UC San Diego

Lecture 13
Graph Convolutional Neural Networks

Presenters: Zhiyao Yan, Sainan Liu, Gautam Nain

Feb 18, 2018

Outline

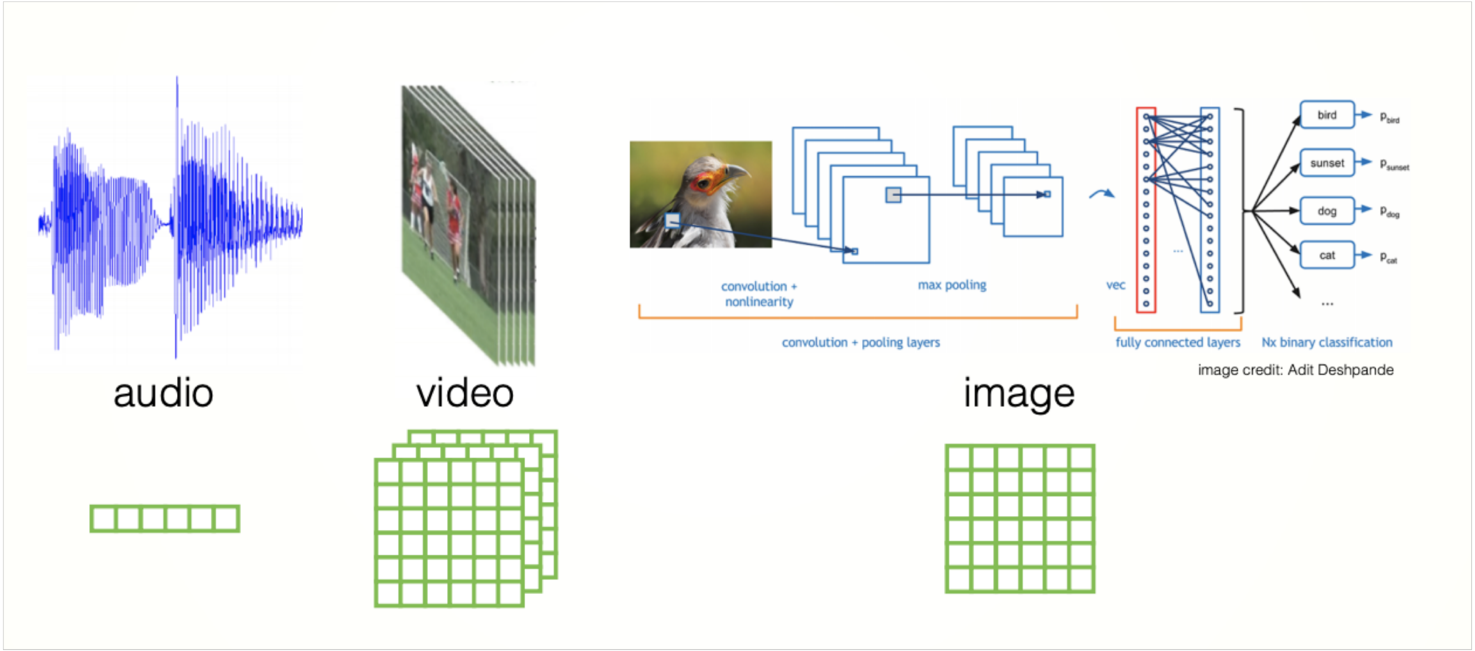
- Motivation
- Background:
  - Graph Definition
  - Spatial vs Spectral Domains
- Challenges
- Paper:
  - DCNN
  - GATs
  - SyncSpecCNN
- Summary
- Takeaway & References

Outline

- Motivation
- Background:
  - Graph Definition
  - Spatial vs Spectral Domains
- Challenges
- Paper:
  - DCNN
  - GATs
  - SyncSpecCNN
- Summary
- Takeaway & References

Motivation

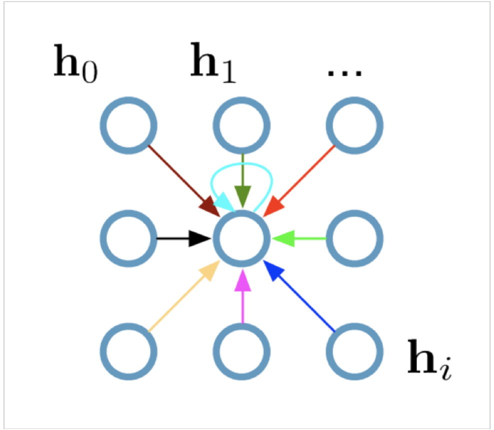
- CNN has been very successful for a wide variety of applications

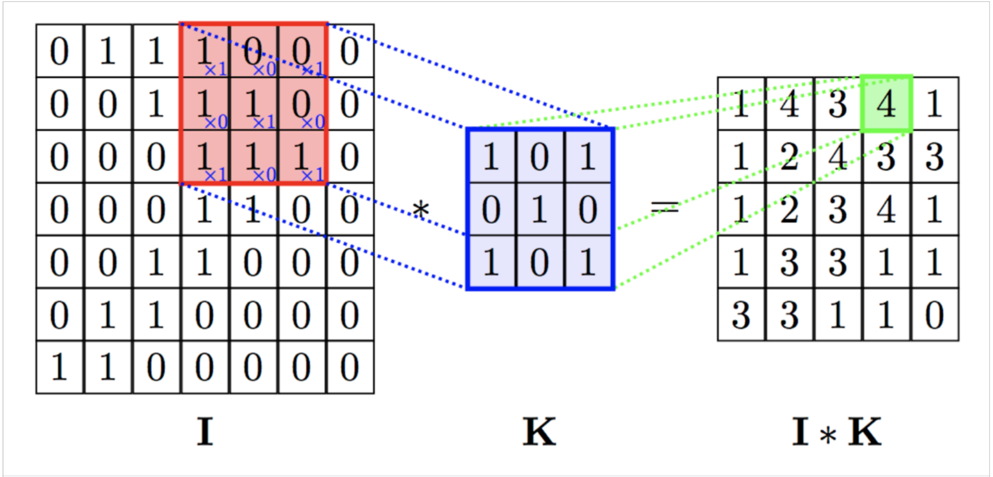


Slide borrowed from Li Yi

Motivation

- CNN nicely exploits the Grid Structure





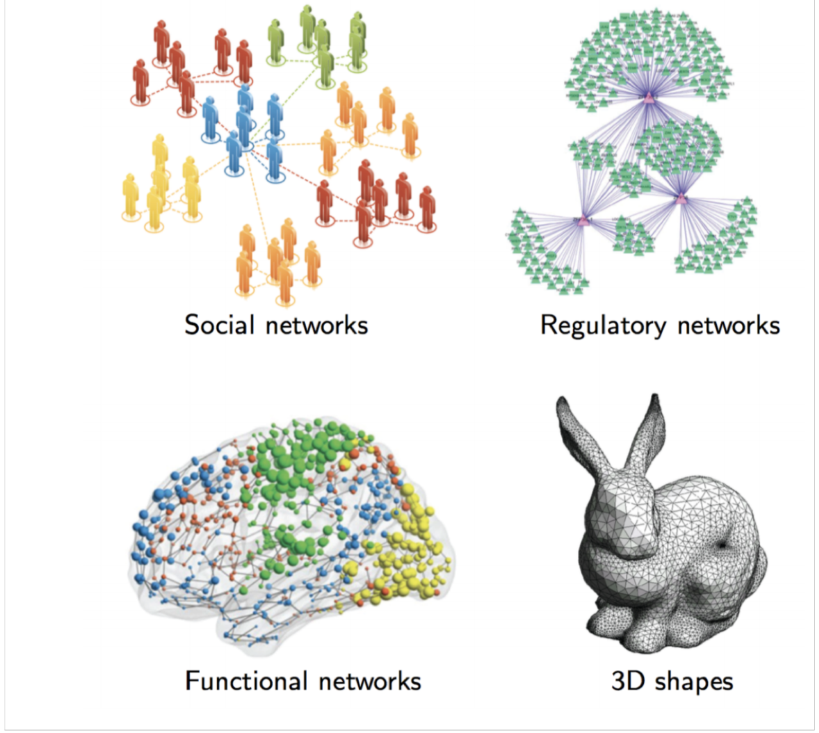
$$\begin{bmatrix} 0&1&1&1&0&0&0\\ 0&0&1&1&1&0&0\\ 0&0&0&1&1&1&0\\ 0&0&0&1&1&0&0\\ 0&0&1&1&0&0&0\\ 0&1&1&0&0&0&0\\ 1&1&0&0&0&0&0 \end{bmatrix} * \begin{bmatrix} 1&0&1\\ 0&1&0\\ 1&0&1 \end{bmatrix} = \begin{bmatrix} 1&4&3&4&1\\ 1&2&4&3&3\\ 1&2&3&4&1\\ 1&3&3&1&1\\ 3&3&1&1&0 \end{bmatrix}$$

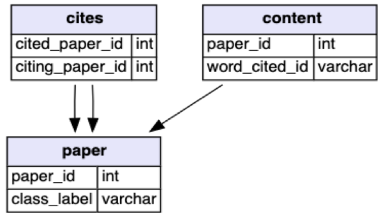
2D convolutional operator as applied to a grid-structured input (e.g. image).

Images borrowed from Graph attention networks

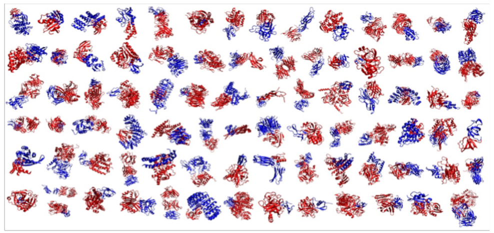
Motivation

- What about data with arbitrary structures like Graphs?

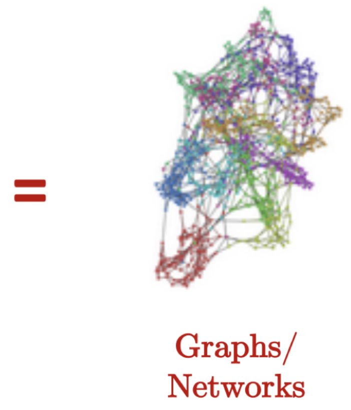




CORA



PPI Affinity 2.0

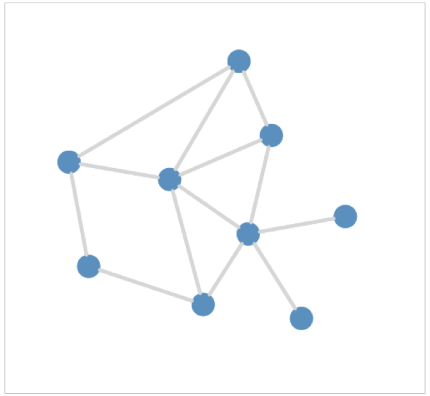


Also chemistry, physics, communication networks, social science etc.

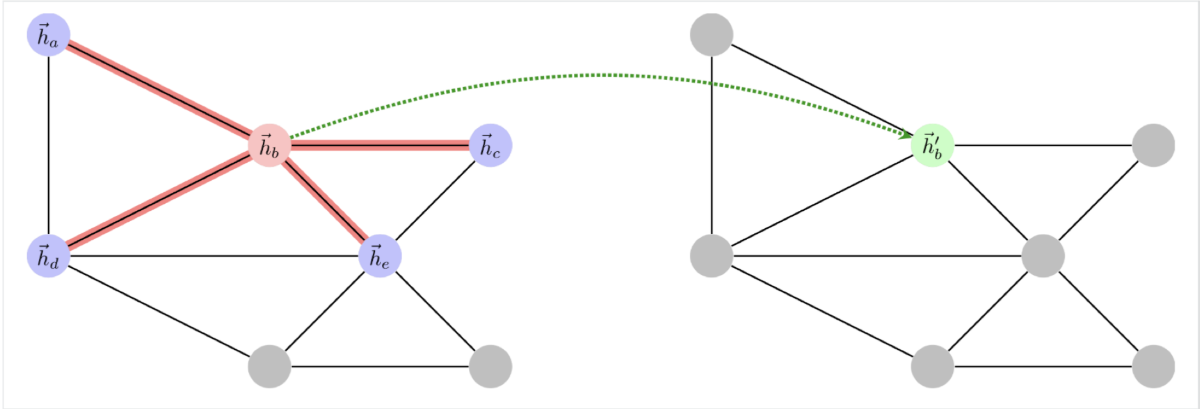
Image borrowed from Bresson's

Motivation

What if the data looks like this →



We can Still Perform Convolution as below



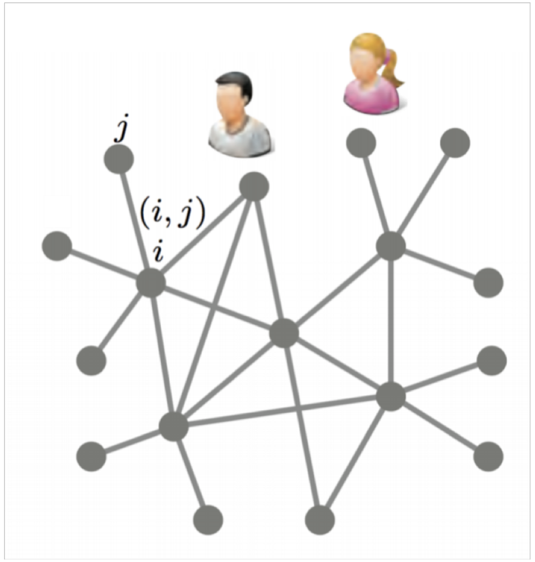
A desirable form of a graph convolutional operator.

Image Borrowed from [Graph attention networks]

Outline

Graph Definition

Graph	$\mathcal{G} = (\mathcal{V}, \mathcal{E})$
Vertices	$\mathcal{V} = \{1, \ldots, n\}$
Edges	$\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$
Vertex weights	$b_i > 0$ for $i \in \mathcal{V}$
Edge weights	$a_{ij} \geq 0$ for $(i, j) \in \mathcal{E}$



Borrowed from Yi Li

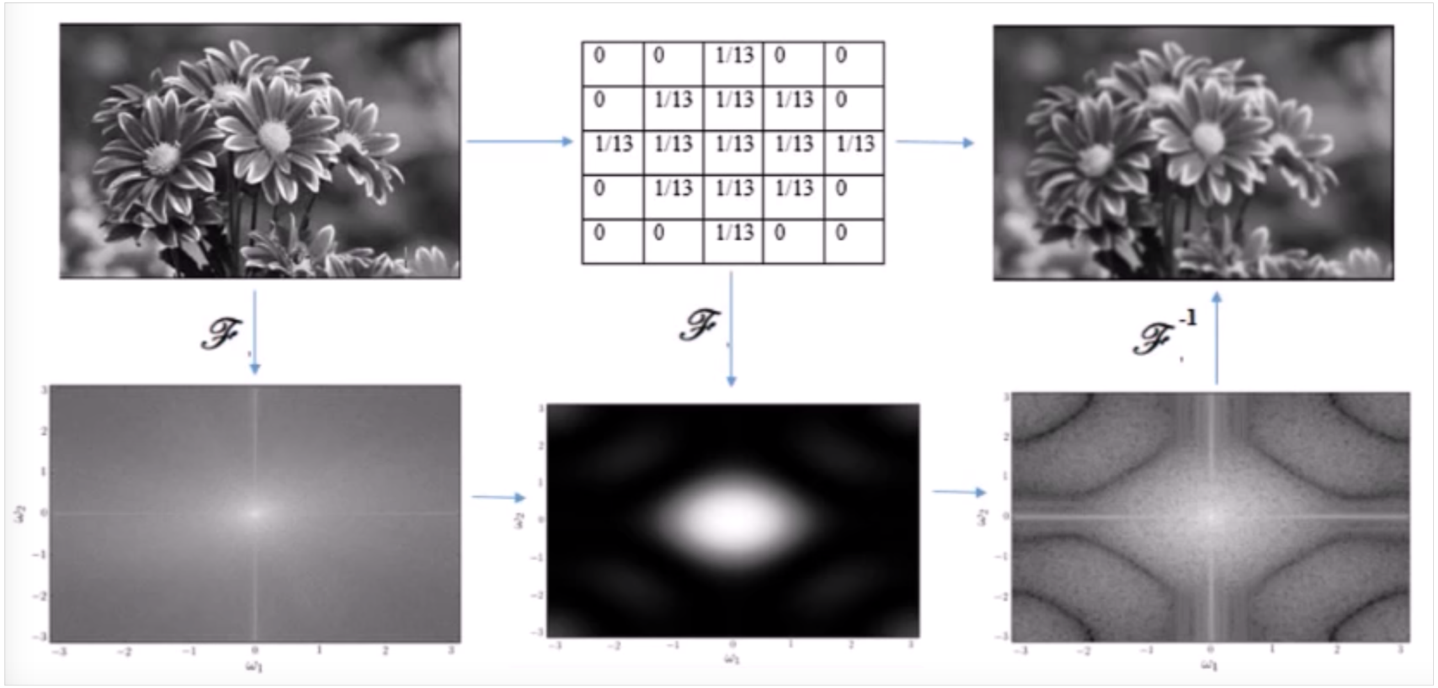
Outline

Spatial vs Spectral

- Two major approaches to build Graph CNNs
  - Spatial Domain:
    - Perform convolution in spatial domain similar to images (euclidean data) with shareable weight parameters
  - Spectral Domain:
    - Convert Graph data to spectral domain data by using the eigenvectors of laplacian operator on the graph data and perform learning on the transformed data.

