HyperGraph Neural Networks

Written by: Feng et al.

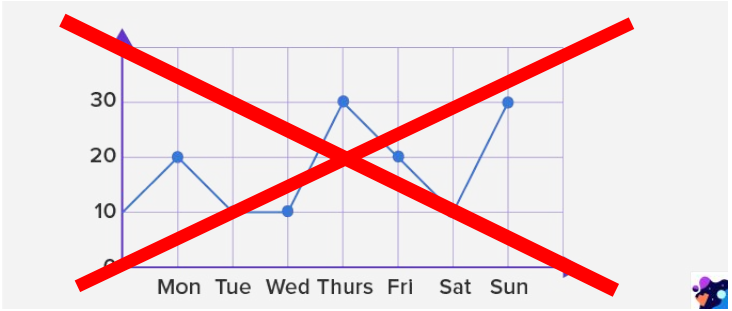
Presented by Shawn Catudal

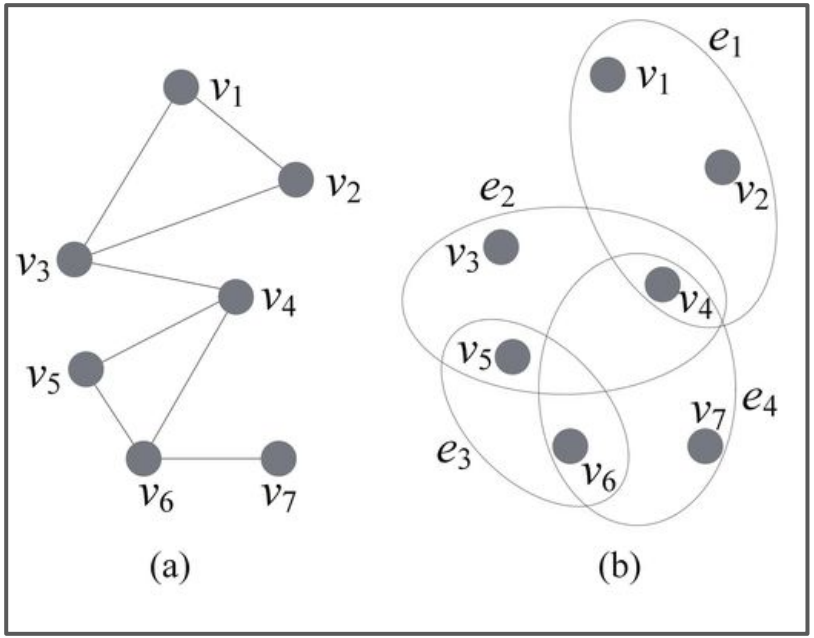
Overview

- Background
- HyperGraph Neural Networks
- Experiments
- Results
- Conclusion

Background

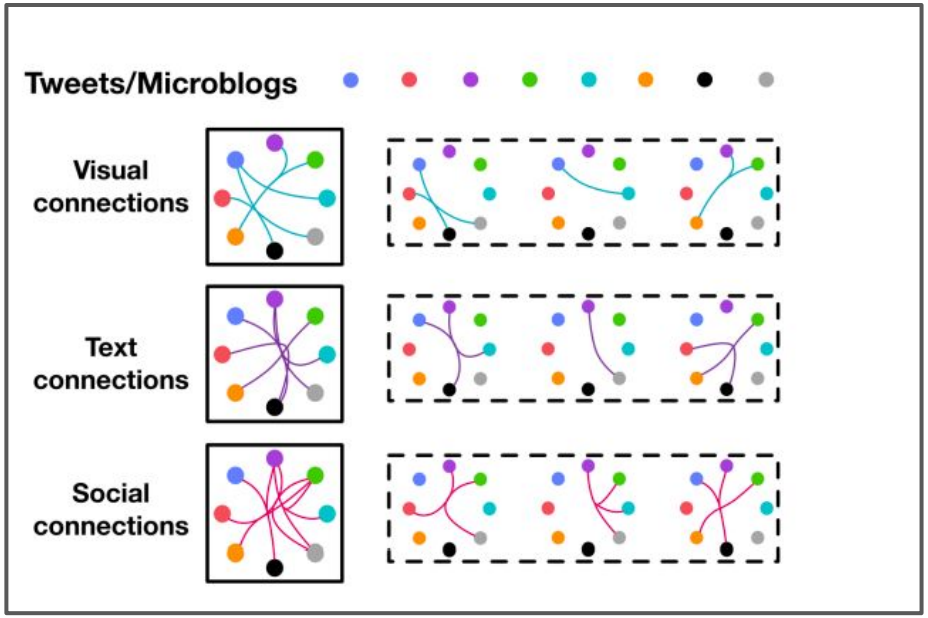
- Graph Based Neural Networks have been gaining popularity
  - Able to encode graph structure of different inputs
  - Have shown Superiority on representation learning
    - Representation learning is about learning features from raw data that effectively represent data in a useful way
- Graphs vs HyperGraphs
  - Traditional Graphs cannot portray more complex data.
  - Only pairwise interactions (link)
    - Edge vs hyperedge
  - Not well suited for multi model
    - Visual, text and social connections
  - Hypergraphs are a generalization of graphs
    - Can have hyperedges that connect any number of nodes





