Utilizing the Graph Structure:

And the Potential Applications in Natural Language Processing

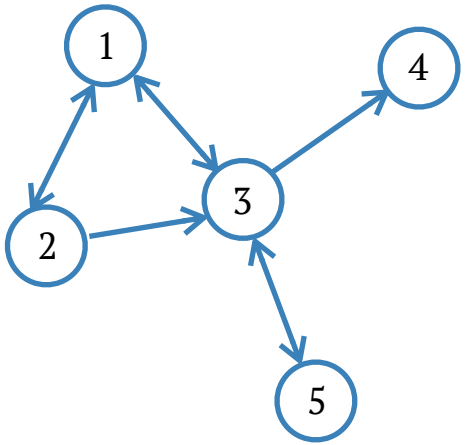
Tony

Zhuiyi Technology

Contents

What is a Graph

- Elements $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
  - Node/vertex $\mathcal{V} = \{v_i | i = 1, ..., N\}$
  - Link/edge $\mathcal{E} = \{e_{ij} | v_i, v_j \in \mathcal{V}\}, \|\mathcal{E}\| \leq N^2$
  - Auxiliary features
    - Node $X \in \mathbb{R}^{\|\mathcal{E}\| \times d_{\text{node}}}$
    - Edge $E \in \mathbb{R}^{\|\mathcal{E}\| \times d_{\text{edge}}}$
- Descriptions
  - Adjacency matrix $A \in \mathbb{R}^{N \times N}, A_{ij} = \begin{cases} a_{ij} \neq 0 & e_{ij} \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases}$
  - Degree $D \in \mathbb{R}^{N \times N}, D_{ii} = \sum_j A_{ij}$
  - Neighborhood $\mathcal{N}(v_i) = \{v_j | e_{ij} \in \mathcal{E}\}$



Variations of Graphs

- Different Edges
  - Weighted & Unweighted
  - Directed & Undirected
  - Single Edge & Multiple Edge
- Auxiliary Informations
  - Labels(discrete):
    - Complete: Homogeneous & Heterogeneous
    - Missed
  - Attributes(continuous) and Features(vector)
  - Unstructured informations: Text

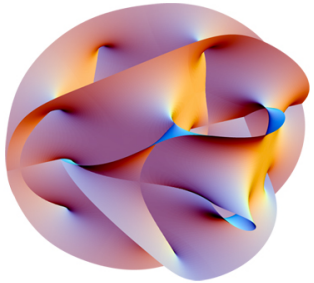
- Graph Structure
  - Static & Dynamic
  - Constructed & Intrinsic

Examples of Graphs

- Citation Graph
- Protein Graph
- Knowledge Graph
- Social Network
- Collaborative Filtering Graph
- Nearest-Neighbor Graph

Interpretation of Graphs

- Nodes are **sampled** from a complex source space
  - High dimensional manifold, Semantic space, ...
  - Sparsity - **bless**
  - Missing values - **curse**
- Structure of source space are preserved:
  - **Local relationships** between nodes are preserved in edge
  - Global structure are preserved in patterns of connectivity



Contents

Utilizing the Graph: Two Paradigms

- Graph Neural Networks: usually with external supervision **Today**
  - **Utilize graph structure** to build representations
  - **Hierarchical** feature **aggregation**, **transformation** and **coarsening**
- Graph Embeddings: usually without external supervision
  - **Approximate graph structure**(source space) with low dimensional vector space
    - **Proximity** structure: closer in embedding space if
      - Connected by edge
      - Share similar neighborhood
    - **Algebraic** structure: king - man + woman = queen
    - ...