Spatial vs Spectral - 2D Image

- Convolution comparison:

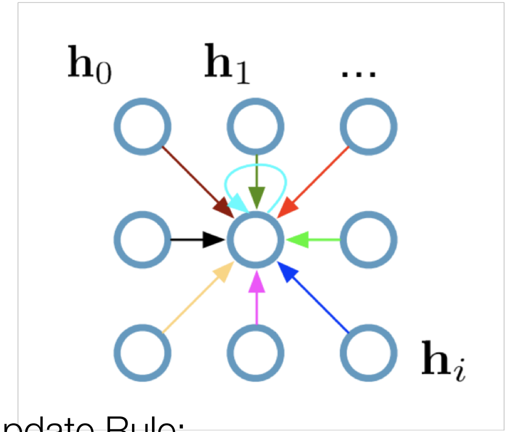


0	0	1/13	0	0
0	1/13	1/13	1/13	0
1/13	1/13	1/13	1/13	1/13
0	1/13	1/13	1/13	0
0	0	1/13	0	0

Borrowed from Sidd Signal

Spatial Domain

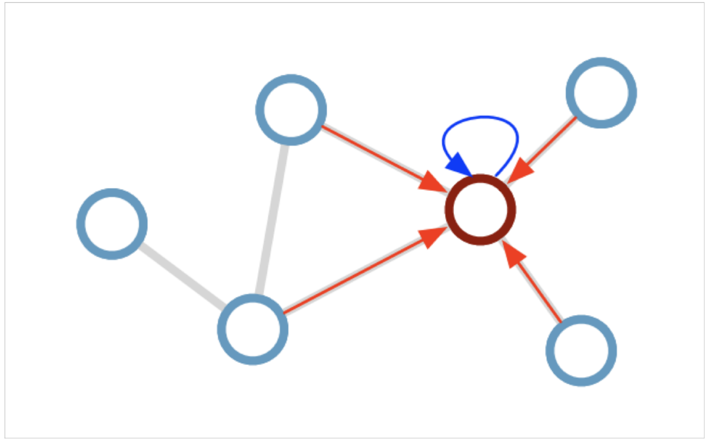
Standard CNN on Images:



Update Rule:

$$\mathbf{h}_4^{(l+1)} = \sigma\left(\mathbf{W}_0^{(l)}\mathbf{h}_0^{(l)} + \mathbf{W}_1^{(l)}\mathbf{h}_1^{(l)} + \cdots + \mathbf{W}_8^{(l)}\mathbf{h}_8^{(l)}\right)$$

CNN on Graphs:



Update Rule:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

$\mathcal{N}_i$: neighbor indices

c_{ij} : norm. constant (per edge)

Images borrowed from Kipf

Spectral Domain -1D Case

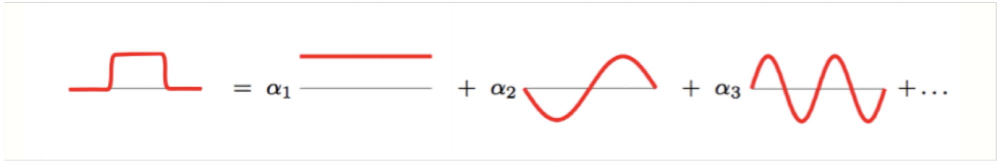
- Frequency Space
  - We can use the Fourier basis to transform the image to frequency space.
  - The Frequency space data provides us with a different representation of our Image data
  - This new representation can be used to learn different sets of features for the image data.
  - Row vectors of $\mathcal{B}$ forms the fourier basis

$$\mathcal{B} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & e^{-j\frac{2\pi}{N}(1)(1)} & e^{-j\frac{2\pi}{N}(1)(2)} & \dots & e^{-j\frac{2\pi}{N}(1)(N-1)} \\ 1 & e^{-j\frac{2\pi}{N}(2)(1)} & e^{-j\frac{2\pi}{N}(2)(2)} & \dots & e^{-j\frac{2\pi}{N}(2)(N-1)} \\ 1 & e^{-j\frac{2\pi}{N}(3)(1)} & e^{-j\frac{2\pi}{N}(3)(2)} & \dots & e^{-j\frac{2\pi}{N}(3)(N-1)} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & e^{-j\frac{2\pi}{N}(K)(1)} & e^{-j\frac{2\pi}{N}(K)(2)} & \dots & e^{-j\frac{2\pi}{N}(K)(N-1)} \end{bmatrix}$$

$$b_k[n] = e^{j\frac{2\pi}{N}kn}$$

$$\alpha[k] = \sum_n x[n]e^{-j\frac{2\pi}{N}kn}$$

$$\boxed{\alpha_k = \langle x, b_k^* \rangle}$$



Spectral Graph Theory

- Discrete Laplacian (Graphs)

Laplacian of f is given by:

$$\Delta f = \underbrace{\nabla \cdot}_{div} \underbrace{(\nabla f)}_{grad}$$

Where Δ is the laplacian operator

For Graphs the laplacian operator can be defined in terms of the incidence matrix as below:

$$\mathcal{L} f = K^T K f$$

Where K is the incidence matrix and $\mathcal{L}$ is the laplacian matrix. Therefore,

$$\mathcal{L} = K^T K$$

- Gradient in graphs is calculated by computing the differences (derivative for the discrete case) along the edges
- The divergence operator for graphs can directly be given by the incidence matrix

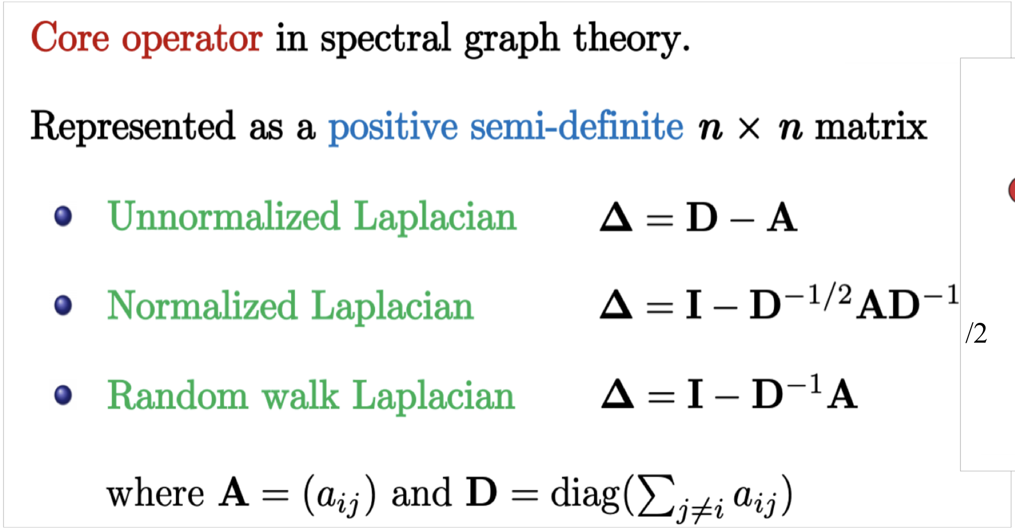
Incidence Matrix:

K is an mxn matrix where m is the number of edges and n is number of nodes/vertices

$$K_{e,v} = \begin{cases} 1 & \text{if } e = (v, w) \text{ and } v < w \\ -1 & \text{if } e = (v, w) \text{ and } v > w \\ 0 & \text{otherwise.} \end{cases}$$

Spectral Graph Theory

- Graph Laplacian

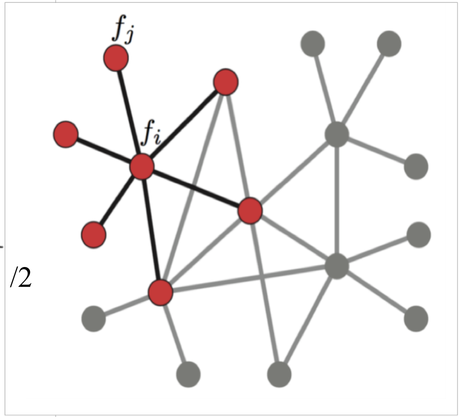


Core operator in spectral graph theory.

Represented as a positive semi-definite $n \times n$ matrix

- Unnormalized Laplacian $\quad \mathbf{\Delta} = \mathbf{D} - \mathbf{A}$
- Normalized Laplacian $\quad \mathbf{\Delta} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$
- Random walk Laplacian $\quad \mathbf{\Delta} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}$

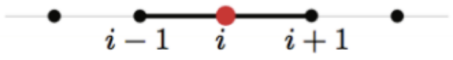
where $\mathbf{A} = (a_{ij})$ and $\mathbf{D} = \mathrm{diag}(\sum_{j \neq i} a_{ij})$



Borrowed from Yi Li

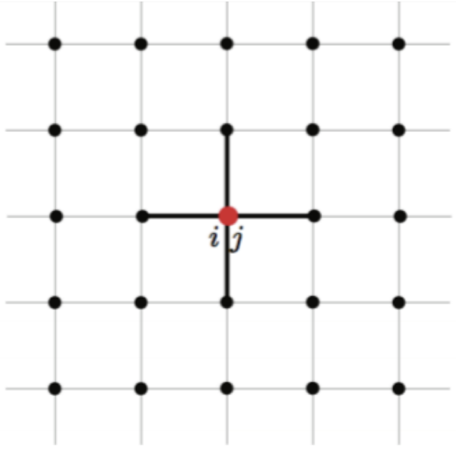
Spectral Graph Theory

- Graph Laplacian - Example



One-dimensional

$$(\Delta f)_i \approx 2f_i - f_{i-1} - f_{i+1}$$



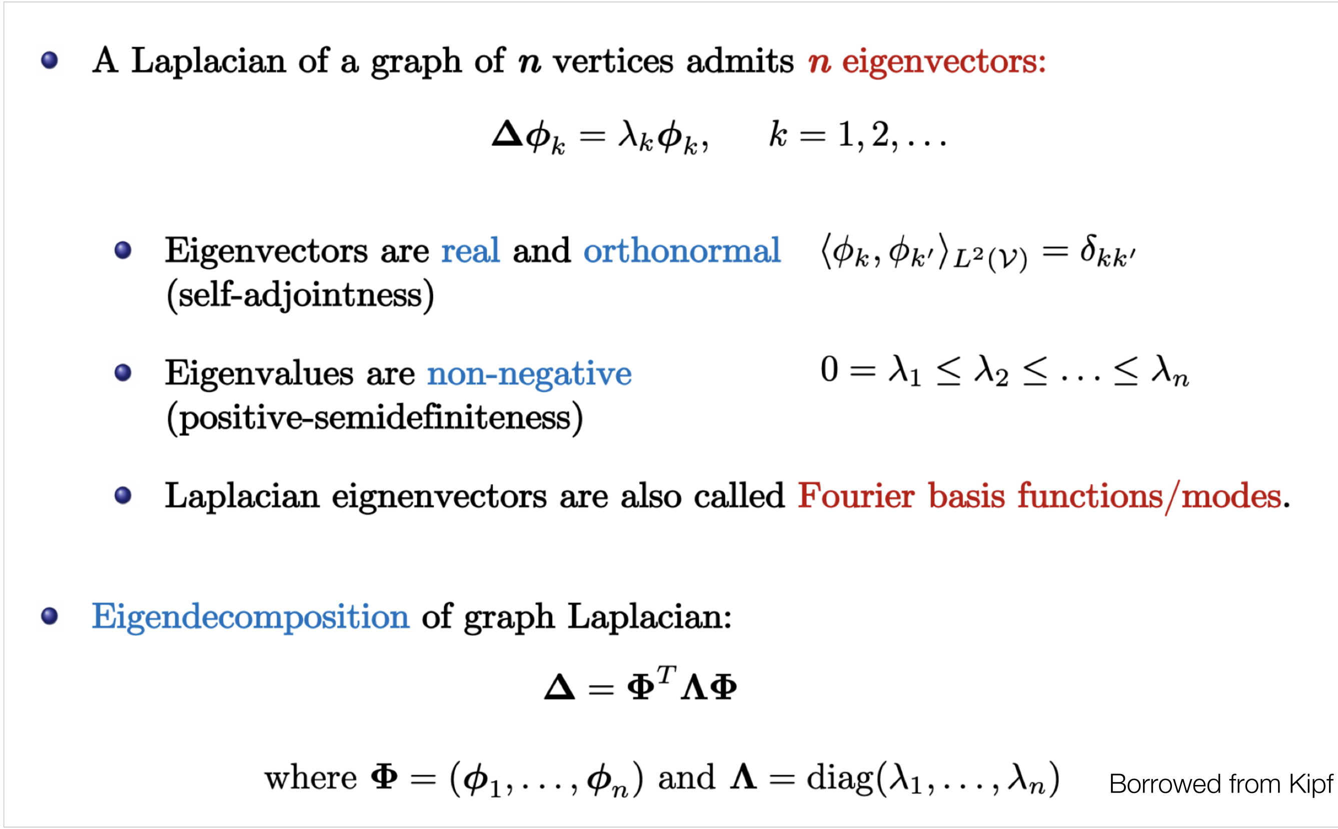
Two-dimensional

$$\begin{aligned}(\Delta f)_{ij} &\approx 4f_{ij} - f_{i-1,j} - f_{i+1,j} \\ &- f_{i,j-1} - f_{i,j+1}\end{aligned}$$

Spectral Graph Theory

- Spectral Decomposition

The laplacian eigenbasis is a generalization of the classical fourier basis to non-Euclidean domains.



- A Laplacian of a graph of n vertices admits n eigenvectors:

$$\mathbf{\Delta}\phi_k = \lambda_k \phi_k, \qquad k = 1, 2, \ldots$$

  - Eigenvectors are real and orthonormal (self-adjointness) $\langle \phi_k, \phi_{k'} \rangle_{L^2(\mathcal{V})} = \delta_{kk'}$
  - Eigenvalues are non-negative (positive-semidefiniteness) $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$
  - Laplacian eignenvectors are also called Fourier basis functions/modes.

- Eigendecomposition of graph Laplacian:

$$\mathbf{\Delta} = \mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi}$$

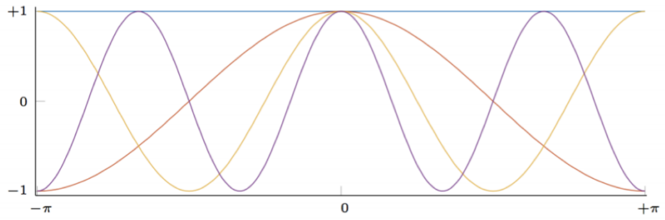
where $\mathbf{\Phi} = (\phi_1, \ldots, \phi_n)$ and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$

Borrowed from Kipf

Spectral Graph Theory

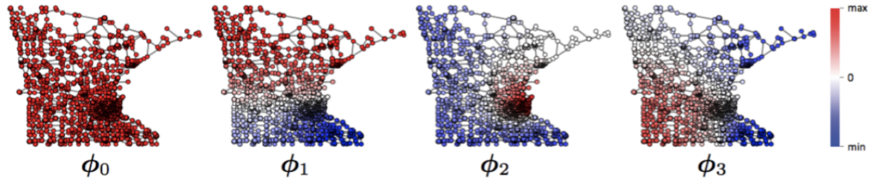
- Relation with Euclidean Spectral Analysis

- Euclidean domain:



First eigenvectors of 1D Euclidean Laplacian = standard Fourier basis

- Graph domain:



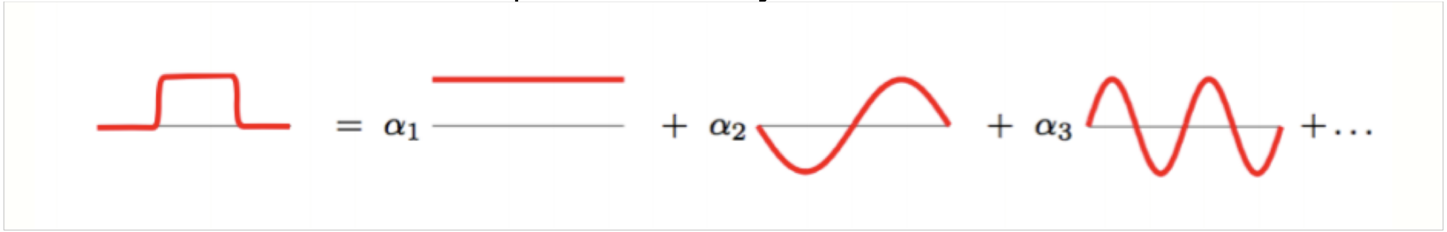
First Laplacian eigenvectors of a graph

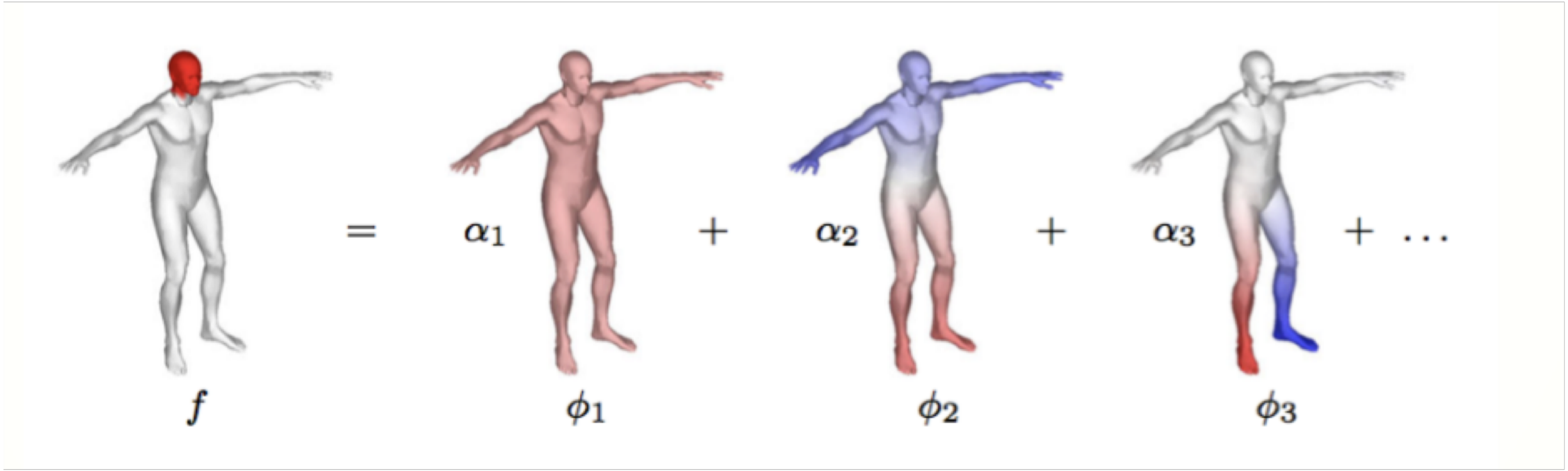
Lap eigenvectors related to graph geometry
(s.a. communities, hubs, etc), spectral clustering[10]

[10] Von Luxburg 2007

Spectral Graph Theory

- Relation with Standard Spectral Analysis

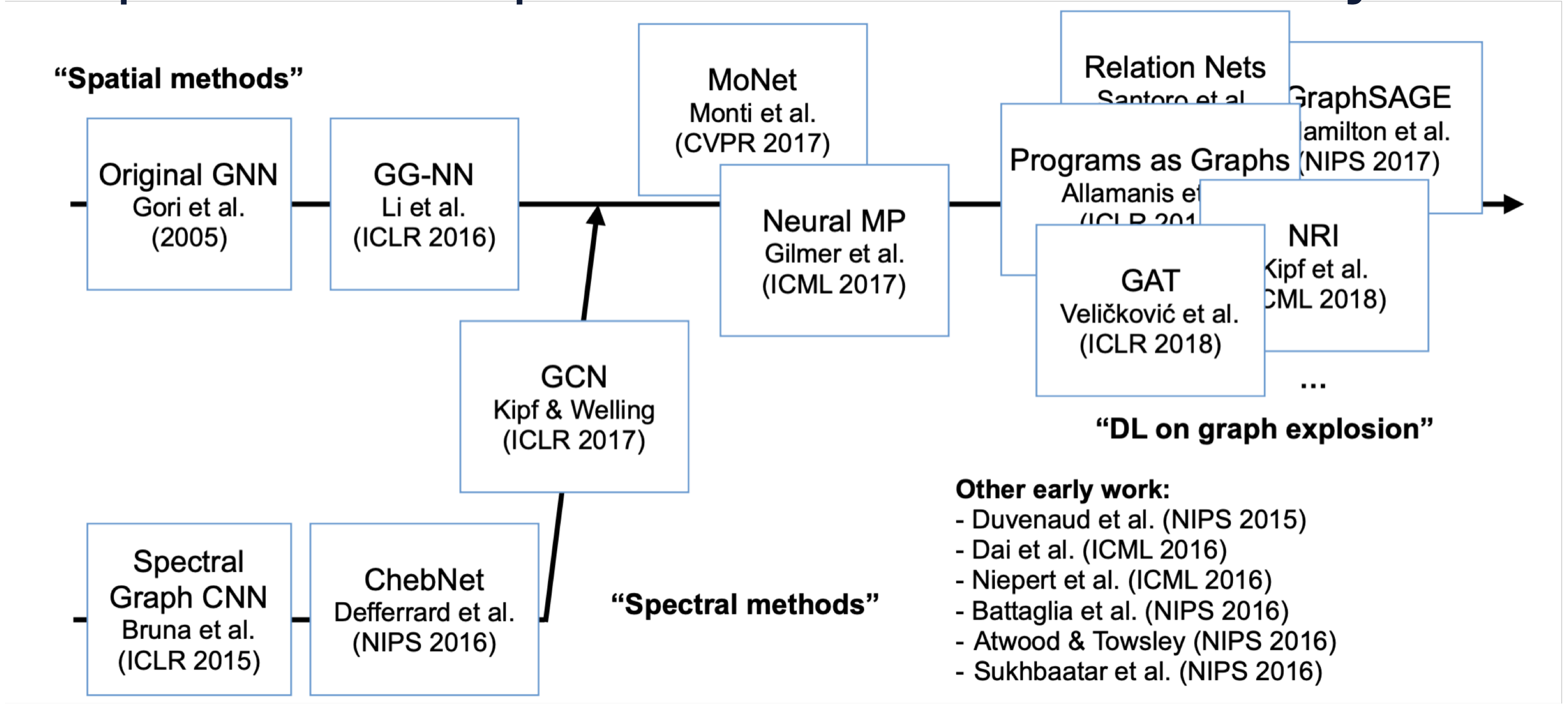




Spectral Graph Theory

- Operate on “spectrum” of a matrix representing the graph (e.g. adjacency or Laplacian matrix)
  - L = D - A
    L - Laplacian matrix
    D - Degree matrix
    A - Adjacency matrix
- Look at the eigenvectors and eigenvalues of the matrix

Spatial vs Spectral - A brief history



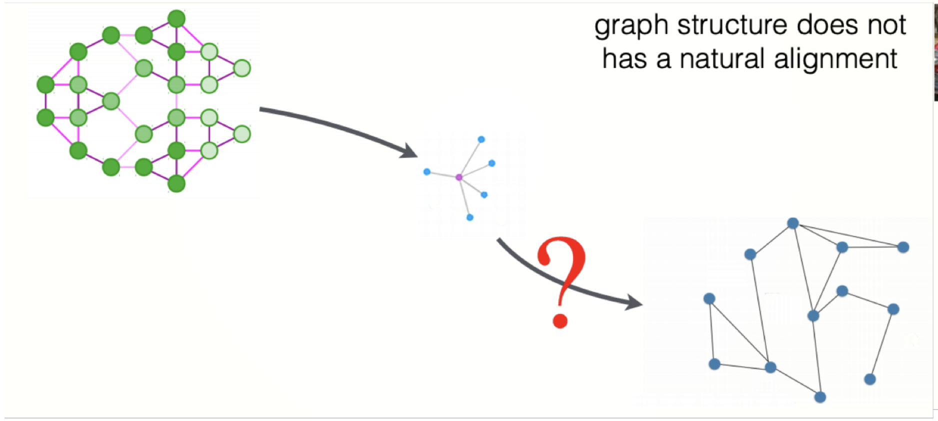
Other early work:
- Duvenaud et al. (NIPS 2015)
- Dai et al. (ICML 2016)
- Niepert et al. (ICML 2016)
- Battaglia et al. (NIPS 2016)
- Atwood & Towsley (NIPS 2016)
- Sukhbaatar et al. (NIPS 2016)

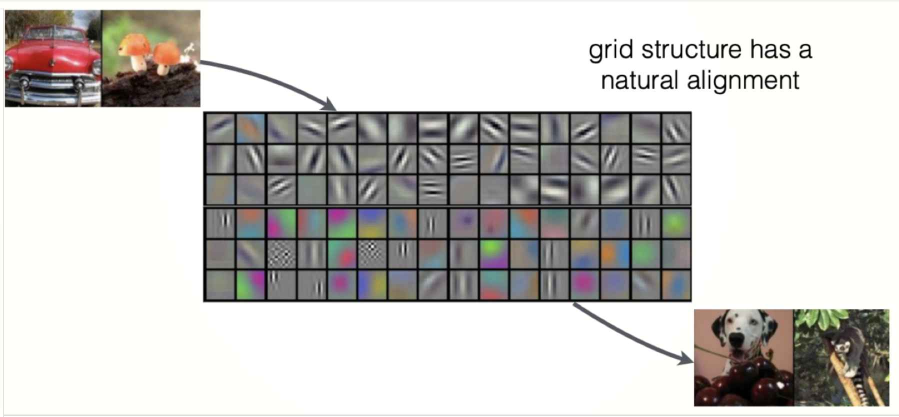
Image borrowed from Kipf

Outline

Challenges faced by Graph CNN:

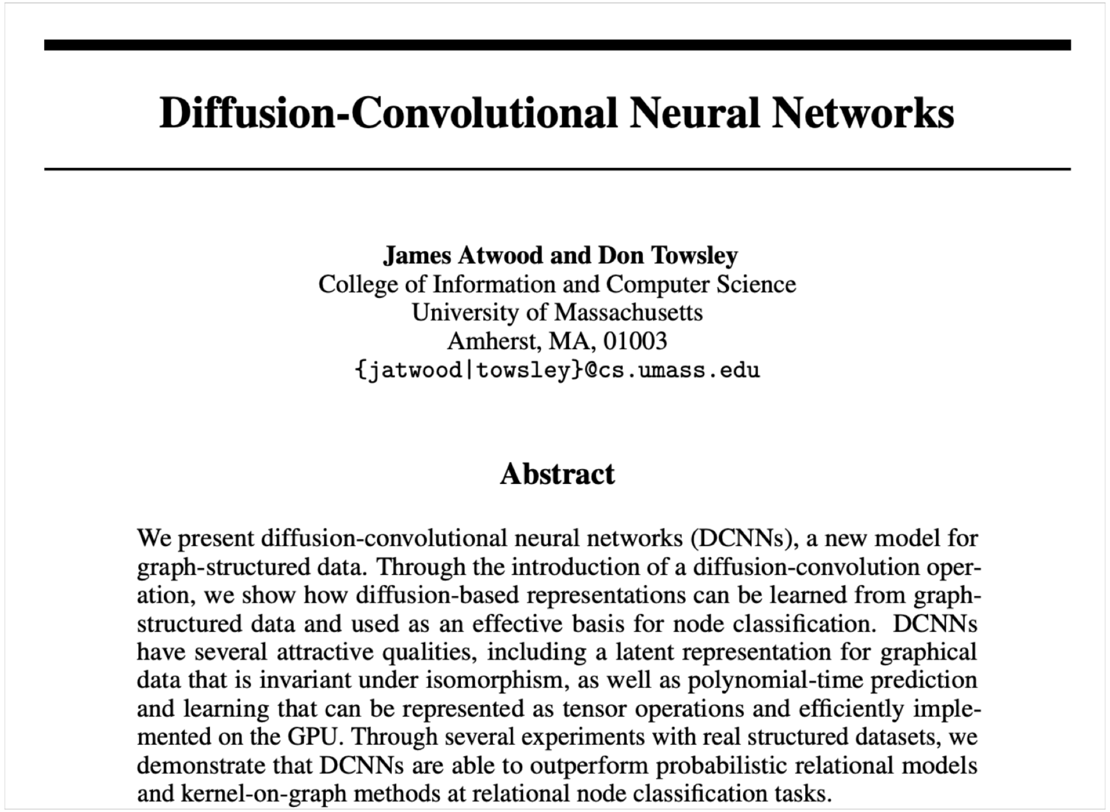
- How to define compositionality on graphs (i.e. convolution, downsampling, and pooling on graphs) ?
- How to ensure generalizability across graphs?
- And how to make them numerically fast? (as standard CNNs)





Outline

DCNN-Graph Diffusion



Diffusion-Convolutional Neural Networks

James Atwood and Don Towsley
College of Information and Computer Science
University of Massachusetts
Amherst, MA, 01003
{jatwood|towsley}@cs.umass.edu

Abstract

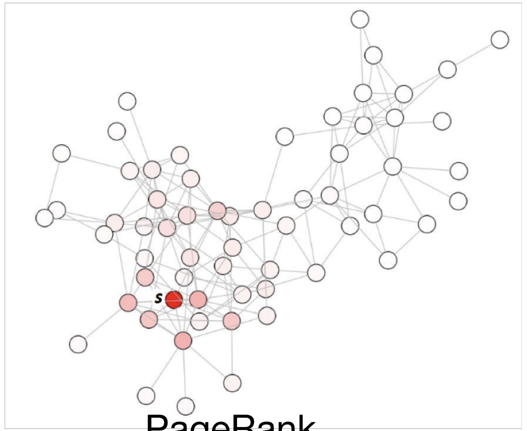
We present diffusion-convolutional neural networks (DCNNs), a new model for graph-structured data. Through the introduction of a diffusion-convolution operation, we show how diffusion-based representations can be learned from graph-structured data and used as an effective basis for node classification. DCNNs have several attractive qualities, including a latent representation for graphical data that is invariant under isomorphism, as well as polynomial-time prediction and learning that can be represented as tensor operations and efficiently implemented on the GPU. Through several experiments with real structured datasets, we demonstrate that DCNNs are able to outperform probabilistic relational models and kernel-on-graph methods at relational node classification tasks.

DCNN-Graph Diffusion

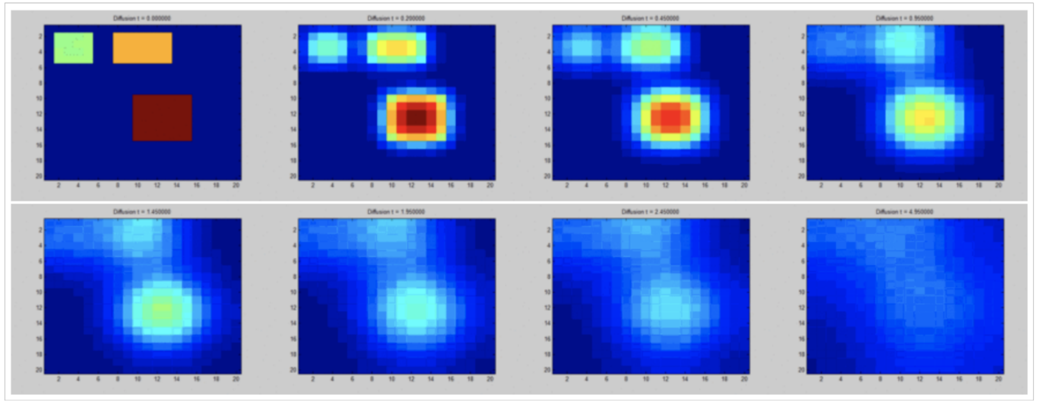
- Problems involving spreading/propagating along edges of a graph
- Information about neighbors of nodes can help classification of nodes or an entire graph

DCNN-Graph Diffusion

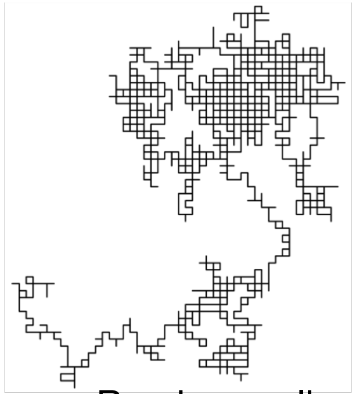
- Random walk
- PageRank
- Heat diffusion