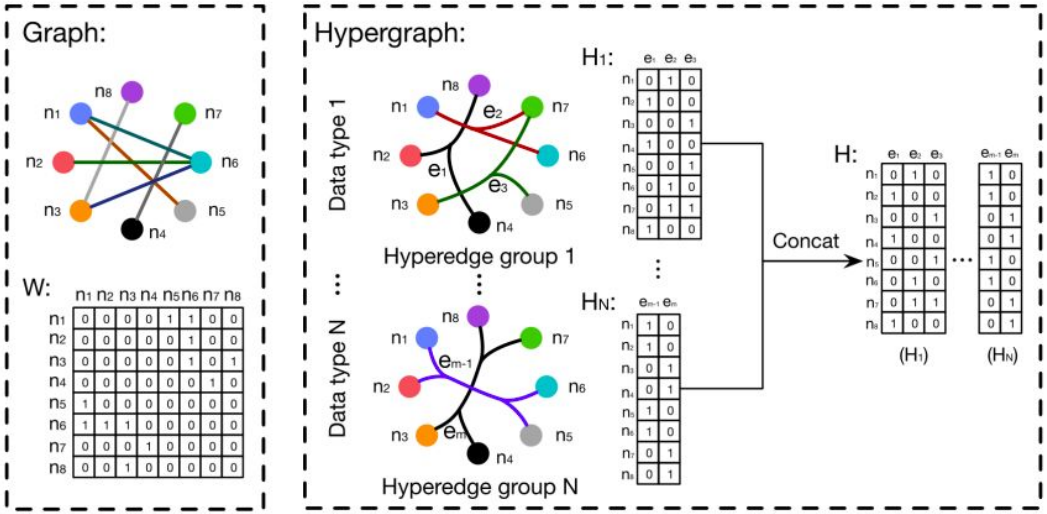
Background

- Example:
  - Social Media Connections
    - Friends
    - Likes
    - Posts
- Becomes very complex with scale
- Hypergraphs allow for relationship between the different types of data
- However, due to complexity, hypergraphs in machine learning are computationally very expensive
  - Limits application



Background

- Hypergraph/Graph Representation
  - Adjacency matrix vs. Incidence matrix
    - node x node for traditional graph adjacency matrix
    - node x edge for hypergraph incidence matrix