Typical Graph Tasks

- Node Prediction
  - Node Classification
- Link Prediction
- Subgraph prediction
  - Subgraph Matching
  - Node Clustering
- Whole graph prediction

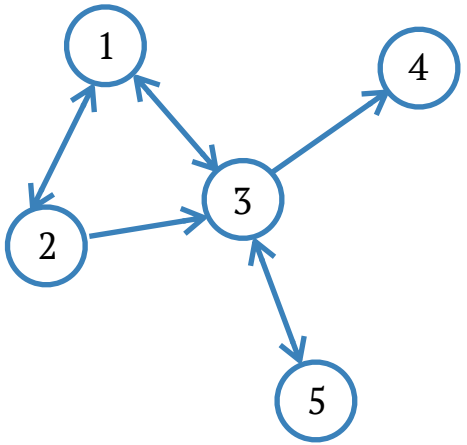
Graph Neural Networks

- Representation learning via composition and transformation
- Desiderata:
  - Parameter sharing
  - Hierarchical
  - Localized transformation - for scaling
- Two stories:
  - Spectral models(**see more in appendix**)
  - Spatial models

Contents

Briefly on Spectral Graph Convolutional Networks

- Graphs are not regularly structured
- Convolution theorem:
  - Convolution in spatial domain
  - Multiplication on spectral domain
- Define spectrum on graph
  - Map graph signal to spectral domain
  - Scale each dimension with parameters
  - Map them back to spatial domain



J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, “Spectral Networks and Locally Connected Networks on Graphs.,” arXiv, vol. cs.LG, 2013.

Briefly on Spectral Graph Convolutional Networks

- Pros
  - Parameter sharing
- Cons:
  - Difficult to scale
    - Spectrum decomposition is computationally demanding
    - Transformation involves all nodes
  - Spectrums are graph dependent

J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, “Spectral Networks and Locally Connected Networks on Graphs.,” arXiv, vol. cs.LG, 2013.

Contents

Spatial Graph Convolutional Networks

- Weighted neighborhood avg

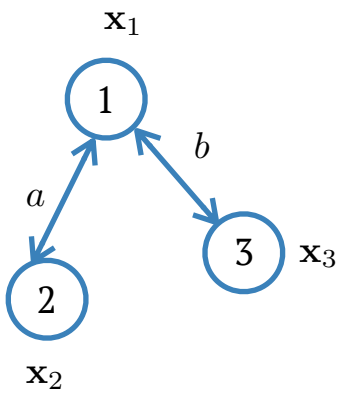
$$\mathbf{z}_1 = \frac{1}{1+a+b}(\mathbf{x}_1 + a\mathbf{x}_2 + b\mathbf{x}_3)$$
$$\mathbf{h}_1 = \sigma(W\mathbf{z}_1)$$

- Vectorize

$$H = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}XW\right) \qquad X = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix}$$

- Adjacency matrix with self loop $\tilde{A} = A + I_N$

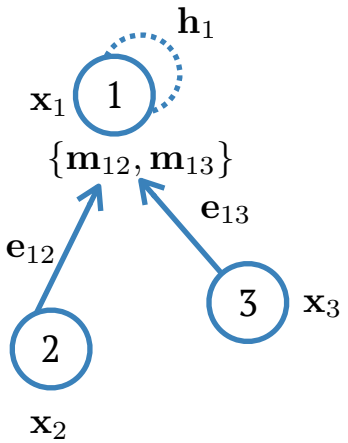
- Normalize with degree $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$
- Approximation of spectral GCN
- **Bless of sparsity!** $A \in \mathbb{R}^{N \times N}$



T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” arXiv.org, vol. cs.LG. 10-Sep-2016.

Message Passing

- Message function $\mathbf{m}_{ij} = \text{message}\left(\mathbf{x}_i, \mathbf{x}_j, \mathbf{e}_{ij}\right)$
- Aggregate function $\mathbf{z}_i = \text{aggregate}\left(\{\mathbf{m}_{ij} | v_j \in \mathcal{N}\left(v_i\right)\}\right)$
- Update function $\mathbf{h}_i = \text{update}\left(\mathbf{x}_i, \mathbf{z}_i\right)$