PageRank



Heat diffusion



Random walk

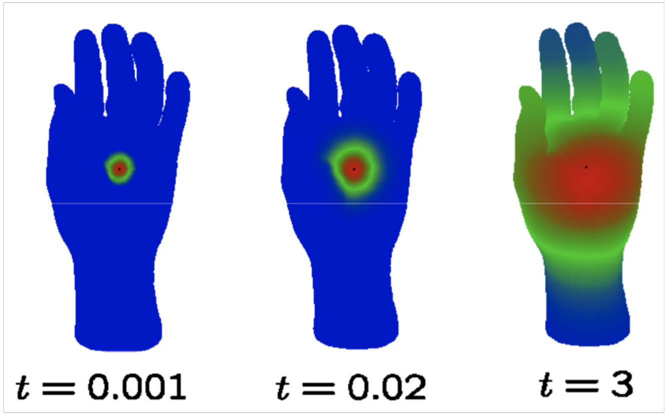
DCNN-Heat Diffusion

continuous

$$\frac{\partial u}{\partial t} - \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) = 0$$

$$\frac{\partial u}{\partial t} - \alpha \nabla^2 u = 0$$

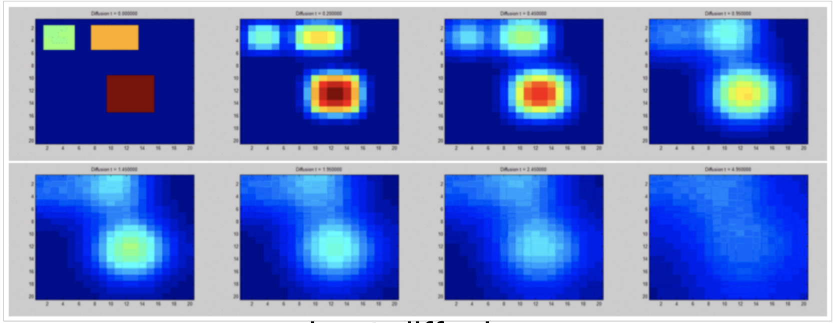
u - temperature
x,y,z - coordinates
t - time
α - thermal diffusivity



graph

$$\frac{d\phi}{dt} - k\nabla^2 \phi = 0$$

ϕ - temperature
k - heat capacity


heat diffusion

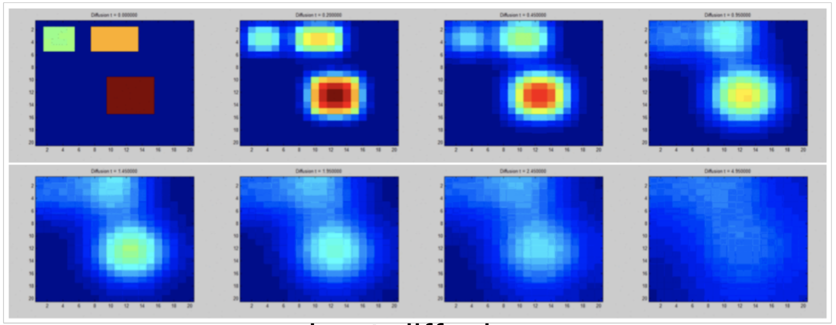
DCNN-Heat Diffusion on Graph

$$\begin{aligned}\frac{d\phi_i}{dt} &= -k\sum_j A_{ij}\left(\phi_i - \phi_j\right)\\ &= -k\left(\phi_i \sum_j A_{ij} - \sum_j A_{ij}\phi_j\right)\\ &= -k\left(\phi_i \ \deg(v_i) - \sum_j A_{ij}\phi_j\right)\\ &= -k\sum_j \left(\delta_{ij} \ \deg(v_i) - A_{ij}\right)\phi_j\\ &= -k\sum_j \left(\ell_{ij}\right)\phi_j .\end{aligned}$$

$$\begin{aligned}\frac{d\phi}{dt} &= -k(D-A)\phi\\ &= -kL\phi,\end{aligned}$$

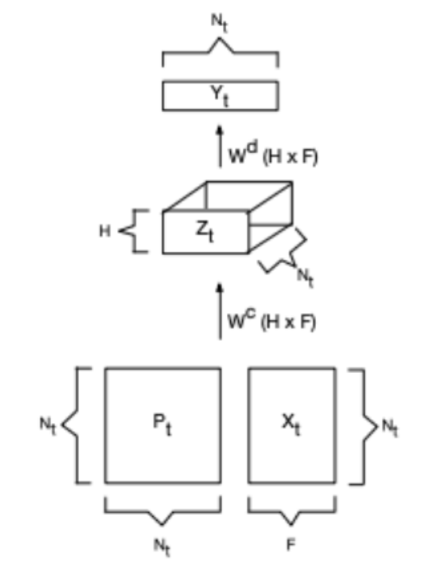
$$\frac{d\phi}{dt} - k\nabla^2\phi = 0$$

ϕ - temperature
k - heat capacity


heat diffusion

DCNN

- Diffusion-Convolutional Neural Networks
- Each node and edge in a graph is represented by some features X
- Use the power series P of the transition matrix to model diffusion
- Using the features X, power series P, and some learned weights W, we can propagate information through the graph



(a) Node classification

DCNN

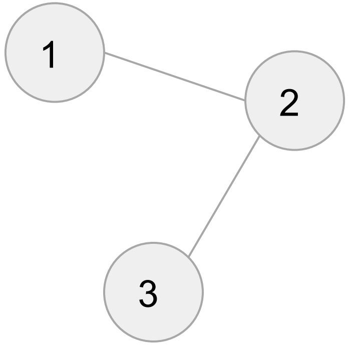
P: degree normalized transition matrix

$\mathbf{P}^* = [\mathbf{P}^1, \mathbf{P}^2, \ldots, \mathbf{P}^{H-1}]$

$P = D^{-1}A$

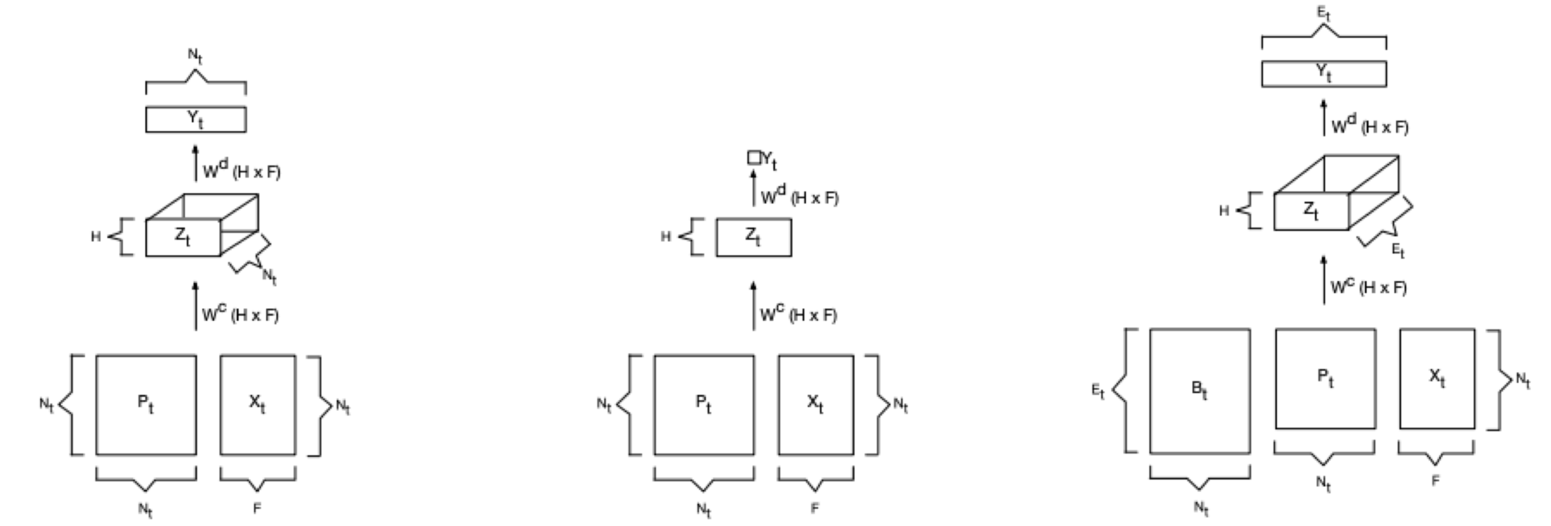
$$P^1 = \begin{matrix} [0.5 & , 0.5 & , 0. &], \\ [0.333 & , 0.333 & , 0.333 &], \\ [0. & , 0.5 & , 0.5 &] \end{matrix}$$

$$P^2 = \begin{matrix} [0.417 & , 0.417 & , 0.167 &], \\ [0.278 & , 0.444 & , 0.278 &], \\ [0.167 & , 0.417 & , 0.417 &] \end{matrix}$$



D - Degree matrix
A - Adjacency matrix
H - number of hops

DCNN



(a) Node classification (b) Graph classification (c) Edge classification

Figure 1: DCNN model definition for node, graph, and edge classification tasks.

DCNN

$$Z_t = f\left(W^c \odot P_t^* X_t\right)$$

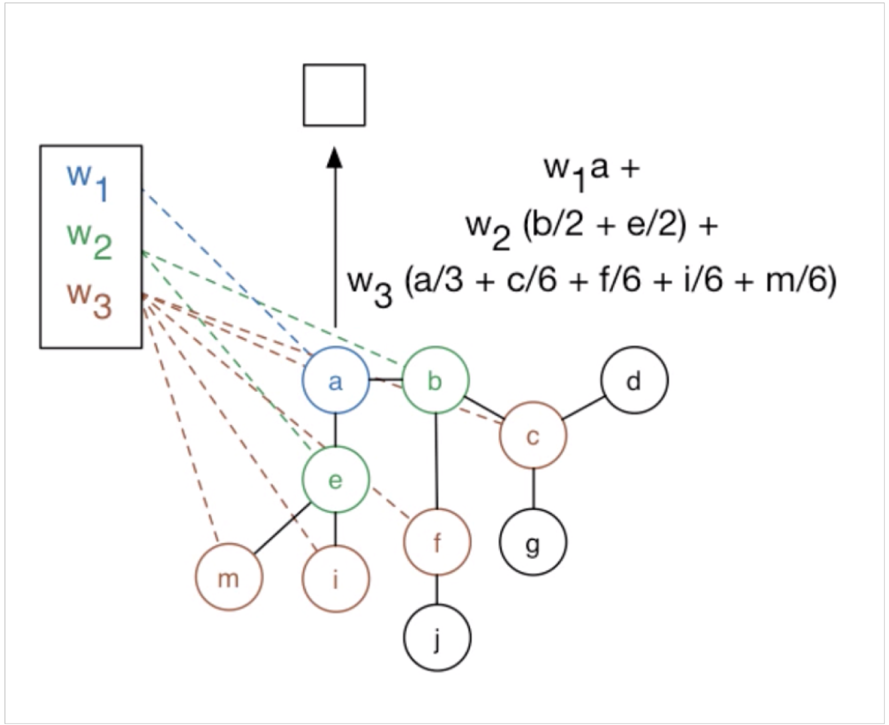
$\mathbf{Z_t}$	:	$N_t \times H \times F$	activation
$\mathbf{W_t}$	:	$H \times F$	learned weights
$\mathbf{P^*_t}$	:	$N_t \times H \times N_t$	power series
$\mathbf{X_t}$	:	$N_t \times F$	node features
f	:		nonlinear function

N_t = number of nodes
H = number of hops
F = number of features
t = graph ID

DCNN

P: degree normalized transition matrix

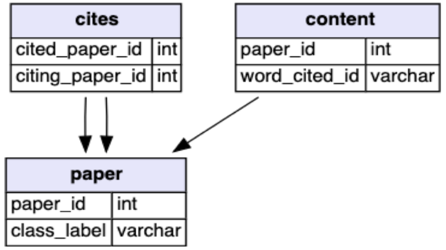
$P^* = [P^1, P^2, \ldots, P^{H-1}]$



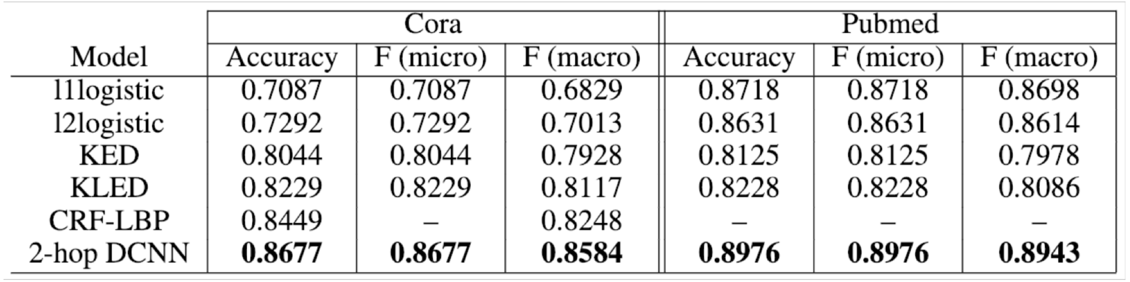
DCNN

- High performance
- Slow
  - Polynomial growth
  - $O(N_t^2F)$
- High memory usage
  - $O(N_t^2H)$
  - Works up to hundreds of thousands of nodes

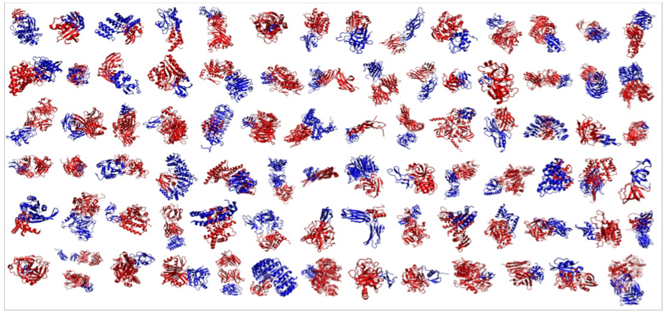
DCNN



CORA



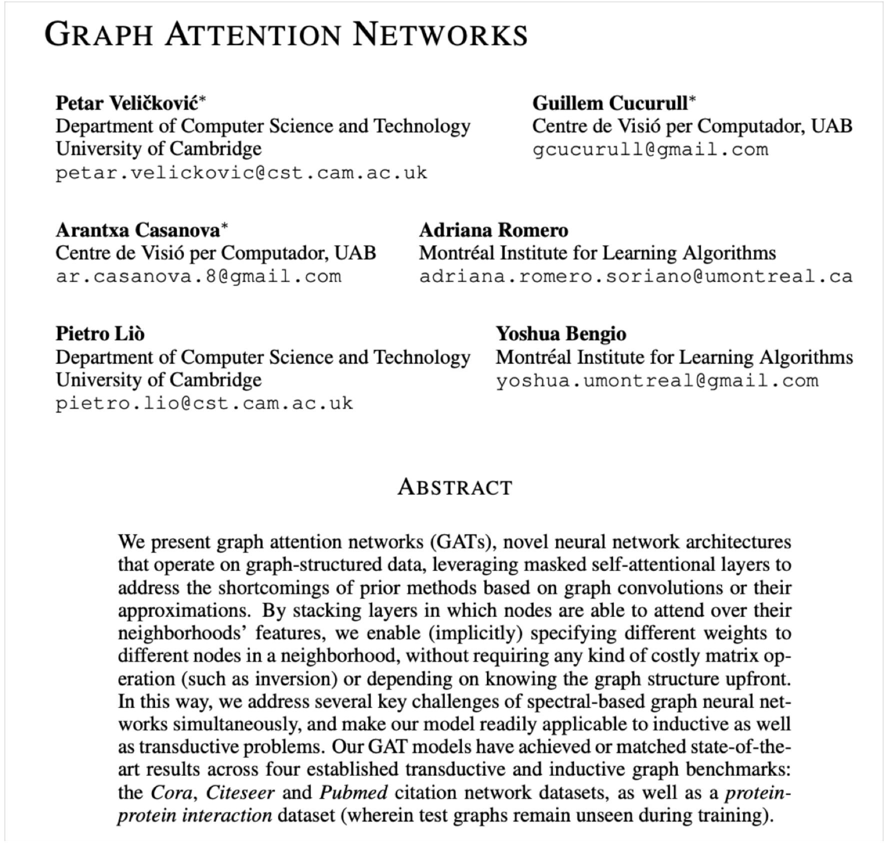
Model	Cora			Pubmed		
	Accuracy	F (micro)	F (macro)	Accuracy	F (micro)	F (macro)
l1logistic	0.7087	0.7087	0.6829	0.8718	0.8718	0.8698
l2logistic	0.7292	0.7292	0.7013	0.8631	0.8631	0.8614
KED	0.8044	0.8044	0.7928	0.8125	0.8125	0.7978
KLED	0.8229	0.8229	0.8117	0.8228	0.8228	0.8086
CRF-LBP	0.8449	–	0.8248	–	–	–
2-hop DCNN	**0.8677**	**0.8677**	**0.8584**	**0.8976**	**0.8976**	**0.8943**



PPI Affinity 2.0

Outline

Graph Attention Networks (GAT)



GRAPH ATTENTION NETWORKS

Petar Veličković*
Department of Computer Science and Technology
University of Cambridge
petar.velickovic@cst.cam.ac.uk

Guillem Cucurull*
Centre de Visió per Computador, UAB
gcucurull@gmail.com

Arantxa Casanova*
Centre de Visió per Computador, UAB
ar.casanova.8@gmail.com

Adriana Romero
Montréal Institute for Learning Algorithms
adriana.romero.soriano@umontreal.ca

Pietro Liò
Department of Computer Science and Technology
University of Cambridge
pietro.lio@cst.cam.ac.uk

Yoshua Bengio
Montréal Institute for Learning Algorithms
yoshua.umontreal@gmail.com

ABSTRACT

We present graph attention networks (GATs), novel neural network architectures that operate on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods based on graph convolutions or their approximations. By stacking layers in which nodes are able to attend over their neighborhoods' features, we enable (implicitly) specifying different weights to different nodes in a neighborhood, without requiring any kind of costly matrix operation (such as inversion) or depending on knowing the graph structure upfront. In this way, we address several key challenges of spectral-based graph neural networks simultaneously, and make our model readily applicable to inductive as well as transductive problems. Our GAT models have achieved or matched state-of-the-art results across four established transductive and inductive graph benchmarks: the *Cora*, *Citeseer* and *Pubmed* citation network datasets, as well as a *protein-protein interaction* dataset (wherein test graphs remain unseen during training).

