HyperGCN: A New Method of Training Graph Convolutional Networks on Hypergraphs

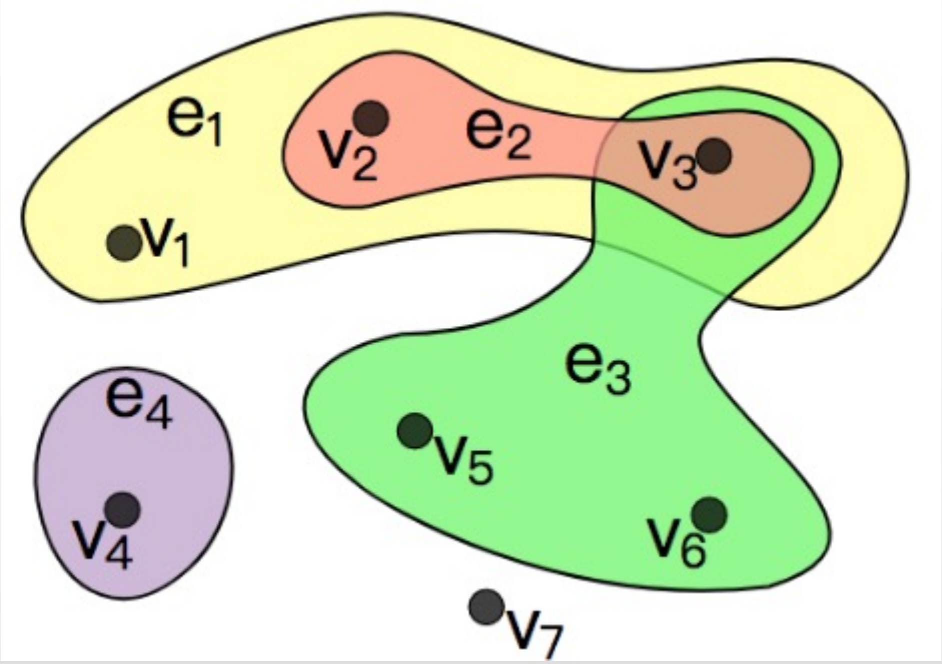
Presenter: Keyan

outline

- Hpyergraph introduction
- GCN backgroud
- Contributions
  - 1-HyperGCN
  - HyperGCN
  - FastHyperGCN
- Result Comparison
- Conclusion
- Questions

Hypergraph introduction

- Hypergraph provides a flexable modeling tool to model a complex and go beyond pairwise relationships.
- datasets examples: co-authors, email communacations
- usually we can define two vertices to construct a common graph. However for a hypergraph, a hypereage will contain multible vertices.
- like the right side picture.



GCN backgroud

- The main idea is to use a node's neighbors and neighbors' neighbors', etc information to represent the node's final represent information. The closer neighbors have more influences.
- The forward model for a simple two-layer GCN formula used here:

$$Z = f_{GCN}(X, A) = \mathrm{softmax}\left(\bar{A}\ \mathrm{ReLU}\left(\bar{A} X \Theta^{(1)}\right)\Theta^{(2)}\right), \tag{1}$$

- Here, X is the feature data matrix which contained graph signals, A is a adjacency matrix. $\theta^{(1)}$ is the input-to-hidden weight matrix, $\theta^{(2)}$ is the hidden-to-output weight matrix. Both θ are trained using gradient descent.

GCN backgroud

- GCN training for SSL: For multi-class, classification with q classes, they minimise cross-entropy:

$$\mathcal{L} = -\sum_{i \in \mathcal{V}_L} \sum_{j=1}^{q} Y_{ij} \ln Z_{ij}, \tag{2}$$

- VL is the set of labeled examples.