Goals of this paper

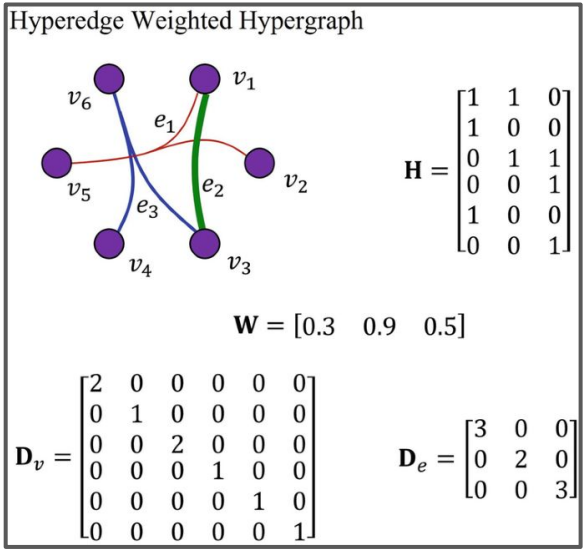
- Propose Hypergraph Neural Network framework for data representation learning
- Design a better convolution operation
- Ability to incorporate multimodal data
- Show improved results compared state of the art

Hypergraph Matrices

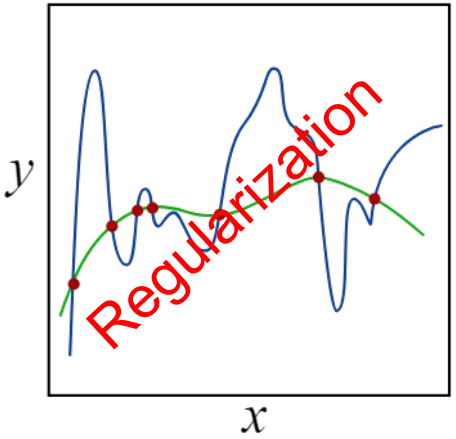
- Hypergraph is defined as
  - $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$
  - $\mathcal{V}$ is vertex set
  - $\mathcal{E}$ is hyperedge set
  - $\mathbf{W}$ is diagonal weight matrix for the hyperedges
- The Hypergraph can be denoted by a $|\mathcal{V}| \times |\mathcal{E}|$ incidence matrix $\mathbf{H}$ with entries:

$$h(v,e) = \begin{cases} 1, & \text{if } v \in e \\ 0, & \text{if } v \notin e, \end{cases}$$

- Degree of vertex defined as $d(v) = \sum_{e \in \mathcal{E}} \omega(e) h(v,e)$
- Degree of edge defined as $\delta(e) = \sum_{v \in \mathcal{V}} h(v,e)$
- D_e and D_v are diagonal matrices of respective degree



Hypergraph Node-Vertex Classification



- Node Vertex Classification Problem
  - Node labels should be “smooth” on hypergraph

$$\arg\min_f \{\mathcal{R}_{emp}(f) + \Omega(f)\},$$

$$\Omega(f) = \frac{1}{2} \sum_{e \in \mathcal{E}} \sum_{\{u,v\} \in \mathcal{V}} \frac{w(e)h(u,e)h(v,e)}{\delta(e)} \left(\frac{f(u)}{\sqrt{d(u)}} - \frac{f(v)}{\sqrt{d(v)}} \right)^2$$

- $R_{emp}(f)$ is supervised empirical loss
- $\Omega(f)$ is regularization term
- f(.) is classification function

Hypergraph Node-Vertex Classification

- Let:

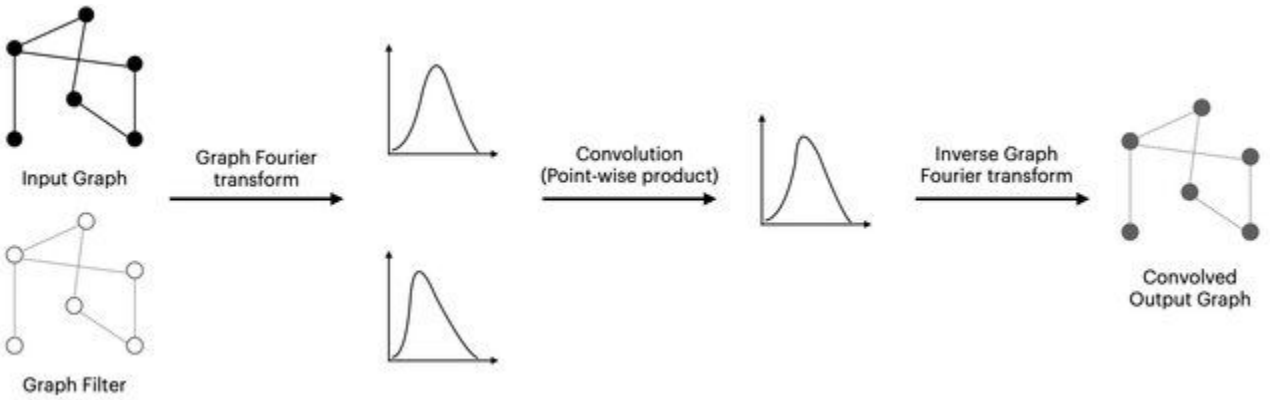
$$\theta = \mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^\top\mathbf{D}_v^{-1/2}$$