Graph Attention Networks (GAT)

- Motivation:
  - Self attention provides powerful models for machine translation task
  - Attention networks has less computational complexity
  - Attention networks can be parallelized for faster computations.
  - Machine translation task are similar to Graphical data
  - This paper utilizes attention networks for graph structured data.

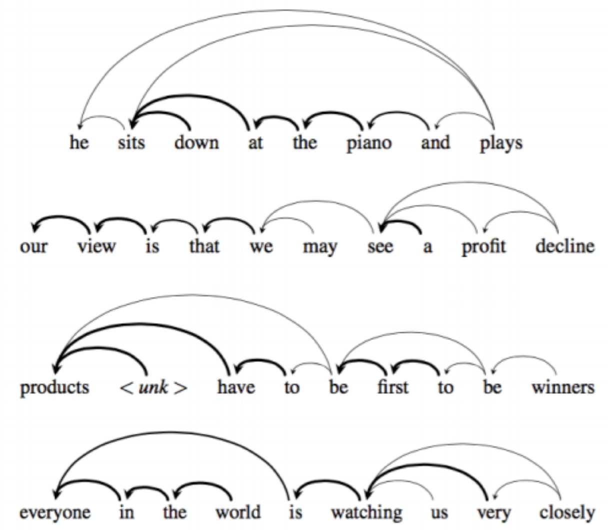

he sits down at the piano and plays
our view is that we may see a profit decline
products < unk > have to be first to be winners
everyone in the world is watching us very closely

Figure 4: Examples of intra-attention (language modeling). Bold lines indicate higher attention scores. Arrows denote which word is being focused when attention is computed, but not the direction of the relation.

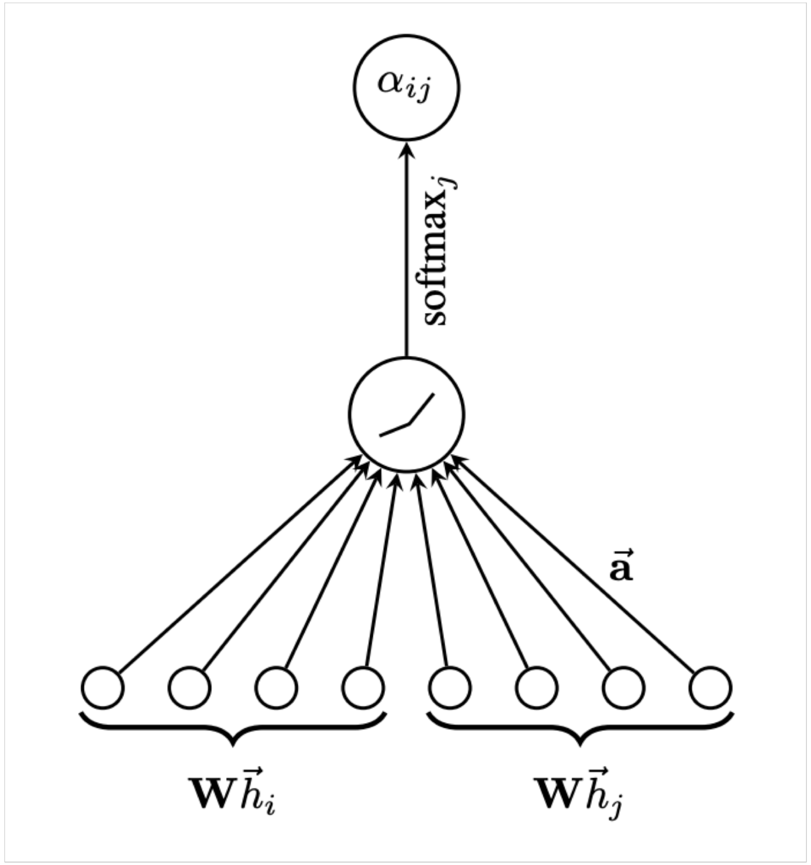
GAT ARCHITECTURE

- Input to the Attention Network is a set of node features $h \in \{\overrightarrow{h_1}, \overrightarrow{h_2} \dots \overrightarrow{h_N}\}$ where $\overrightarrow{h_i} \in \mathbb{R}^F$ and F is the number of features each node.
- A single Graph Attentional Layer produces a new set of node features $h' \in \{\overrightarrow{h'_1}, \overrightarrow{h'_2} \dots \overrightarrow{h'_N}\}$ where $\overrightarrow{h'_i} \in \mathbb{R}^{F'}$ and F' is potentially new cardinality for each node.
- A shared linear transformation, parameterized by the weight matrix $\mathbf{W} \in \mathbb{R}^{F' \times F}$ is applied to every node.
- Then a self attention mechanism is performed on these transformed features.

GAT ARCHITECTURE

- Attention Mechanism
    - a single layer feed- forward network
    - Parameterized by a weight vector $\overrightarrow{a} \in \mathbb{R}^{2F'}$
    - Masked attention is performed by computing α_{ij} for nodes $j \in$ some neighborhood of node i

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$

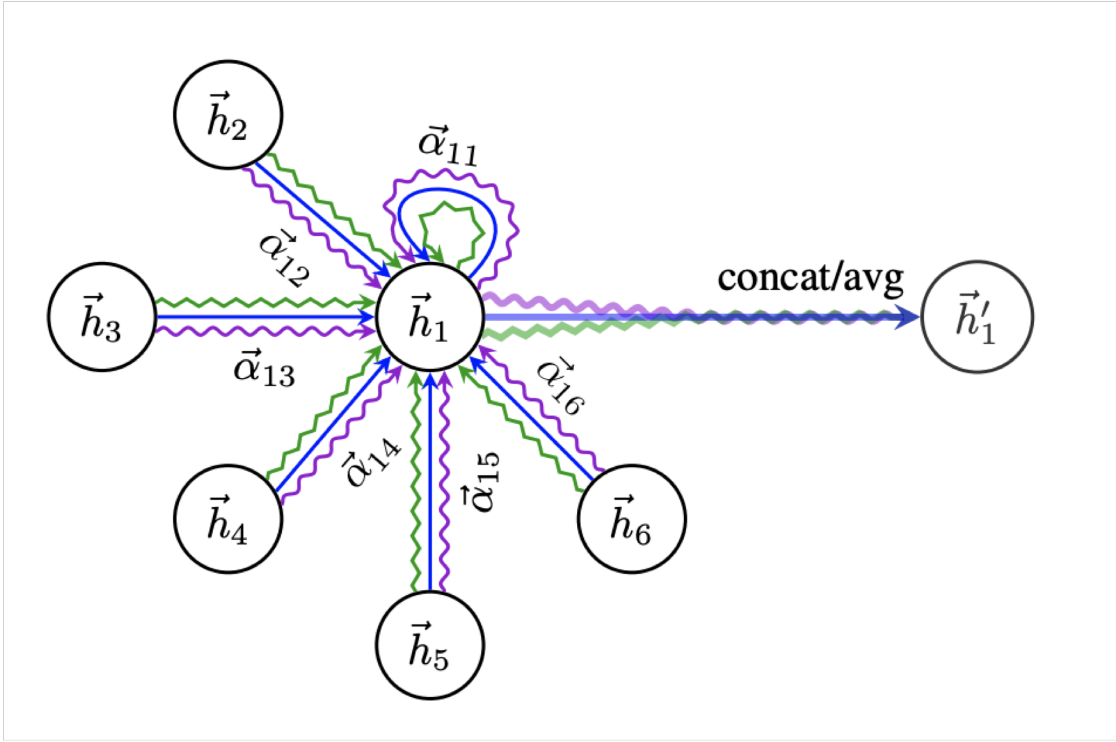


GAT ARCHITECTURE

- Multi-head Attention
  - Brings stability to the learning process
  - Parallelly executes K attention mechanisms

$$\vec{h}'_i = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{k}\mathbf{W}^k\vec{h}_j\right)$$

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{k}\mathbf{W}^k\vec{h}_j\right)$$



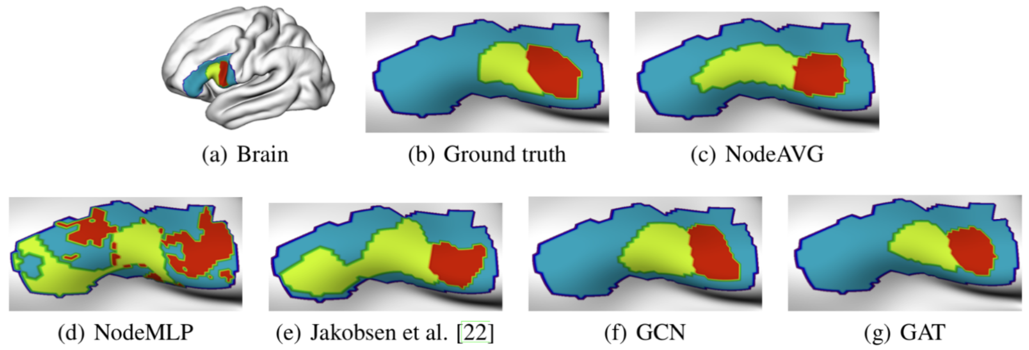
An illustration of multi-head attention with K=3 heads by node 1 on its neighbourhoods

Properties of GAT

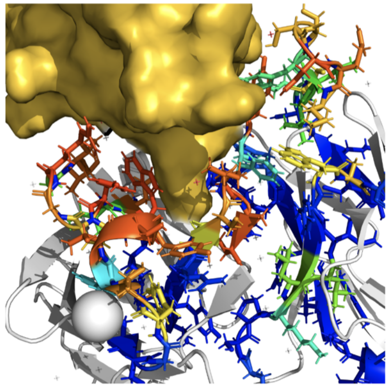
- Computational and Storage Efficiency
- Fixed number of parameters (independent of input graph size)
- Localisation - acting on a local neighbourhood of a node
- Ability to specify arbitrary importances to different neighbours
- Applicability to inductive problems

GAT Applications

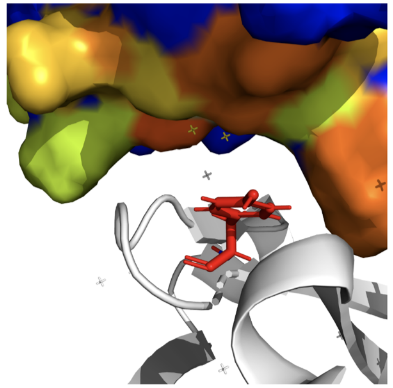
- Mesh Based Parcellation of the cerebral cortex
- Neural Paratope Prediction



Area parcellation qualitative results for several methods on a test subject. The approach of Jakobsen et al. (2016) is the prior state-of-the-art.



Antibody amino acid binding probabilities to the antigen (in gold) assigned by our model for a test antibody-antigen complex. Warmer colours indicate higher probabilities.

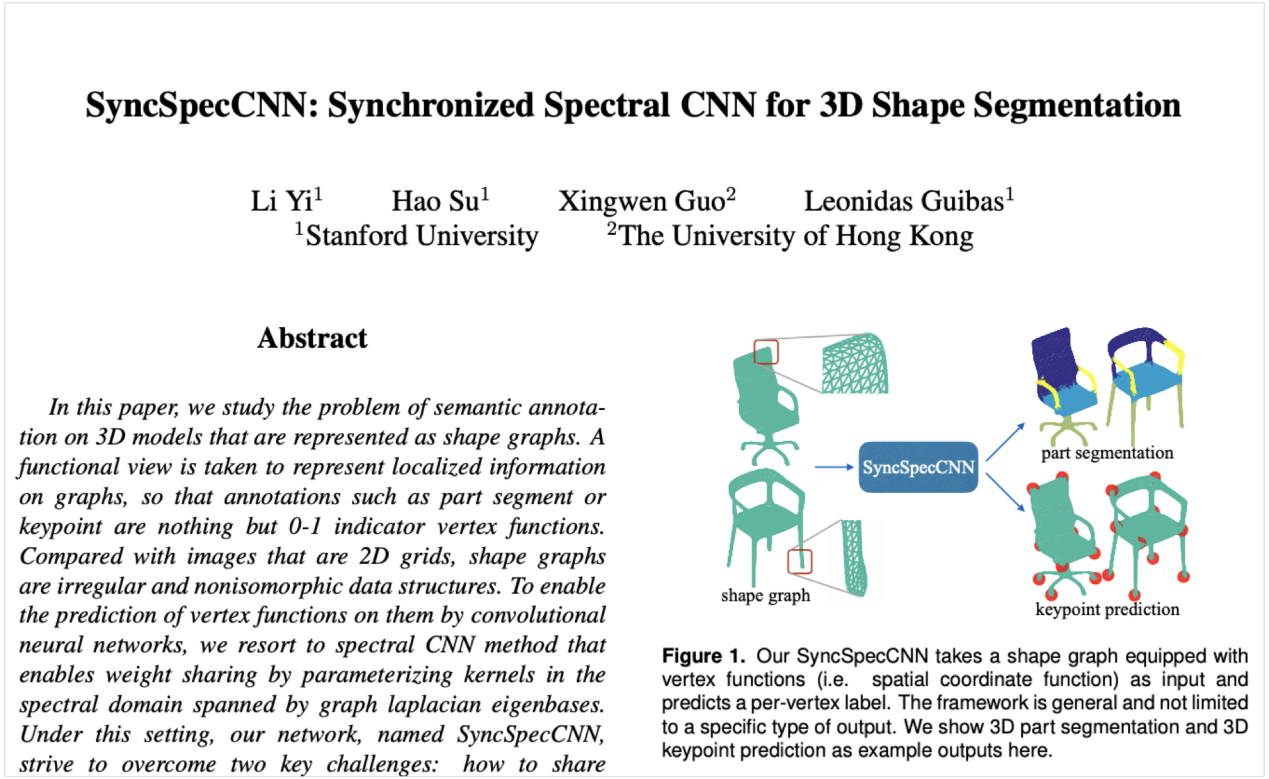


Normalised antigen attention coefficients for a single (binding) antibody amino acid (in red). Warmer colours indicate higher coefficients.

Outline

- Motivation
- Background:
  - Graph Definition
  - Spatial vs Spectral Domains
- Challenges
- Paper:
  - DCNN
  - GATs
  - SyncSpecCNN
- Summary
- Takeaway & References

SyncSpecCNN



SyncSpecCNN: Synchronized Spectral CNN for 3D Shape Segmentation

Li Yi[1] Hao Su[1] Xingwen Guo[2] Leonidas Guibas[1]
[1]Stanford University [2]The University of Hong Kong

Abstract

In this paper, we study the problem of semantic annotation on 3D models that are represented as shape graphs. A functional view is taken to represent localized information on graphs, so that annotations such as part segment or keypoint are nothing but 0-1 indicator vertex functions. Compared with images that are 2D grids, shape graphs are irregular and nonisomorphic data structures. To enable the prediction of vertex functions on them by convolutional neural networks, we resort to spectral CNN method that enables weight sharing by parameterizing kernels in the spectral domain spanned by graph laplacian eigenbases. Under this setting, our network, named SyncSpecCNN, strive to overcome two key challenges: how to share

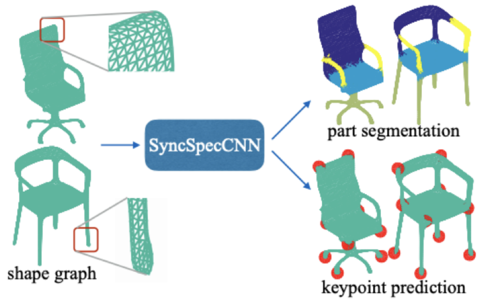
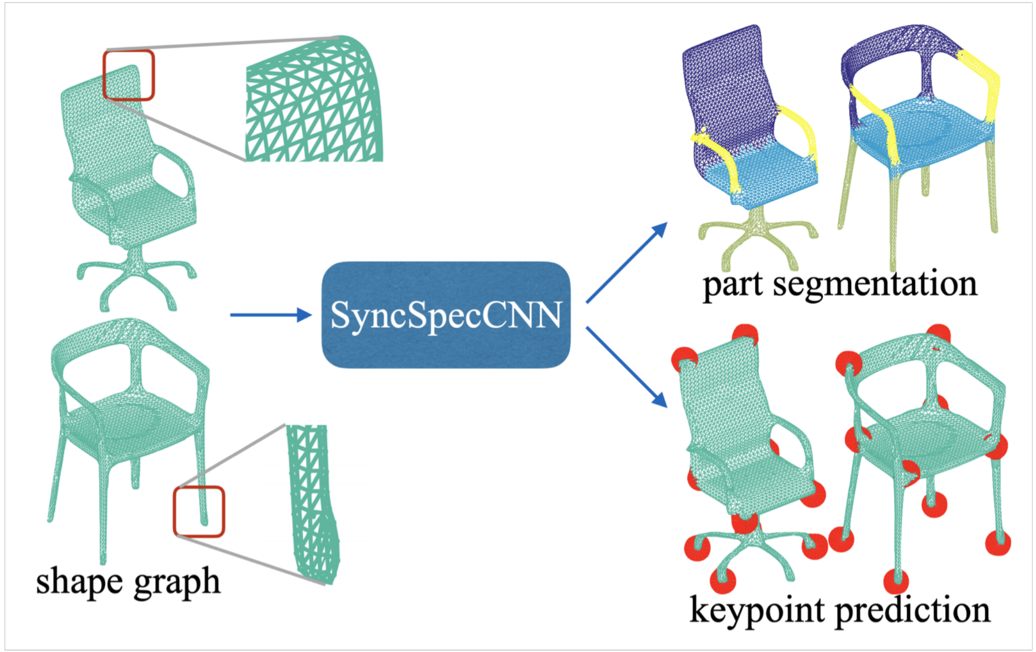


Figure 1. Our SyncSpecCNN takes a shape graph equipped with vertex functions (i.e. spatial coordinate function) as input and predicts a per-vertex label. The framework is general and not limited to a specific type of output. We show 3D part segmentation and 3D keypoint prediction as example outputs here.