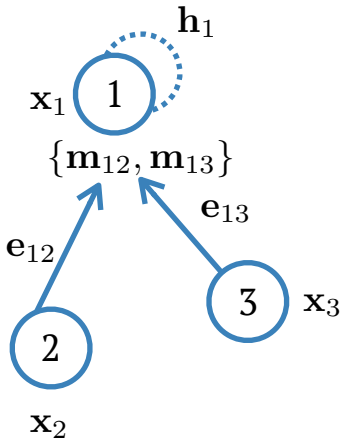


- Neighborhood information: adjacency matrix A
- Extension from MPNN
- Implementation model of popular libraries:
  - DGL
  - Pytorch Geometric

J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, “Neural Message Passing for Quantum Chemistry,” arXiv.org, vol. cs.LG. 05-Apr-2017.

Concrete Example: GraphSAGE-pool

- Message function $\mathbf{m}_{ij} = \sigma\left(W\mathbf{x}_j + \mathbf{b}\right)$
- Aggregate function $\mathbf{z}_i = \text{maxpool}\left(\{\mathbf{m}_{ij} | v_j \in \mathcal{N}\left(v_i\right)\}\right)$
- Update function $\mathbf{h}_i = \sigma\left(U \text{ concat}\left(\mathbf{x}_i, \mathbf{z}_i\right)\right)$



W. L. Hamilton, R. Ying, and J. Leskovec, “Inductive Representation Learning on Large Graphs,” arXiv.org, vol. cs.SI. 07-Jun-2017.

Concrete Example: GAT

- Message function

$$s_{ij} = \text{sim}(\hat{\mathbf{h}}_i, \hat{\mathbf{h}}_j)$$
$$\hat{\mathbf{h}}_j = W\mathbf{x}_j$$
$$\mathbf{m}_{ij} = \left[s_{ij}, \hat{\mathbf{h}}_j\right]$$

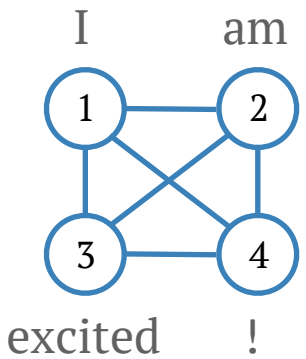
- Aggregate function

$$\mathbf{z}_i = \sum_{v_j \in \mathcal{N}(v_i)} \frac{e^{s_{ij}}}{\sum_{v_k \in \mathcal{N}(v_i)} e^{s_{ik}}} \hat{\mathbf{h}}_j$$

- Update function

$$\mathbf{h}_i = \sigma\left(\mathbf{z}_i\right)$$

- Multihead
  - **Transformer: complete graph**



G. Cucurull, A. Casanova, A. Romero, Pietro Liò, and Y. Bengio, “Graph Attention Networks,” arXiv.org, vol. stat.ML. 30-Oct-2017.

Recurrence: Reusing Parameters among Layers

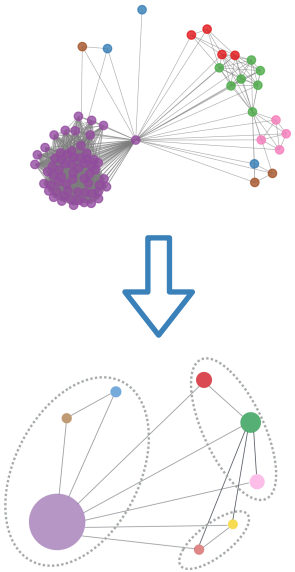
- Gated Graph Neural Networks

$$\mathbf{h}_i^t = \text{GRU}\left(\mathbf{h}_i^{t-1}, \sum_{v_j \in N(v_i)} \mathbf{W}\mathbf{h}_j^t\right)$$

K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio,
“Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.” 03-Jun-2014.

