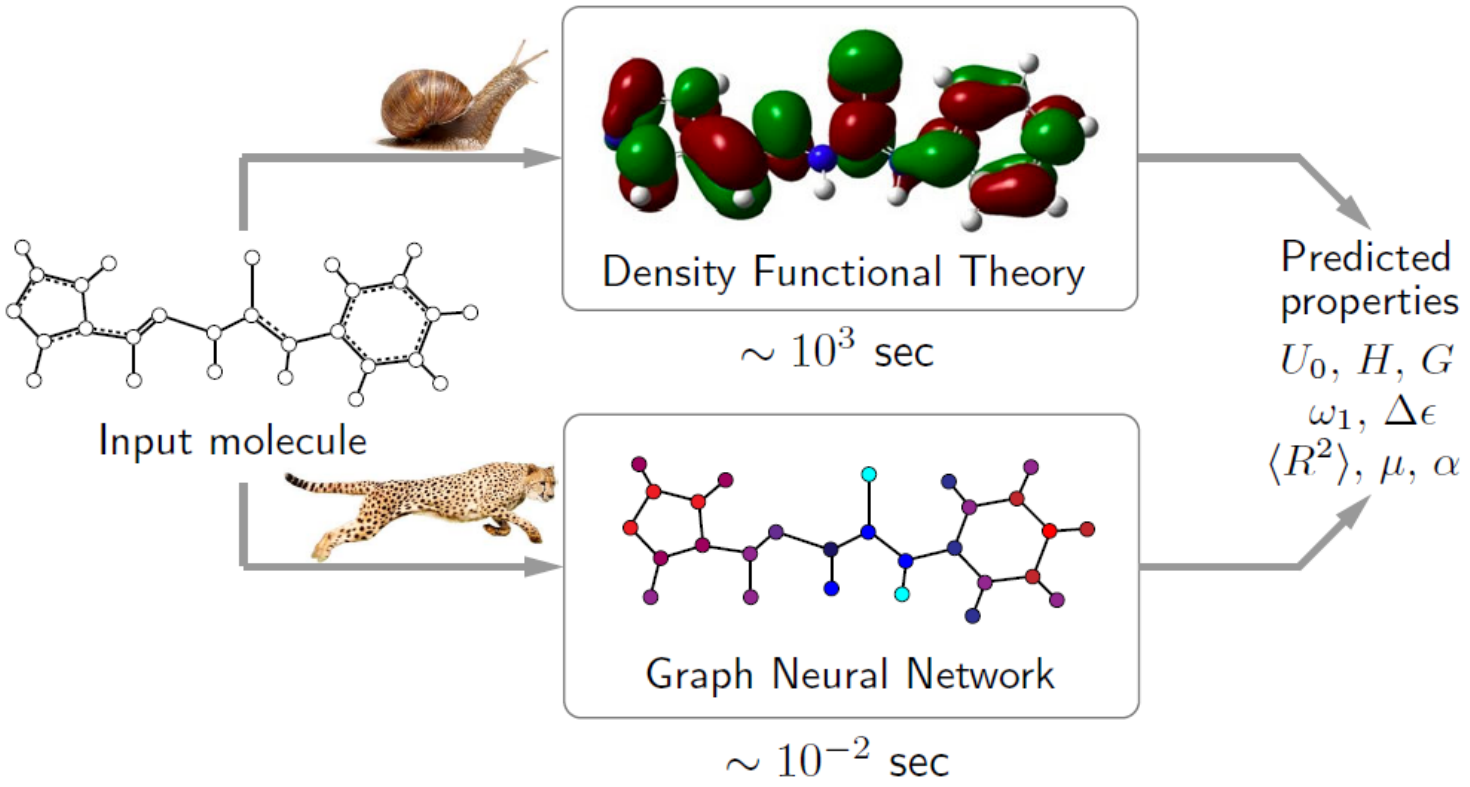
Classification accuracy comparison with different label rates

Shape completion with Intrinsic VAE



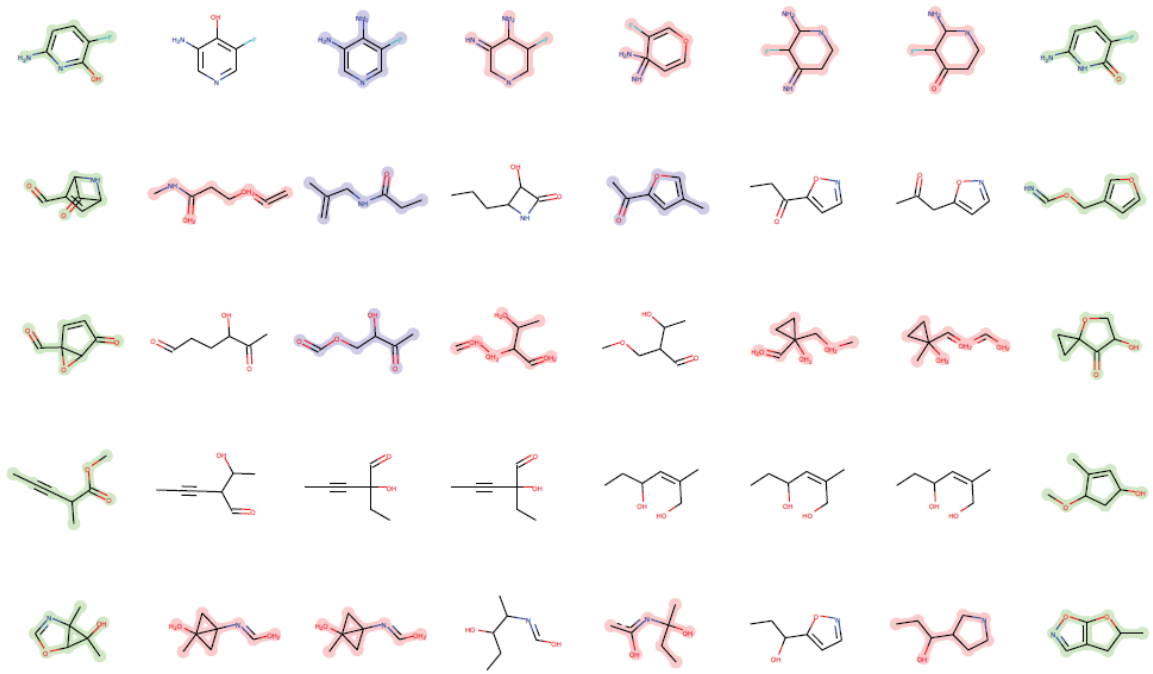
Litany, BB, Makadia 2018; data: Bogo et al. 2017 (FAUST scans)

Molecule property prediction



Duvenaud et al. 2015; Gomez-Bombarelli et al. 2016; Gilmer et al. 2017

Molecule generation



Molecules generated with a graph VAE

Simonovsky, Komodakis 2017; You et al. 2018
Collaboration with A. Aspuru-Guzik et al. (Harvard)

Open questions and future directions

- How to construct a graph? (graph learning)
- How to avoid over-fitting? (deep structure)
- How to achieve efficient computations? (large-scale graph processing)

Open questions and future directions

- Time-varying domains (e.g., dynamic 3D point clouds)
- Directed graphs (e.g., scene graph, citation network)
- Synthesis problems (generative model, e.g., point cloud GAN)
- Reinforcement learning on graphs?
- Killer applications?

北京大学

Thank you!

forhuwei@pku.edu.cn
http://www.icst.pku.edu.cn/huwei/