$$\mathbf{\Delta} = \mathbf{I} - \mathbf{\Theta}$$

- $\mathbf{\Theta}$ is basically the structure of the graph, including weights
- $\mathbf{\Delta}$ is the hypergraph Laplacian
  - Positive semi-definite
  - The Laplacian reflects the structure of the hypergraph
- Normalized regularization term can then be written as $\Omega(f) = f^\top \mathbf{\Delta}$

Spectral Convolution

- Graphs and Hypergraphs have an irregular shape
    - Unlike images or video with grid like shapes
- High level overview:



Spectral Convolution

- $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{\Delta})$ is a hypergraph with n vertices
- $\mathbf{\Delta}$ is a n x n positive semi-definite matrix
- $\mathbf{\Delta} = \mathbf{\Phi \Lambda \Phi}^\top$ is then the eigen decomposition
- $\mathbf{\Phi}$ is matrix of eigenvectors
- $\mathbf{\Lambda}$ is diagonal matrix of eigenvalues
- $\mathbf{x} = (\mathbf{x_1}, \ldots, \mathbf{x_n})$ is a signal or features

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \\ x_d \end{bmatrix}$$

Feature vector

Spectral Convolution

- Eigenvectors are regarded as the Fourier bases and eigenvalues are interpreted as frequencies.
- $\hat{\mathbf{x}} = \mathbf{\Phi}^\top \mathbf{x}$ Is the Fourier transform of the signal **x**
- Spectral convolution of signal **x** and filter **g**

$$\mathbf{g} \star \mathbf{x} = \mathbf{\Phi}((\mathbf{\Phi}^\top \mathbf{g}) \odot (\mathbf{\Phi}^\top \mathbf{x})) = \mathbf{\Phi} g(\mathbf{\Lambda}) \mathbf{\Phi}^\top \mathbf{x}$$

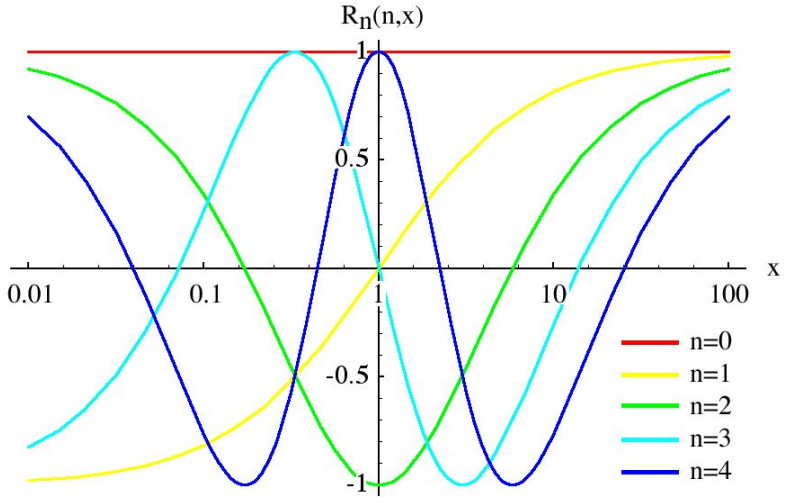
- $\odot$ is the element-wise Hadamard product
- $g(\mathbf{\Lambda}) = \mathrm{diag}(\mathbf{g}(\lambda_1), \ldots, \mathbf{g}(\lambda_n))$ is a function of fourier coefficients
- However this computation cost of the forward and inverse fourier transforms are very high making it not ideal. Moving on...