Pooling: DIFFPool

- Compute representation with GNN: $Z^l = \text{GNN}^l_{\text{embed}}\left(A^l, X^l\right)$
  - Node features $X^l \in \mathbb{R}^{N_l \times d}$
  - Adjacency matrix $A^l \in \mathbb{R}^{N_l \times N_l}$
- Compute cluster assignment matrix: $S^l = \text{softmax}\left(\text{GNN}^l_{\text{pool}}\left(A^l, X^l\right)\right)$
  - Cluster assignment matrix: $S^l \in \mathbb{R}^{N_{l+1} \times N_l}$
- Pool with cluster assignment matrix
  - Aggregate features: $X^{l+1} = (S^l)^T Z^l$ $X^{l+1} \in \mathbb{R}^{N_{l+1} \times d}$
  - Update adjacencies: $A^{l+1} = (S^l)^T A^l S^l$ $A^{l+1} \in \mathbb{R}^{N_{l+1} \times N_{l+1}}$



R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec, “Hierarchical Graph Representation Learning with Differentiable Pooling,” arXiv.org, vol. cs.LG. 23-Jun-2018.

Structural choices

- Node updates
  - message, aggregate, update
- Layers
  - MLP, residual, pooling
  - Recurrent - parameter sharing among layers
- Global message propagation
  - Node connectivity
    - special nodes, subgraphs, paths
  - Update concurrency

Wrap ups

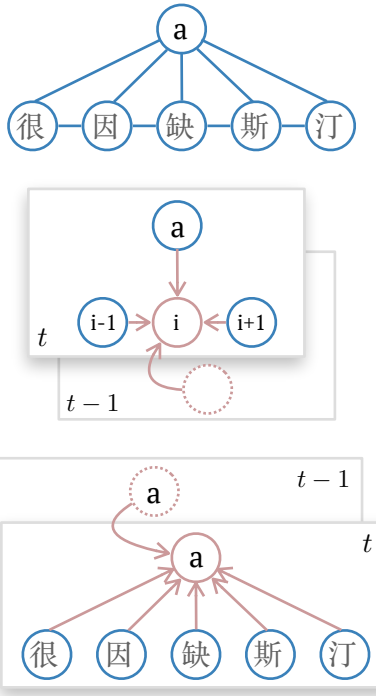
- **Utilize graph structure** to build representations
- **Hierarchical** feature **aggregation**, **transformation** and **coarsening**
  - **Hierarchical**: n-order neighborhood takes n layers to reach
  - **Aggregation**: receive message from neighborhood nodes and aggregate
  - **Transformation**: (aggregated feature, node feature)
  - **Coarsening**: graph pooling

Contents

Examples in NLP: Sentence-state LSTM

- LSTM with multiple input
- Word nodes depend on $\mathbf{h}_{i-1}^{t-1}, \mathbf{h}_{i}^{t-1}, \mathbf{h}_{i+1}^{t-1}, \mathbf{x}_i, \mathbf{a}^{t-1}$
- Global node depends on $\mathbf{h}_{1}^{t-1}, \ldots, \mathbf{h}_{n}^{t-1}, \mathbf{h}_{a}^{t-1}$

- Global node as sentence embedding: no more pooling
- Gains on text classification & sequence labeling

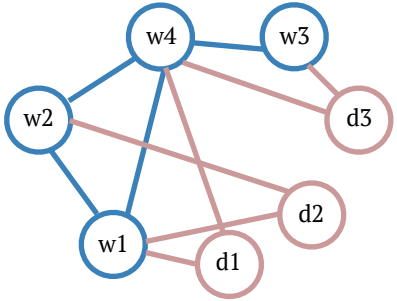


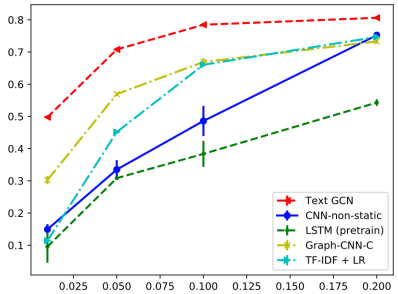
> “ … **our contribution is similar** to that of Kim (2014)[TextCNN] and Bahdanau et al. (2015)[Attention mechanism] in introducing a neural representation to the NLP literature.