Yi, L., Su, H., Guo, X., Guibas, L. (2017)

SyncSpecCNN

- Problem:
  - Input:
    - Shape Graph (L)
    - Vertex Functions
  - Output:
    - Vertex Label

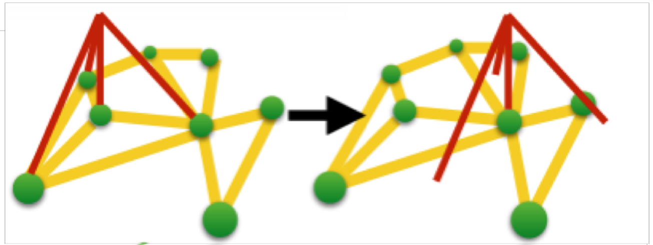


Yi, L., Su, H., Guo, X., Guibas, L. (2017)

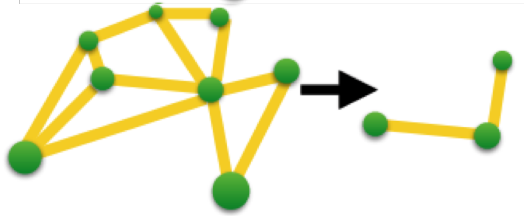
SyncSpecCNN

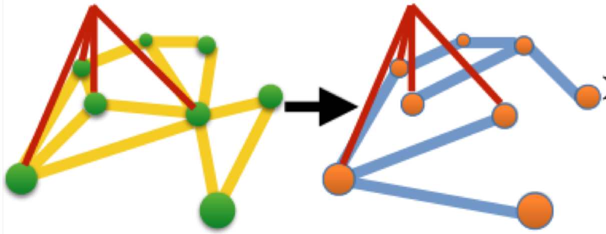
- Challenges:

➢ Lack of a translation structure, which is the key of using convolution filters and allowing weight sharing



➢ It is not trivial to downsample a graph, setting challenge for effective multi-scale analysis.





➢ Connectivity between different graphs varies a lot. Filters learned for one graph cannot easily generalize to new graphs.

Yi, L., Su, H., Guo, X., Guibas, L. (2017)

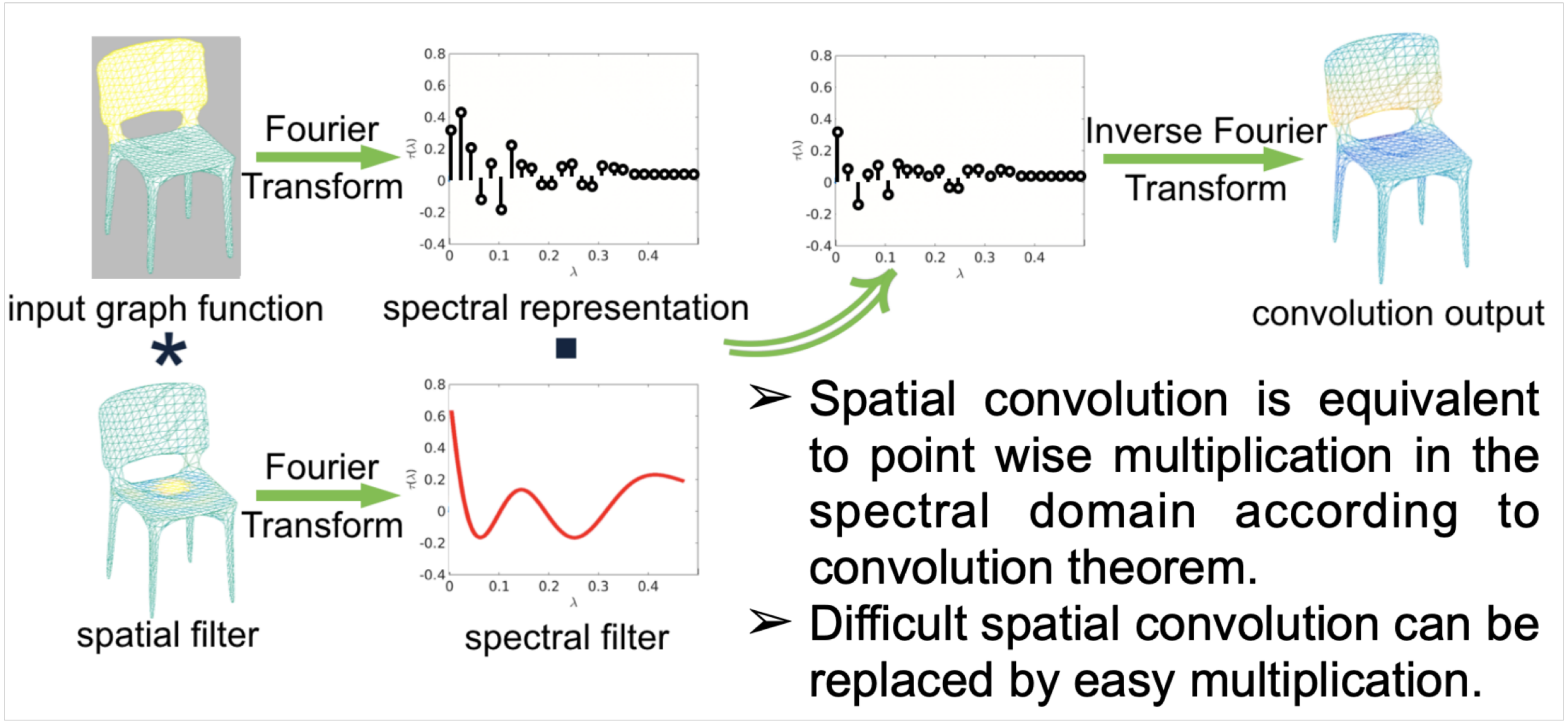
SyncSpecCNN

- Solutions:
  - Lack of translation structure
    ⇨ Spatial convolution = point-wise multiplication in spectral domain.
  - Hard to do effective multi-scale analysis
    ⇨ Spectral dilated convolution
  - Non-isometric shapes does not generalize well
    ⇨ Functional map for synchronization

SyncSpecCNN

- Solutions:
  - Lack of translation structure
    ⇨ Spatial convolution = point-wise multiplication in spectral domain.
  - Hard to do effective multi-scale analysis
    ⇨ Spectral dilated convolution
  - Non-isometric shapes does not generalize well
    ⇨ Functional map for synchronization

SyncSpecCNN



- ➢ Spatial convolution is equivalent to point wise multiplication in the spectral domain according to convolution theorem.
- ➢ Difficult spatial convolution can be replaced by easy multiplication.

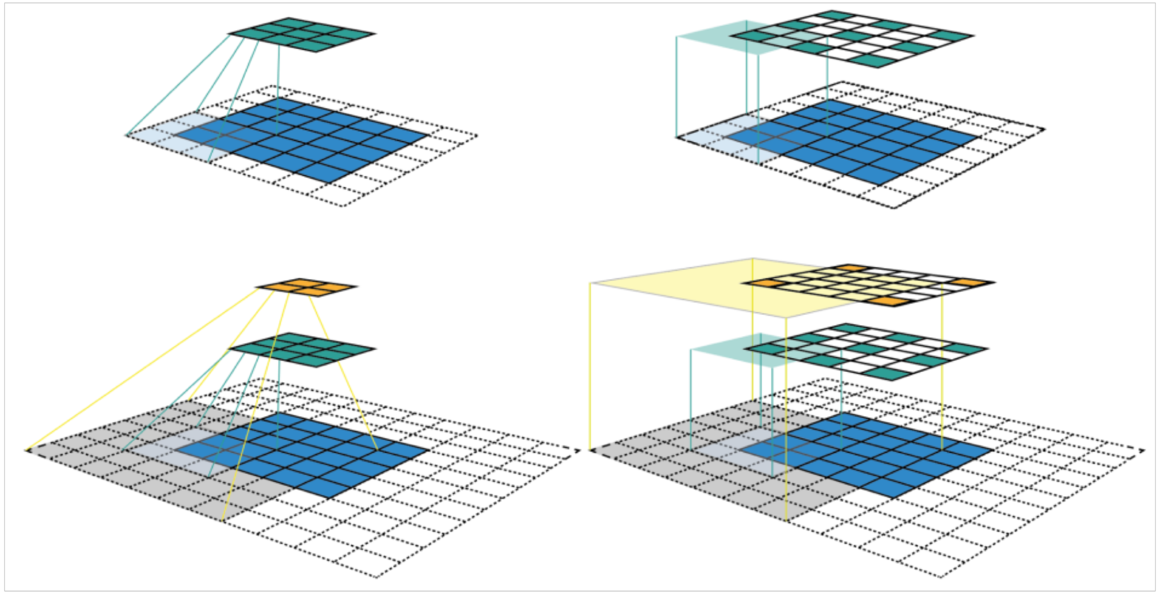
[Yi, L., Su, H., Guo, X., Guibas, L. (2017)]

SyncSpecCNN

- Solutions:
  - Lack of translation structure
    ⇨ Spatial convolution = point-wise multiplication in spectral domain.
  - Hard to do effective multi-scale analysis
    ⇨ Spectral dilated convolution
  - Non-isometric shapes does not generalize well
    ⇨ Functional map for synchronization

SyncSpecCNN

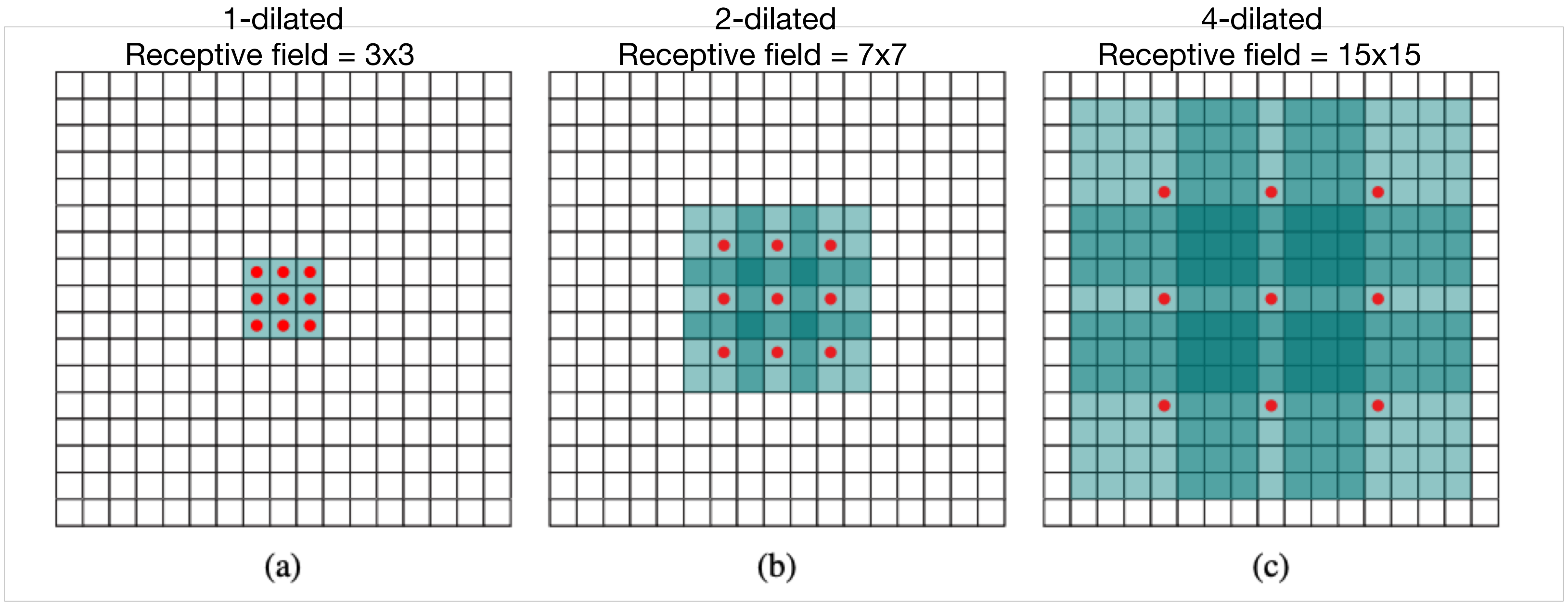
- Receptive Field Recap for Images



by Dang Ha The Hien

SyncSpecCNN

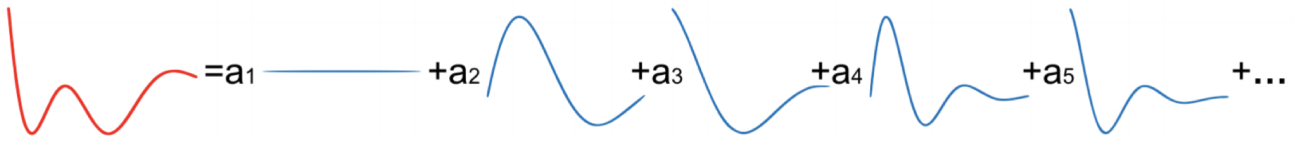
- Dilated Convolution in 2D



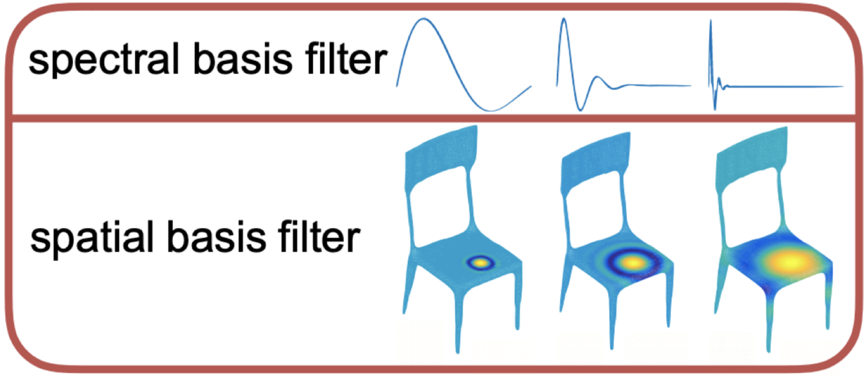
Yu, Fisher & Koltun, V (2016)

SyncSpecCNN

- Spectral Dilated Convolution
  - Parameterize spectral filters as linear combinations of basis filters



  - Control the spatial support of spectral filters through adjusting the basis filter bandwidth
  - The smaller the bandwidth is, the less smoother the filter is, the larger spatial support it will have



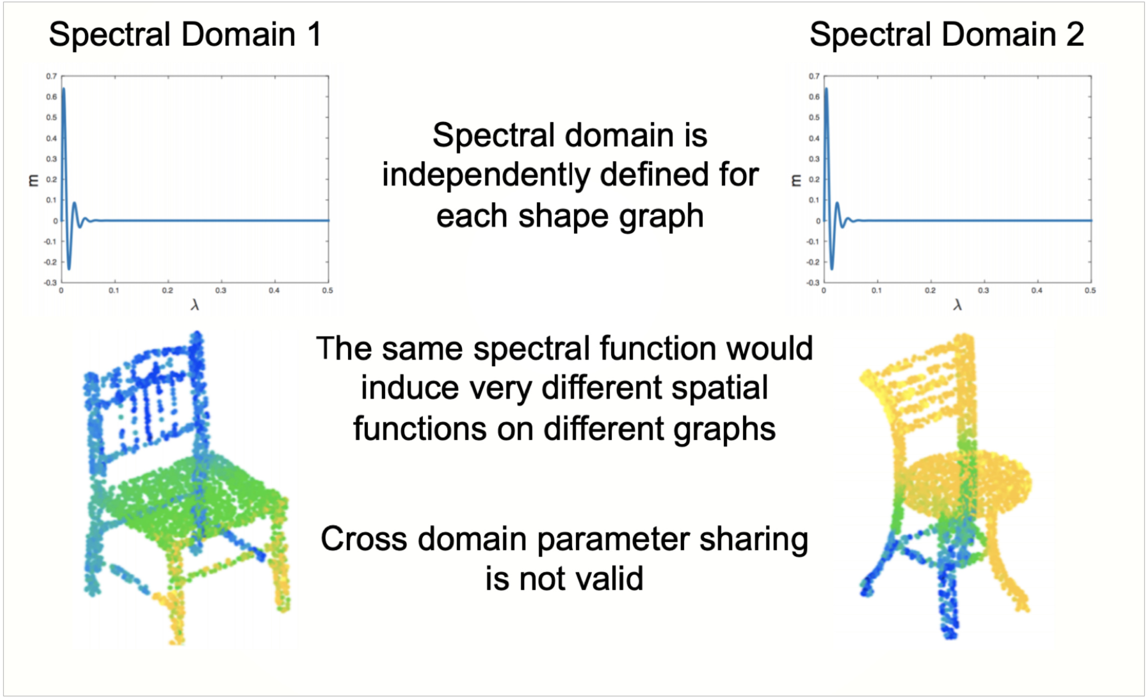
[Yi, L., Su, H., Guo, X., Guibas, L. (2017)](#)