Hyperedge Convolution

- Parametrize with k order polynomials
  - Truncated Chebyshev polynomials
  - $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$
  - Limit k to 1
  - $T_0(x) = 1 \quad T_1(x) = x$
- Convolution parameterized as follows

$$\mathbf{g} \star \mathbf{x} \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{\mathbf{\Delta}})\mathbf{x}$$

  - Scaled Laplacian $\tilde{\mathbf{\Delta}} = \frac{2}{\lambda_{max}}\mathbf{\Delta} - \mathbf{I}$
  - $\lambda_{max} \approx 2$



Hyperedge Convolution

- Further Simplification of convolution operation

$$\mathbf{g} \star \mathbf{x} \approx \theta_0 \mathbf{x} - \theta_1 \mathbf{D}^{-1/2} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-1/2} \mathbf{x}$$

$$\begin{cases} \theta_1 = -\frac{1}{2}\theta \\ \theta_0 = \frac{1}{2}\theta \mathbf{D}_v^{-1/2} \mathbf{H} \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-1/2} \end{cases}$$

$$\begin{aligned} \mathbf{g} \star \mathbf{x} &\approx \frac{1}{2}\theta \mathbf{D}_v^{-1/2} \mathbf{H} (\mathbf{W} + \mathbf{I}) \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-1/2} \mathbf{x} \\ &\approx \theta \mathbf{D}_v^{-1/2} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-1/2} \mathbf{x}, \end{aligned}$$

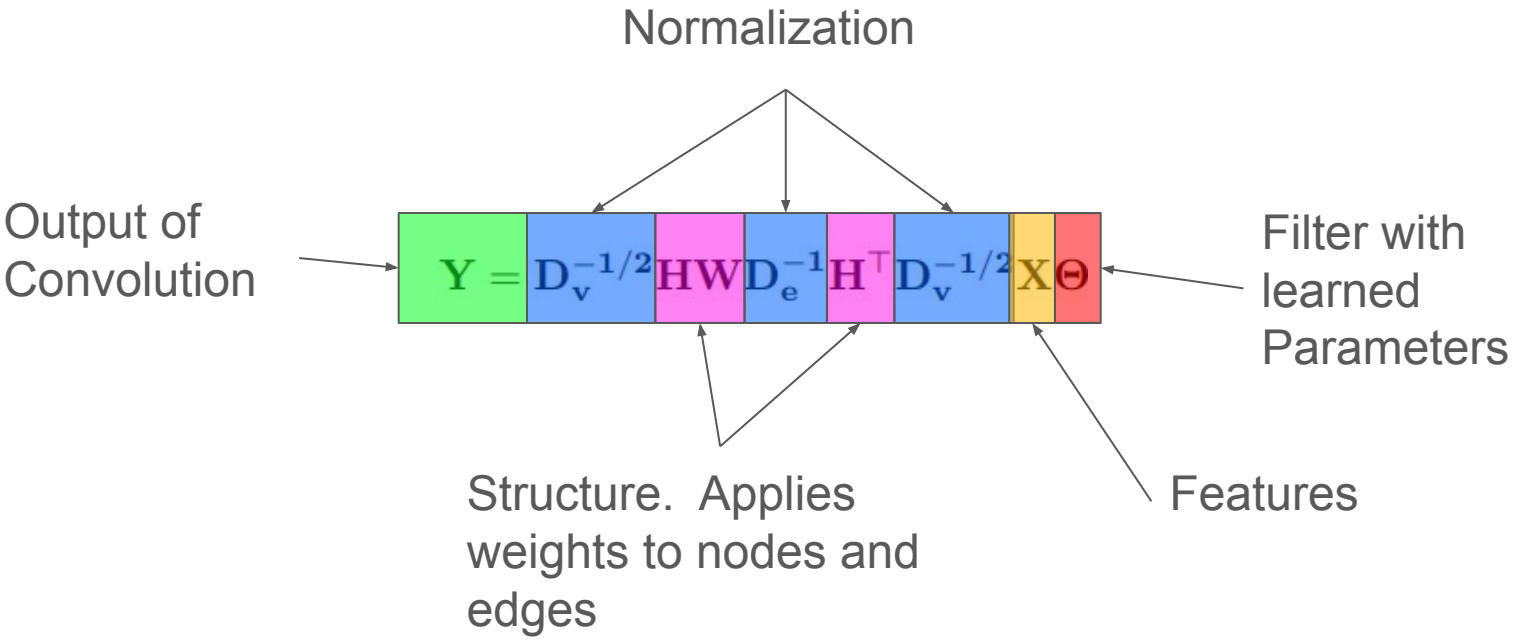
$$\mathbf{Y} = \mathbf{D}_v^{-1/2} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-1/2} \mathbf{X} \mathbf{\Theta}$$