Y. Zhang, Q. Liu, and L. Song, “Sentence-State LSTM for Text Representation,” arXiv.org, vol. cs.CL. 07-May-2018.

Examples in NLP: GCN in Text Classification

- Heterogeneous graph:
  - Word node - word node: co-occurrence PPMI
  - Document node - word node: TF-IDF
  - Self connection: 1
- Each node has a trainable embedding
- GCN: $Pr = \text{softmax}\left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\,\text{ReLU}\left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}XW_0\right)W_1\right)$
- Data efficiency
  - Possibly due to transductive learning





(a) 20NG

L. Yao, C. Mao, and Y. Luo, “Graph Convolutional Networks for Text Classification.,” arXiv, vol. cs.CL, 2018.

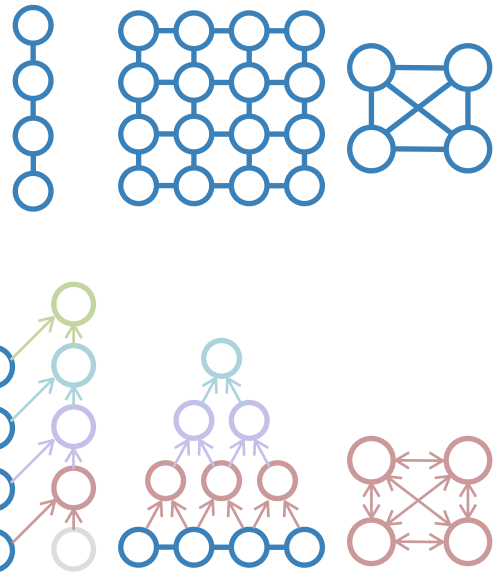
Contents

Perspective

- The Deep Learning alchemy:
  - **Loss and input to inject information**
  - Learning procedure to tame the model
  - **Models as substrate to catch regularities**
- Elements of a model:
  - Transformation
    - All the fancy stuffs
  - **Composition**

Compositional Expressivity of GNN

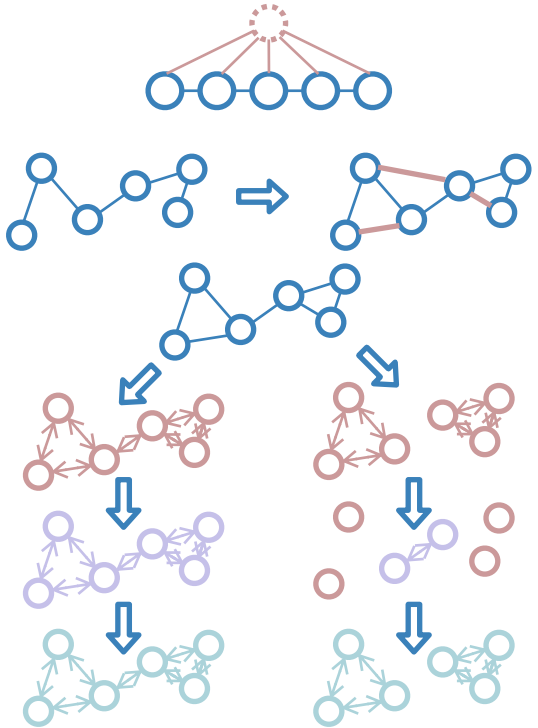
- Special cases of a Graph:
  - Sequence as chain-structured graph
  - Image as grid-structured graph
  - Transformer treat a sequence as a complete graph
- Regular(limited) compositional pattern&order:
  - RNN: 1 shallow + 1 deep, N steps aggregate all
  - CNN: k median, log_k(N) steps aggregate all
  - Transformer: N shallow/median, 1 steps aggregate all



A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” arXiv.org, vol. cs.CL. 13-Jun-2017.