SyncSpecCNN

- Solutions:
  - Lack of translation structure
    ⇨ Spatial convolution = point-wise multiplication in spectral domain.
  - Hard to do effective multi-scale analysis
    ⇨ Spectral dilated convolution
  - Non-isometric shapes does not generalize well
    ⇨ Functional map for synchronization

SyncSpecCNN

- Cross Domain Consistency



Spectral domain is independently defined for each shape graph

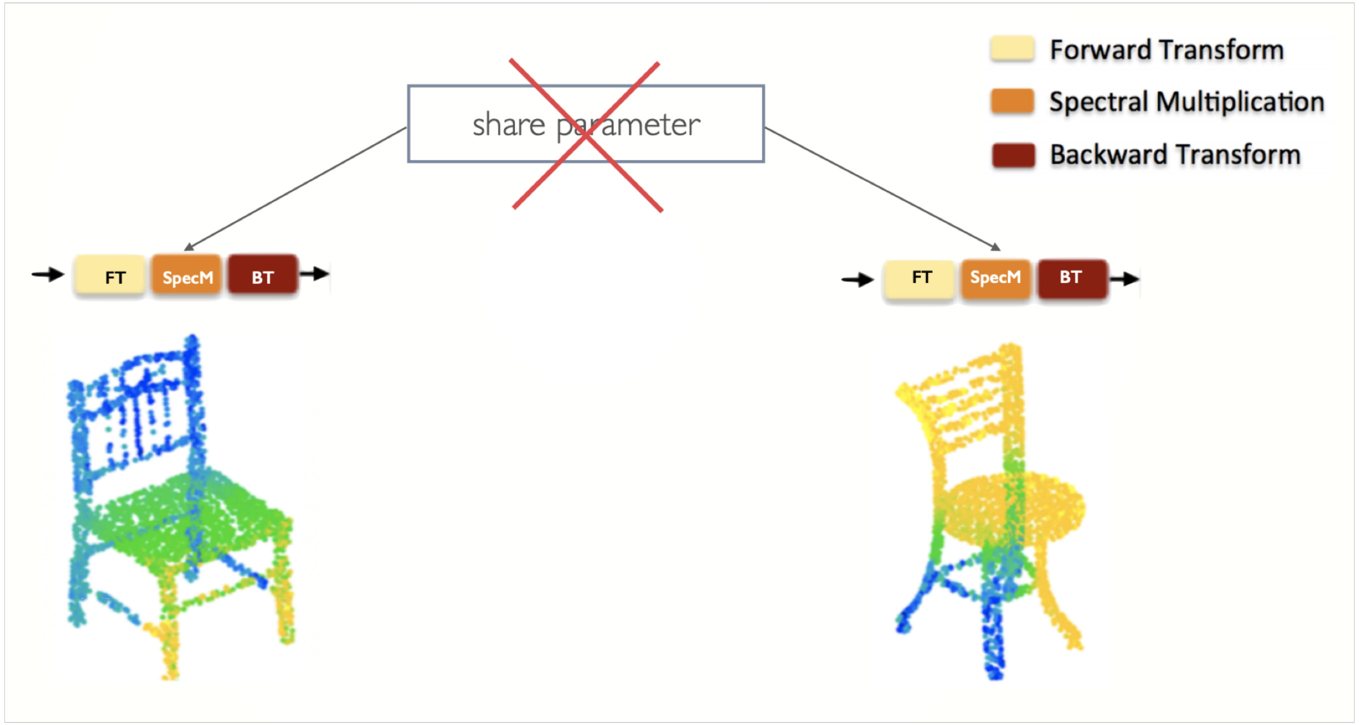
The same spectral function would induce very different spatial functions on different graphs

Cross domain parameter sharing is not valid

[Yi, L., Su, H., Guo, X., Guibas, L. (2017)](#)

SyncSpecCNN

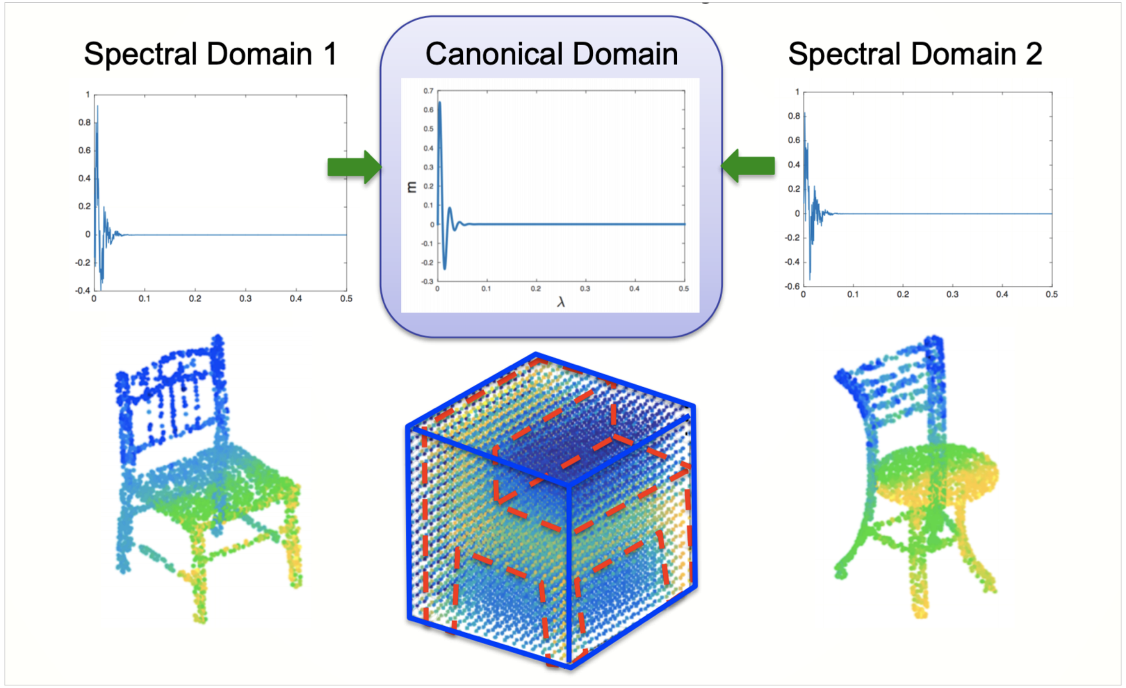
- Cross Domain Consistency



[Yi, L., Su, H., Guo, X., Guibas, L. (2017)]

SyncSpecCNN

- Different domains need to be synchronized.



Yi, L., Su, H., Guo, X., Guibas, L. (2017)

SyncSpecCNN

- Functional Map

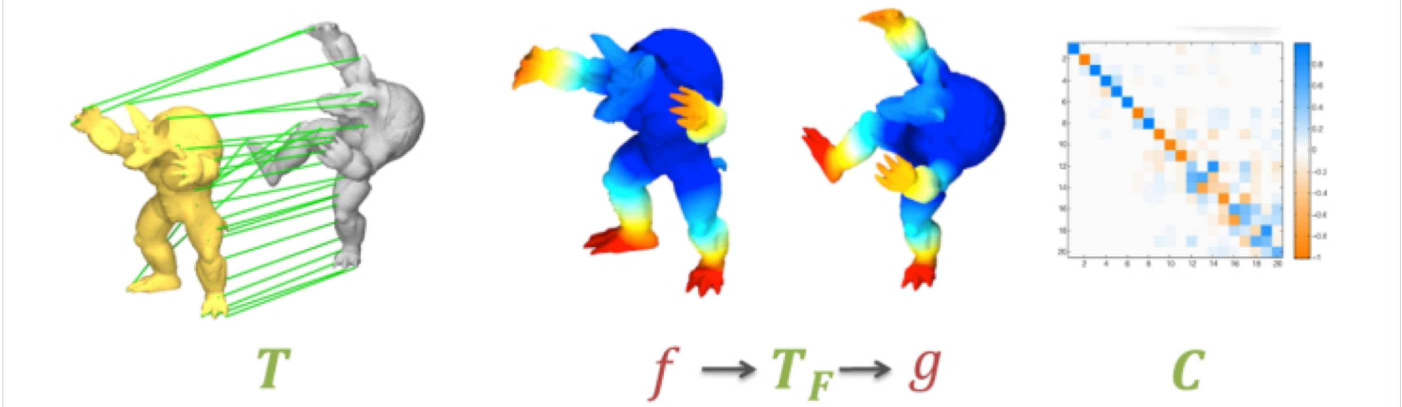
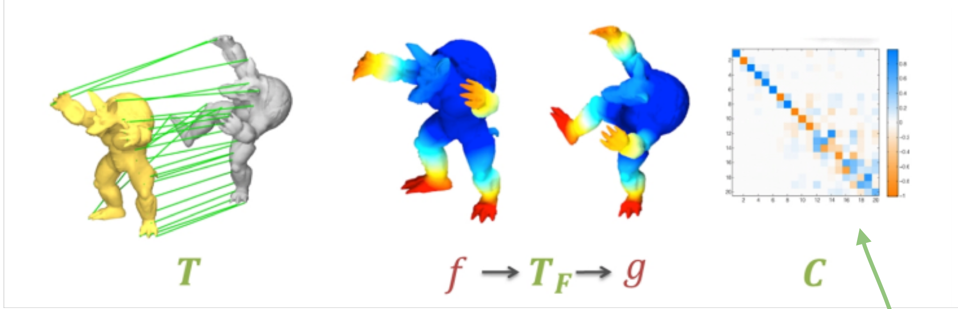


Figure 2.1: A point to point map $T : M \rightarrow N$ can be represented as a *functional map* T_F: a correspondence between functions $f : M \rightarrow \mathbb{R}$ and functions $g : N \rightarrow \mathbb{R}$. Given a choice of basis for functions on M and N, the functional map can be concisely represented as a matrix C.

SIGGRAPH2014 Course Notes

SyncSpecCNN

- Functional Map
  - Let $g : N \to \mathbb{R}$ $\quad g = \sum_i b_i \phi_i^N$
    $f : M \to \mathbb{R}$ $\quad f = \sum_i a_i \phi_i^M$
  - Original Mapping: $T : M \to N$
  - Functional representation of T is: $T_F : \mathcal{F}(M, \mathbb{R}) \to \mathcal{F}(N\mathbb{R})$
  - While T maybe a complicated mapping between surfaces, T_F acts linearly between function spaces.



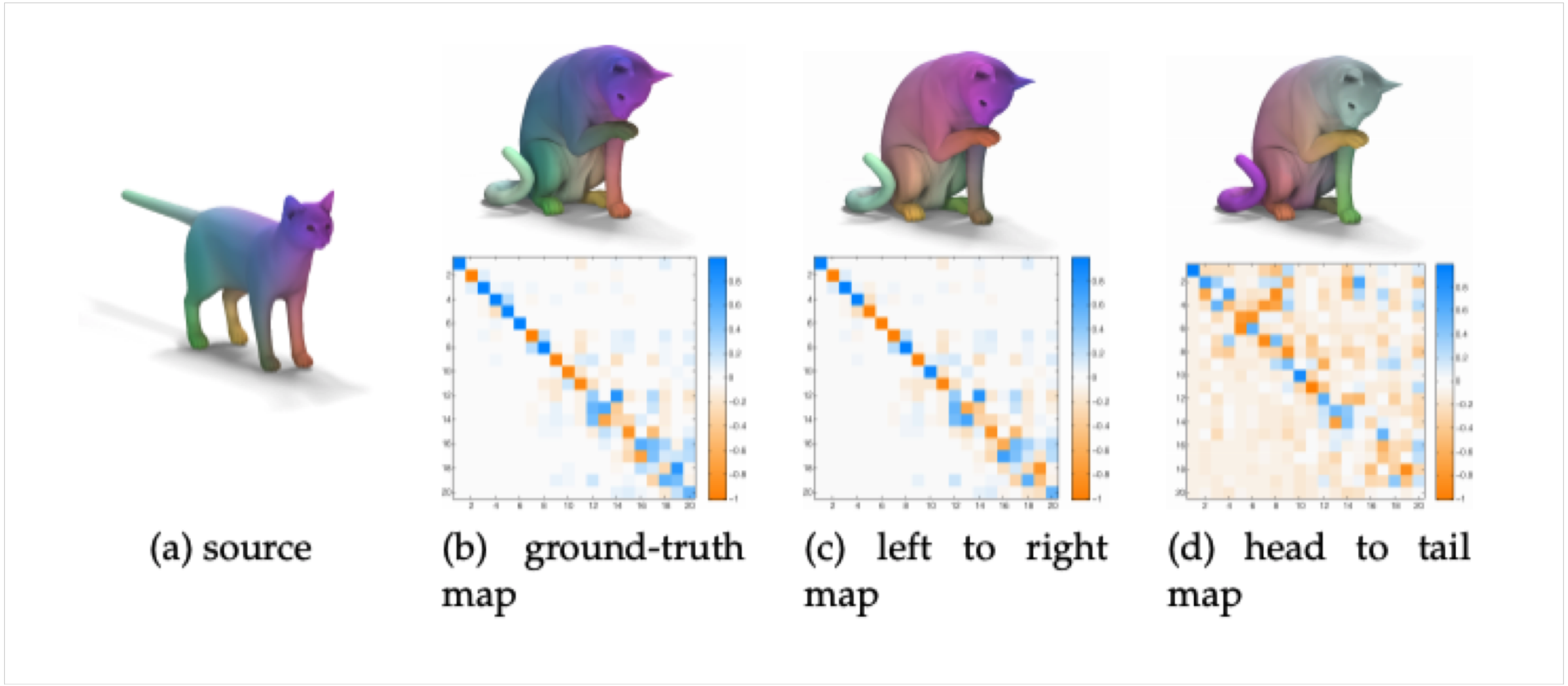
$$g = T_F(f) = T_F(\sum_i a_i \phi_i^M) = \sum_i a_i T_F(\phi_i^M) = \sum_i a_i \sum_j c_j \phi_j^N = \sum_j \sum_i a_i c_j \phi_j^N$$

$$c_{i,j} = < T_F(\phi_i^M), \phi_j^N >$$

SIGGRAPH2014 Course Notes

SyncSpecCNN

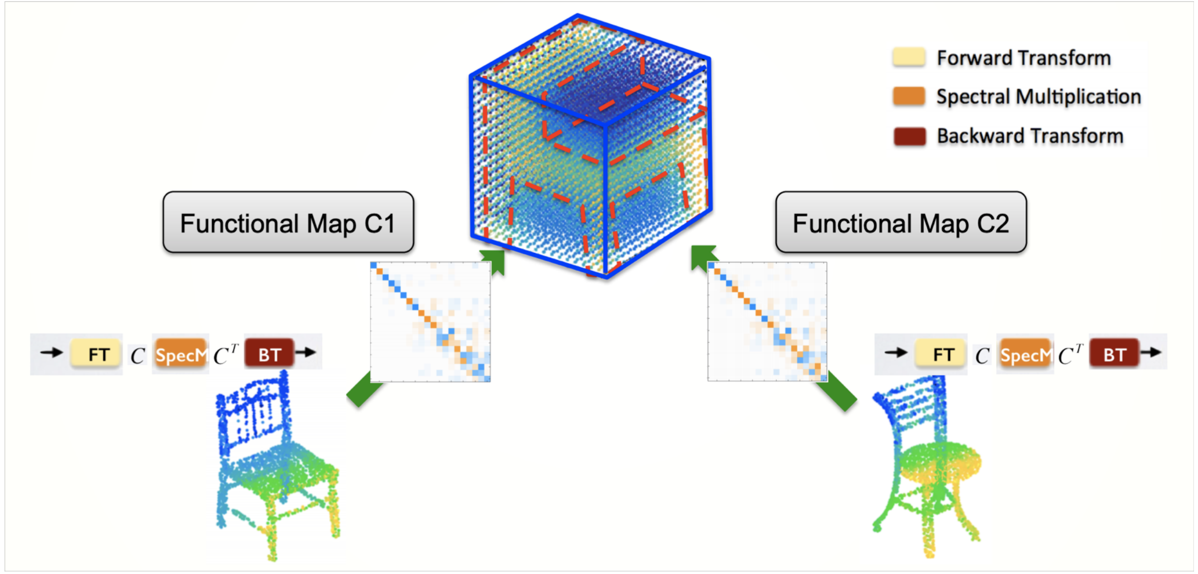
- Functional Map



SIGGRAPH2014 Course Notes

SyncSpecCNN

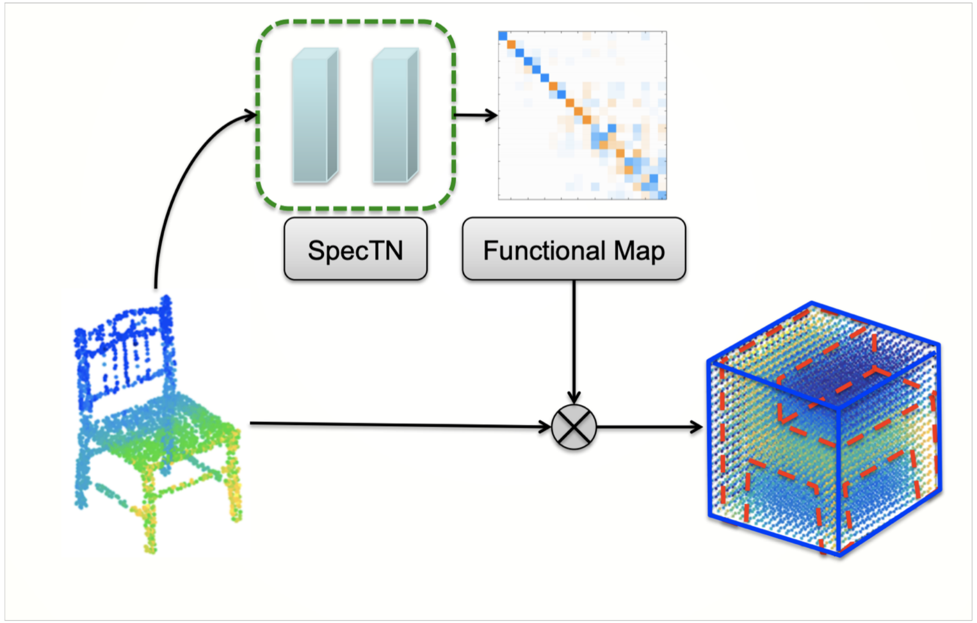
- Functional map for domain synchronization
  - Use linear maps to align individual graph bases with bases from a canonical domain, for better generalizability of spectral filters across domains.