- $\mathbf{\Theta}$ is the parameter learned during training
- Y is the output of the convolution
  - Similar to a feature map in standard convolution

$$\mathbf{g} \star \mathbf{x} \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{\mathbf{\Delta}}) \mathbf{x}$$

$$\mathbf{D}_v^{-1/2} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{D}_v^{-1/2}$$

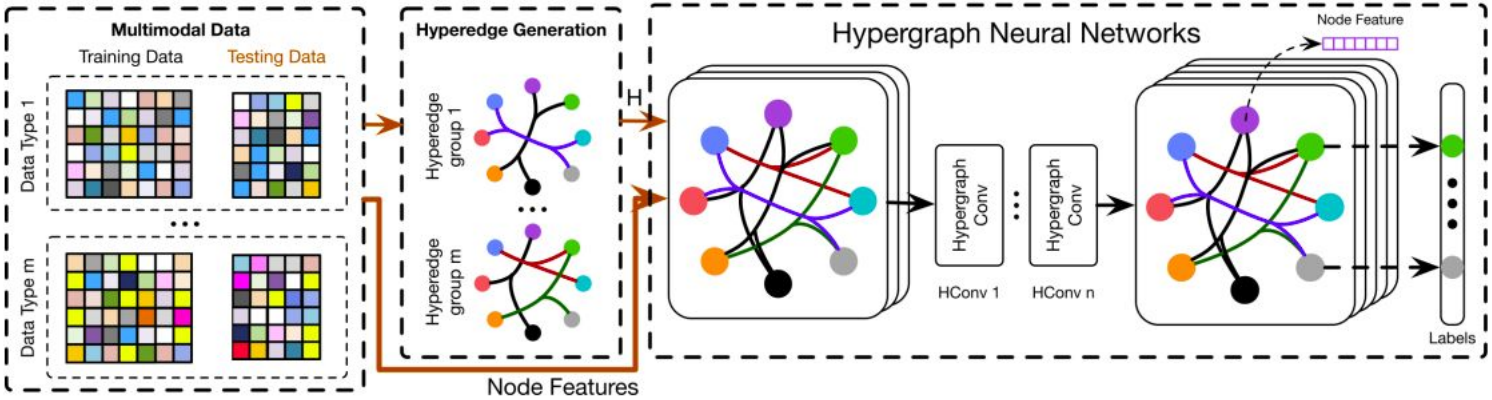
Equation explained



Multiple hyperedge groups are constructed from data

H and node features fed into HGNN

Resulting transformed hyperedge features



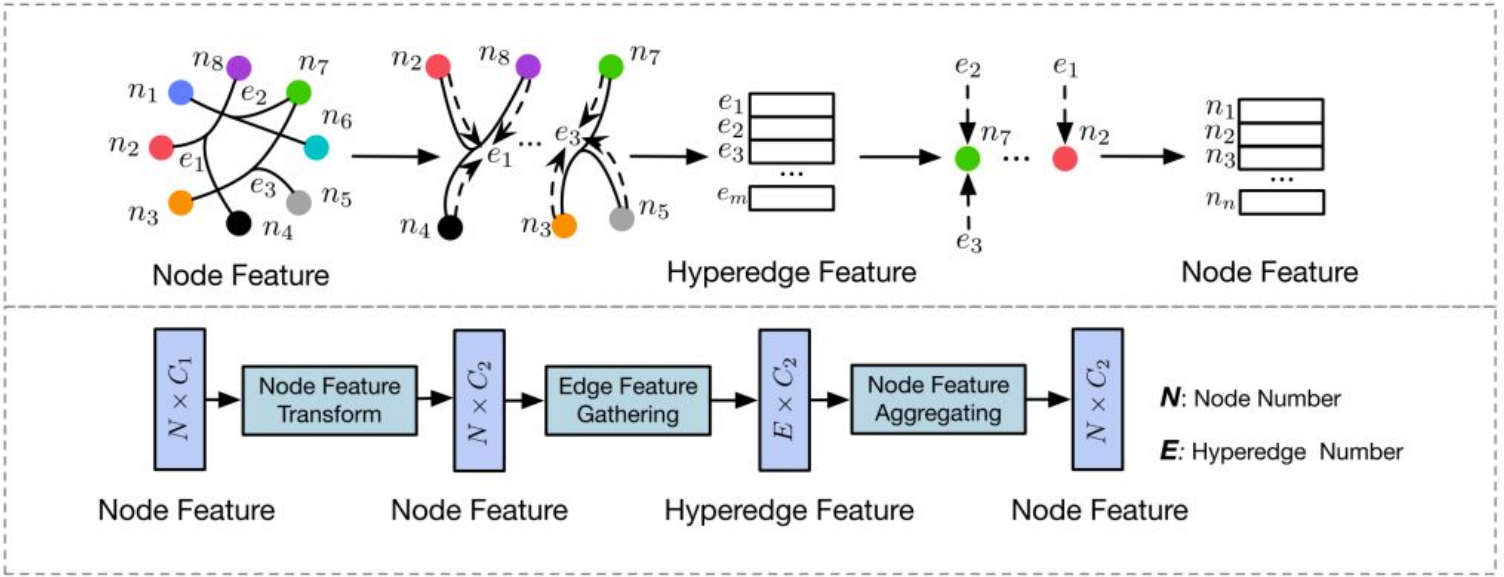
Multi-modality datasets divided into training and test

Hyperedge groups are concatenated and form hypergraph adjacency matrix **H**