Go Wild with GNN

- Special nodes
- Sparse graph with local connectivity
  - Diverse compositional order
  - Freedom to define compositional path
    - Hierarchical according to subgraphs
    - Graph trasversal



R. Liao, M. Brockschmidt, D. Tarlow, A. L. Gaunt, R. Urtasun, and R. Zemel, “Graph Partition Neural Networks for Semi-Supervised Classification,” arXiv.org, vol. cs.LG. 16-Mar-2018.

Challenges

- Where does the graph come from?
  - Labeled, automatic constructed or learned
- Huge graph: scaling problems
  - Localized operators and sampling
  - Distributed computation
- Missing data: extreme sparsity
- Graph are irregularly unstructured and diverse: operator definition
- Complex structures
  - Similarity via adjacency v.s. similarity via common neighborhood

Contents

Generalizing CNN to Graph Domain

- Graph is spatially irregularly structured:
  - Convolution: translation is undefined
    - Shift the filter left by 5?
  - Pooling: principled coarsening is undefined
    - Max pooling of 2*2 block?
- Yet convolution is well defined in frequency domain
  - What is "frequency" in graph?

Characterizing Signal Variation

- Graph signal: $\mathbf{f} = [f(v_1), ..., f(v_N)] \in \mathbb{R}^N$
- Weighted undirected graph: $A_{ij} = A_{ji}$
- Smooth(vary slowly) if **similar** nodes has similar values:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} A_{ij}(f(v_i) - f(v_j))^2$$

- Vectorize:

$$\mathbf{f}^T L\mathbf{f} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} A_{ij}(f(v_i) - f(v_j))^2$$

  - Graph Laplacian: $L = D - A$
  - Node Degree:

$$\begin{aligned} D &= \text{diag}(\deg(v_1), ..., \deg(v_N)) \\ &= \text{diag}(\sum_{v_j \in \mathcal{N}(v_1)} A_{1j}, ..., \sum_{v_j \in \mathcal{N}(v_N)} A_{Nj}) \end{aligned}$$

Graph Laplacian: Example

- Graph signal: $\mathbf{f} = [f(v_1), f(v_2), f(v_3)]^T, f(v_i) \in \mathbb{R}$

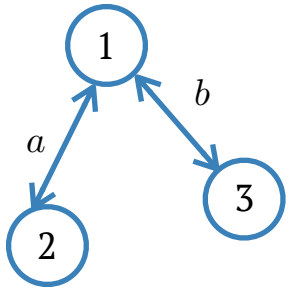
$$A = \begin{bmatrix} 0 & a & b \\ a & 0 & 0 \\ b & 0 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} a+b & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \end{bmatrix} \quad L = \begin{bmatrix} a+b & -a & -b \\ -a & a & 0 \\ -b & 0 & b \end{bmatrix}$$

$$\mathbf{f}^T L \mathbf{f} = [f(v_1), f(v_2), f(v_3)] \begin{bmatrix} a+b & -a & -b \\ -a & a & 0 \\ -b & 0 & b \end{bmatrix} \begin{bmatrix} f(v_1) \\ f(v_2) \\ f(v_3) \end{bmatrix}$$

$$= \sum \begin{pmatrix} af(v_1)^2 + bf(v_1)^2 & -af(v_1)f(v_2) & -bf(v_1)f(v_3) \\ -af(v_2)f(v_1) & af(v_2)^2 & 0 \\ -bf(v_1)f(v_3) & 0 & bf(v_3)^2 \end{pmatrix}$$

$$= a(f(v_1) - f(v_2))^2 + b(f(v_1) - f(v_3))^2$$

$$= \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} A_{ij} (f(v_i) - f(v_j))^2$$