Yi, L., Su, H., Guo, X., Guibas, L. (2017)

SyncSpecCNN

- Spectral Transformer Network
    - SpecTN is used to learn the linear map. SpecTN is trained together with the end task



Yi, L., Su, H., Guo, X., Guibas, L. (2017)

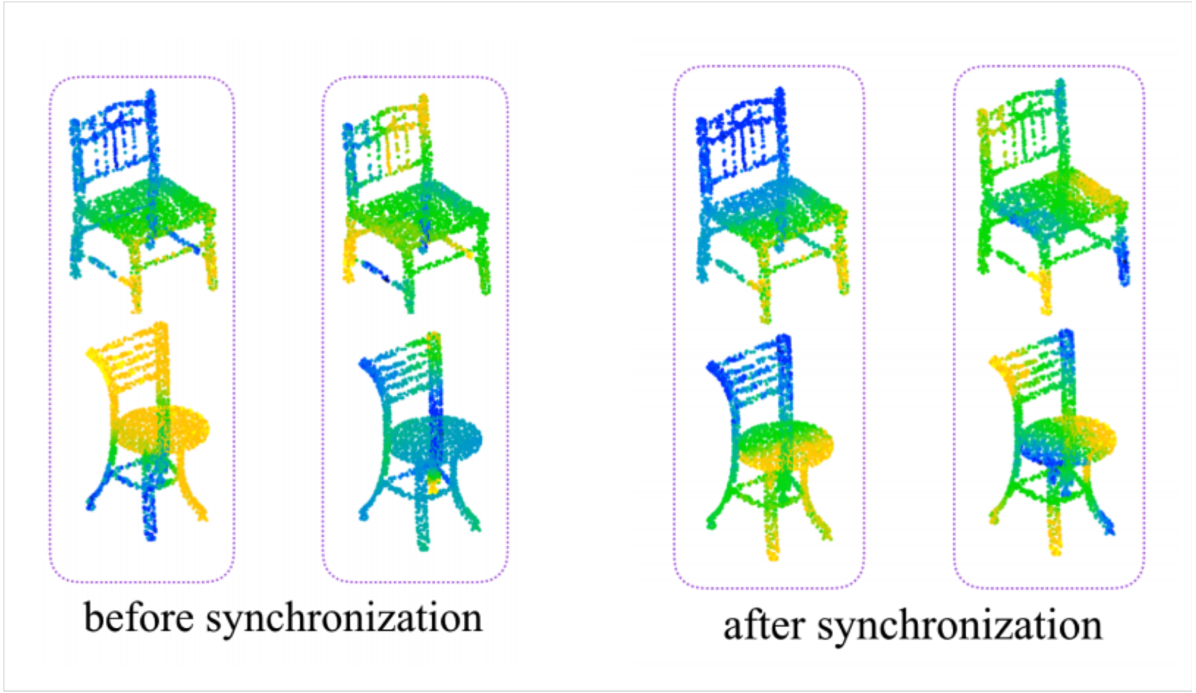
SyncSpecCNN

- Spectral Transformer Network
  - Generates high dimensional transformation, sensitive to initialization. (15 x 45 matrix)
    $C_{pre,i}$
  - Pre-trained to get a good starting point
    $minimize_{\Theta} \sum_i ||SpecTN(B_{v,i} : \Theta) - C_{pre,i}||^2$
  - Fine tuned with the end task learning

Yi, L., Su, H., Guo, X., Guibas, L. (2017)

SyncSpecCNN

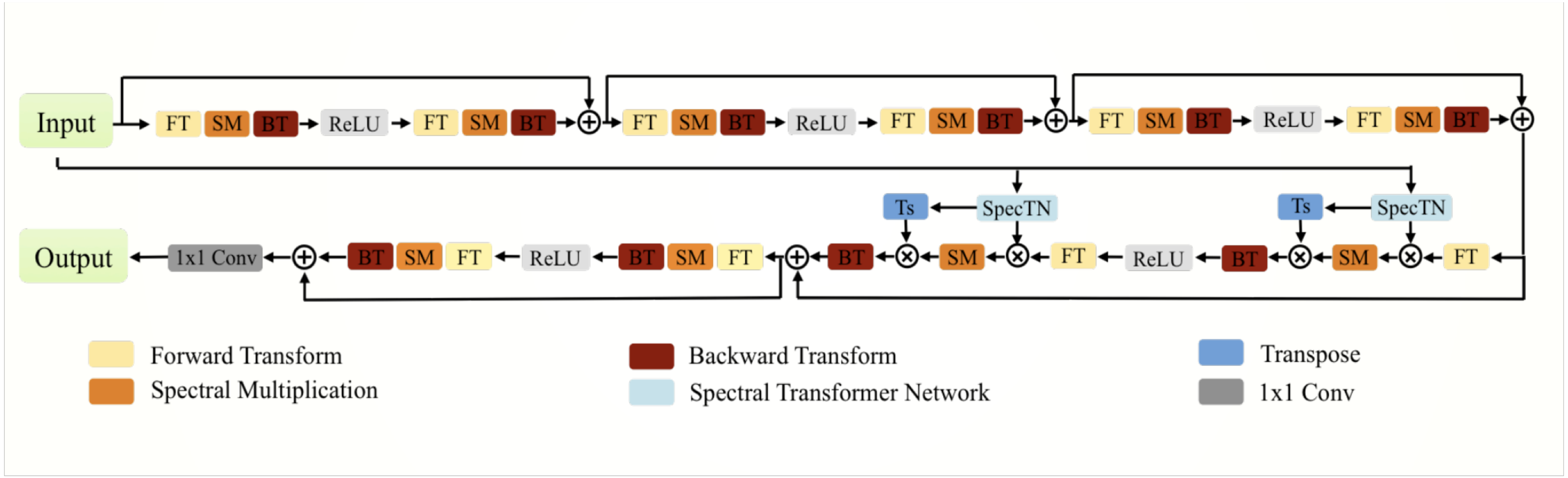
- Synchronization visualization



Yi, L., Su, H., Guo, X., Guibas, L. (2017)

SyncSpecCNN

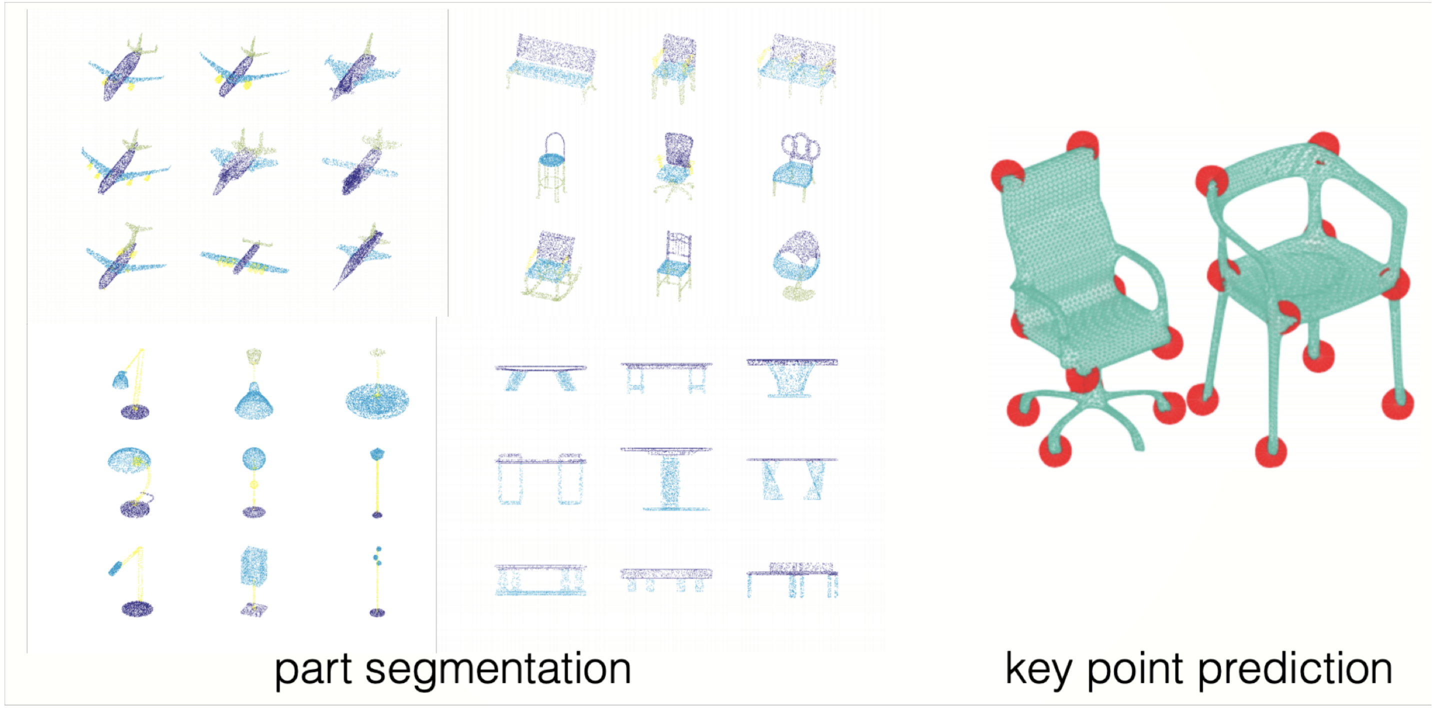
- Network Architecture



Yi, L., Su, H., Guo, X., Guibas, L. (2017)

SyncSpecCNN

- Part segmentation and keypoint detection results.



Yi, L., Su, H., Guo, X., Guibas, L. (2017)

Outline

- Motivation
- Background:
    - Graph Definition
    - Spatial vs Spectral Domains
- Challenges
- Paper:
    - DCNN
    - GATs
    - SyncSpecCNN
- Summary
- Takeaway & References

Graph CNN Summary

- Spatial construction is usually more efficient but less principled. (Diffusion and Attention papers)
- Spectral construction is more principled but usually slow. Computing Laplacian eigenvectors for large scale data could be painful. (SyncSpecCNN)
- Research tries to bridge the gap.(GCN by Kipf & Welling , Fast Localized Spectral Filtering by Defferrard et al)
- Generalization issue on generic graphs is still a challenge.

Graph CNN Summary

- Review the challenges
    - How to define compositionality on graphs (i.e. convolution, downsampling, and pooling on graphs) ?
    - How to ensure generalizability across graphs?
    - And how to make them numerically fast? (as standard CNNs)

Outline

Takeaways

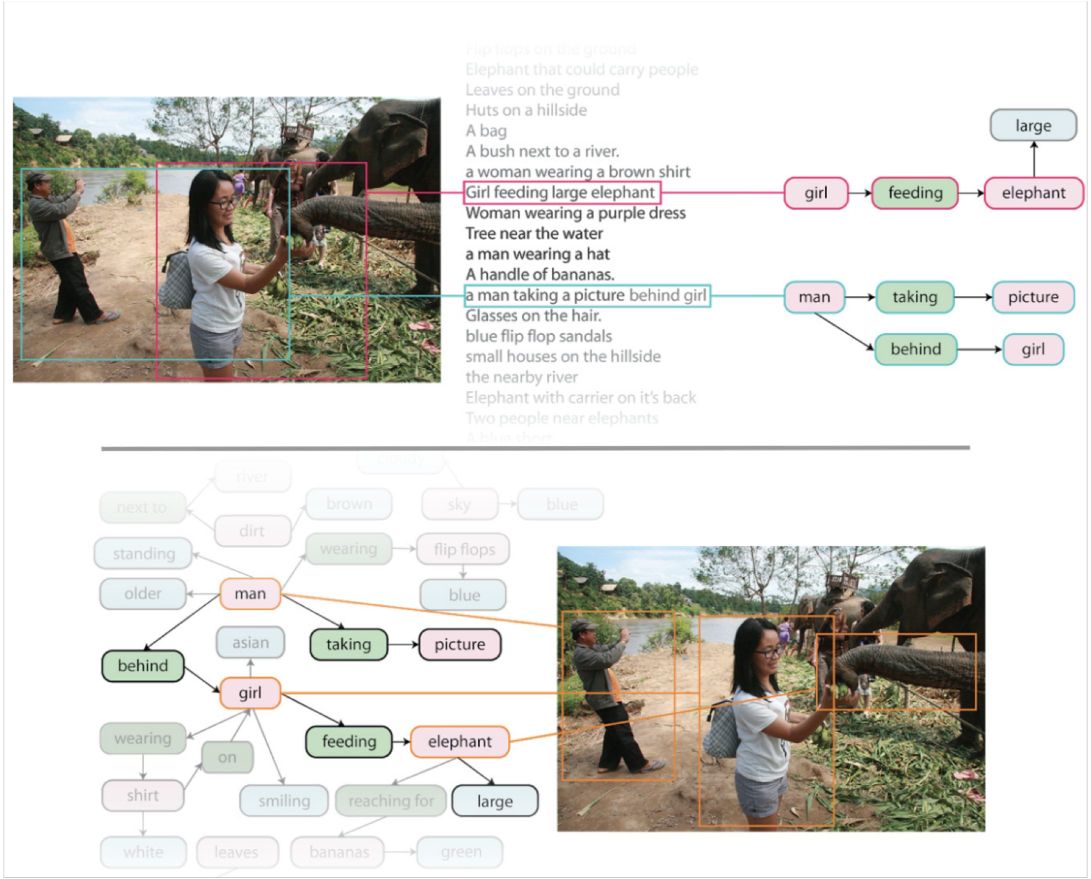
- What are graph networks?
- What are the two domains the works are relying on?
- What are their +’s and -’s?
- What are the applications that can use graph networks?

References

- Atwood, J. and Towsley, D. Diffusion-Convolutional Neural Networks (2016)
- Velickovic, P. et al. Graph Attention Networks (2018)
- Boscaini, D. et al. Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks (2015)
- Yi, L., Su, H. , Guo, X. , Guibas, L. SyncSpecCNN: Synchronized Spectral CNN for 3D Shape Segmentation (2016)
- Protein-Protein Interaction Affinity Database 2.0
- Relational Dataset Repository - CORA
- Defferrard, M. Bresson, X., Vandergheynst, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering

Scene Graphs

- Visual Genome
  - 2D natural images can be understood as a scene graph
  - Applications to scene understanding
  - Graph networks used in generation of scene graphs from natural images



Flip flops on the ground
Elephant that could carry people
Leaves on the ground
Huts on a hillside
A bag
A bush next to a river.
a woman wearing a brown shirt
Girl feeding large elephant
Woman wearing a purple dress
Tree near the water
a man wearing a hat
A handle of bananas.
a man taking a picture behind girl
Glasses on the hair.
blue flip flop sandals
small houses on the hillside
the nearby river
Elephant with carrier on it's back
Two people near elephants
A blue short
girl
feeding
elephant
large
man
taking
picture
behind
girl
river
cloudy
next to
brown
sky
blue
dirt
standing
wearing
flip flops
older
man
blue
asian
taking
picture
behind
girl
wearing
on
feeding
elephant
shirt
smiling
reaching for
large
white
leaves
bananas
green