Convolution on hypergraph to extract higher order correlations

Resulting label from classification of each node

Hyperedge convolution layer



$$\mathbf{Y} = \mathbf{D}_\mathbf{v}^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_\mathbf{e}^{-1}\mathbf{H}^\top\mathbf{D}_\mathbf{v}^{-1/2}\mathbf{X}\mathbf{\Theta},$$

Datasets for Experiments

- Task 1
  - Classify citation data
  - Feature for each data is bag-of-words representation of the documents

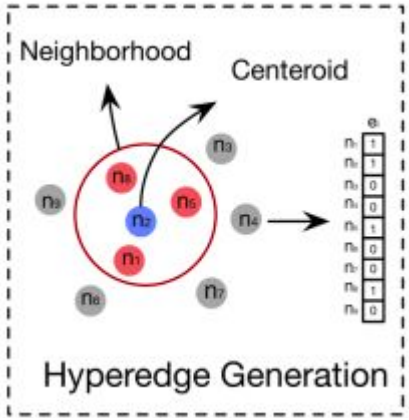
Dataset	Cora	Pumbed
Nodes	2708	19717
Edges	5429	44338
Feature	1433	500
Training node	140	60
Validation node	500	500
Testing node	1000	1000
Classes	7	3

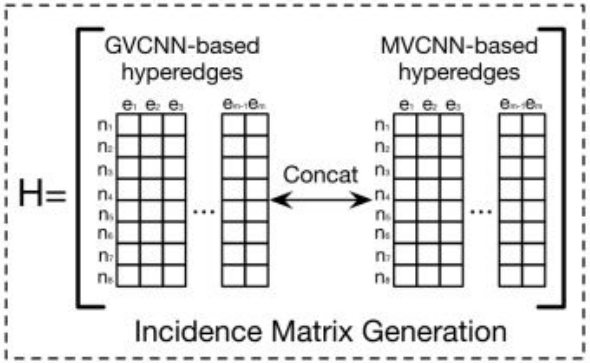
- Task 2
  - Classify visual objects
  - Datasets are of 3d objects.