Eigenvectors of Graph Laplacian

- Graph Laplacian has a **complete** set of **orthonormal** eigenvectors
- They can also defined iteratively via Rayleigh quotient:

$$\lambda_0 = \min_{\|\mathbf{f}\|_2=1} \mathbf{f}^{\mathrm{T}} L\mathbf{f},$$

$$\lambda_l = \min_{\substack{\|\mathbf{f}\|_2=1 \\ \mathbf{f}\perp \mathrm{span}\{\mathbf{u}_0,\ldots,\mathbf{u}_{l-1}\}}} \mathbf{f}^{\mathrm{T}} L\mathbf{f}$$

$$L = U^T \Lambda U = \begin{bmatrix} \mathbf{u}_0^T \\ \vdots \\ \mathbf{u}_N^T \end{bmatrix} \begin{bmatrix} \lambda_0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_N \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 & \cdots & \mathbf{u}_N \end{bmatrix}$$

- Where $\lambda_0 < \lambda_1 < \ldots < \lambda_N$
  - Smaller eigenvalue - eigenvector vary slowly across similar nodes
  - "Frequency" - "variation speed"

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,” arXiv.org, vol. cs.DM, no. 3. pp. 83–98, 01-Nov-2012.

Fourier Transform on Graph

One Dimensional Laplacian

$$\nabla^2 f = \frac{\partial^2 f}{\partial^2 t}$$

Fourier Spectrum

$$\nabla^2(e^{-i2\pi\xi t}) = \frac{\partial^2}{\partial t^2}(e^{-i2\pi\xi t}) = (2\pi\xi)^2(e^{-i2\pi\xi t})$$

Fourier Transform

$$\int_t f(t)e^{-i2\pi\xi t}dt$$

Graph Laplacian

$$L$$

Graph Fourier Spectrum

$$L\mathbf{u} = \lambda\mathbf{u}$$

Graph Fourier Transform

$$U\mathbf{f}$$

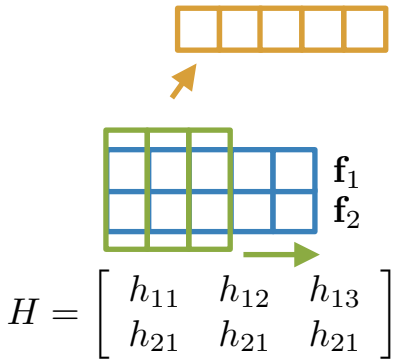
D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains,” arXiv.org, vol. cs.DM, no. 3. pp. 83–98, 01-Nov-2012.

Convolution Theorem

- Convolution - Point-wise product

$$\mathcal{F}\{h * f\} = \mathcal{F}\{h\}\mathcal{F}\{f\}$$

- Breaking down Convolutional Neural Network
  - Each channel is a signal
  - Weighted sum among channels
  - Convolution among time/grid
    - Applicable with convolution theorem



Graph Convolutional Network: Spectral

- Convolution on Graph Spectrum(single channel in/out):

$$\mathbf{h} * \mathbf{f} = U(U^T\mathbf{h} \odot U^T\mathbf{f}) = U(\text{diag}(U^T\mathbf{h})U^T\mathbf{f})$$

- Parameterize the filter - graph convolutional network

$$\mathbf{g} = \sigma(U(\text{diag}(\mathbf{w})U^T\mathbf{f}))$$

- Multi-channels: j-th in-channel to i-th out-channel

$$\mathbf{g}_i = \sigma(U(\sum_j \text{diag}(\mathbf{w}_{ij})U^T\mathbf{f}_j))$$

J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, “Spectral Networks and Locally Connected Networks on Graphs.,” arXiv, vol. cs.LG, 2013.

Contents

Review Recommendation

- Graph signal processing
  - Shuman, David I, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2012. “The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains.” arXiv.org. doi:10.1109/MSP.2012.2235192.
- Graph Neural Networks
  - Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A Comprehensive Survey on Graph Neural Networks,” arXiv.org, vol. cs.LG. 03-Jan-2019.
- Graph Embedding
  - H. Cai, V. W. Zheng, K. C. I. T. on, 2018, “A comprehensive survey of graph embedding: Problems, techniques, and applications,” ieeexplore.ieee.org