Dataset	ModelNet40	NTU
Objects	12311	2012
MVCNN Feature	4096	4096
GVCNN Feature	2048	2048
Training node	9843	1639
Testing node	2468	373
Classes	40	67

Experiments

- Two tasks
- 1. Citation Network Classification
  - Use two-layer HGNN
    - Feature dimension of hidden layer is 16
    - ReLU as nonlinear activation function
    - Adam optimizer to minimize cross entropy loss
- 2. Visual object recognition
  - Hypergraph built according to distance between two features
  - Hyperedges formed based on k-nearest neighbor of vertices
  - Features are extracted using MVCNN and GVCNN and then concatenated
- Compare to state of the art





Results of Citation Network classification

Method	Cora	Pubmed
DeepWalk (Perozzi, Al-Rfou, and Skiena 2014)	67.2%	65.3%
ICA (Lu and Getoor 2003)	75.1%	73.9%
Planetoid (Yang, Cohen, and Salakhutdinov 2016)	75.7%	77.2%
Chebyshev (Defferrard, Bresson, and Vandergheynst 2016)	81.2%	74.4%
GCN (Kipf and Welling 2017)	81.5%	79.0%
HGNN	**81.6%**	**80.1%**

Results of Visual object Classification

Feature	Features for Structure					
	GVCNN		MVCNN		GVCNN+MVCNN	
	GCN	HGNN	GCN	HGNN	GCN	HGNN
GVCNN (Feng et al. 2018)	91.8%	**92.6%**	91.5%	**91.8%**	92.8%	**96.6%**
MVCNN (Su et al. 2015)	92.5%	**92.9%**	86.7%	**91.0%**	92.3%	**96.6%**
GVCNN+MVCNN	-	-	-	-	94.4%	**96.7%**

Table 4: Comparison between GCN and HGNN on the ModelNet40 dataset.

Feature	Features for Structure					
	GVCNN		MVCNN		GVCNN+MVCNN	
	GCN	HGNN	GCN	HGNN	GCN	HGNN
GVCNN ((Feng et al. 2018))	78.8%	**82.5%**	78.8%	**79.1%**	75.9%	**84.2%**
MVCNN ((Su et al. 2015))	74.0%	**77.2%**	71.3%	**75.6%**	73.2%	**83.6%**
GVCNN+MVCNN	–	–	–	–	76.1%	**84.2%**

Table 5: Comparison between GCN and HGNN on the NTU dataset.

Method	Classification Accuracy
PointNet (Qi et al. 2017a)	89.2%
PointNet++ (Qi et al. 2017b)	90.7%
PointCNN (Li et al. 2018)	91.8%
SO-Net (Li, Chen, and Lee 2018)	93.4%
HGNN	**96.7%**

Table 6: Experimental comparison among recent classification methods on ModelNet40 dataset.

Discussion

- Proposed HGNN method outperforms state of the art methods on specified datasets
- Citation network saw slight improvements
  - This is due to the generated hypergraphs being quite similar to the regular graph structure
- Visual object recognition task saw significant improvement
  - 4.8% and 3.2% over PointCNN and SO-Net
  - Improvements over GCN as well of up to 10.4%

Conclusion

- Proposed a framework for Hypergraph Neural Network (HGNN)
  - Generalizes the convolution operation to hypergraphs
  - Spectral convolution is parameterized and truncated
- HGNN is more general framework which can handle the complex and high order correlations through hypergraphs for representation learning
- Experiments show improvements in several aspects over state of the art and GCN

Questions?

Thanks :)