Contributions

- proposed a new training method for training Semi supervised learning on hypergraph and also introduce its variants.
- 1-HyperGCN
- HyperGCN
- FastHyperGCN

1-HyperGCN

- new idea is to use Hypergraph Laplacian over a simplified hypergraph
  - 1.For each hyperedge e, let $(i_e, j_e) = argmax_{i,j\in e}|S_i - S_j|$, breaking ties randomly. S is the signal defined on the hypernodes. this step is to find two vertices that represent largest signals of each hyperedge.
  - 2.A weighted simple graph G is constructed by adding step1 edges with weights $w(\{i_e, j_e\}) = w(e)$, where $w(e)$ is the weights of the hypereage e. let As denote the weighted adjacency matrix of G.
  - 3.The symmetrically normalised hypergraph Laplacian is:

$$\mathbb{L}(S) := (I - D^{-\frac{1}{2}} A_S D^{-\frac{1}{2}})S$$

    - the D is degree matrix of As.
- The L(s) laplacian matrix is the input $\bar{A}$ for GCN to perform.

1-HyperGCN

- in the above GCN process, when applied to a hypernode, in the neural message-passing framework(2017) $h_v^{(\tau+1)} = \sigma\left((\Theta^{(\tau)})^T \sum_{u \in \mathcal{N}(v)} ([\bar{A}_S^{(\tau)}]_{v,u} \cdot h_u^{(\tau)})\right).$
  - $h_v^{(\tau+1)}$ is the new hidden layer representation of node v
  - σ is a non-linear activation function
  - τ is the epoch number
  - N (u) is the set of neighbours of v
  - $[\bar{A}_S^{(\tau)}]_{v,u}$ is the weight on the edge {v, u} after normalisation

1-HyperGCN

for epoch τ, use two nodes based on max L2 norm max hidden layer represtation to determine hyperedge

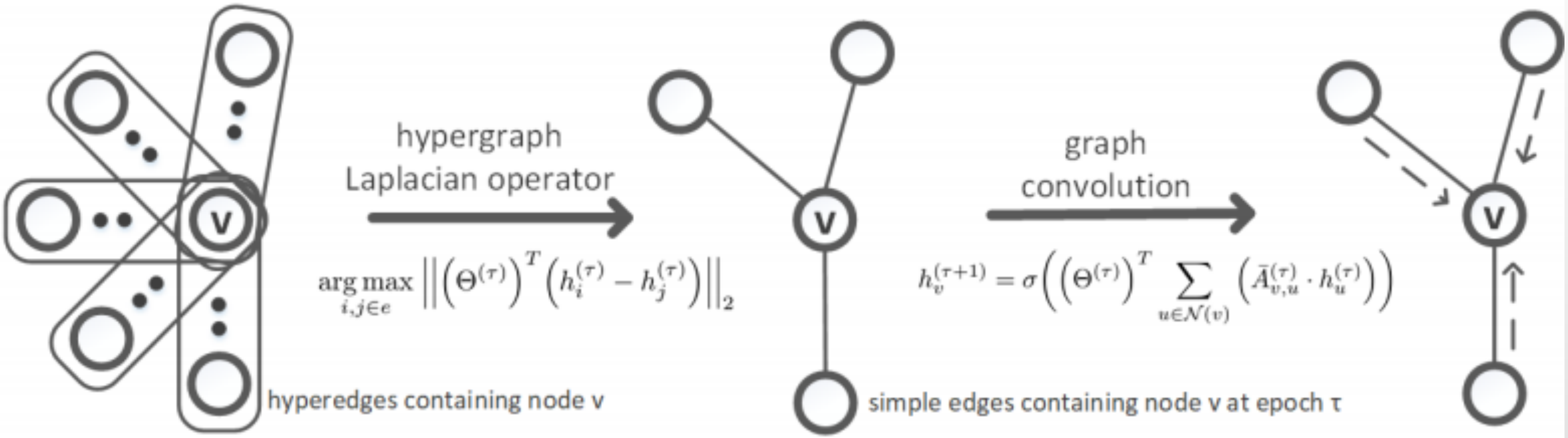


Figure 1: Graph convolution on a hypernode v using HyperGCN.

HyperGCN

- HyperGCN: a variant enhancing 1-HyperGCN with mediators.
- what are mediators:
  - those ignored nodes on each hyperedge. Since those ignored nodes cause signal loss. they use these nodes as mediators.
- weights of each mediators edge:

  When adding one connecting mediators, the corelated edge will be 2|e|-3, so the weight of each mediators edge is $\frac{1}{2|e|-3}$.

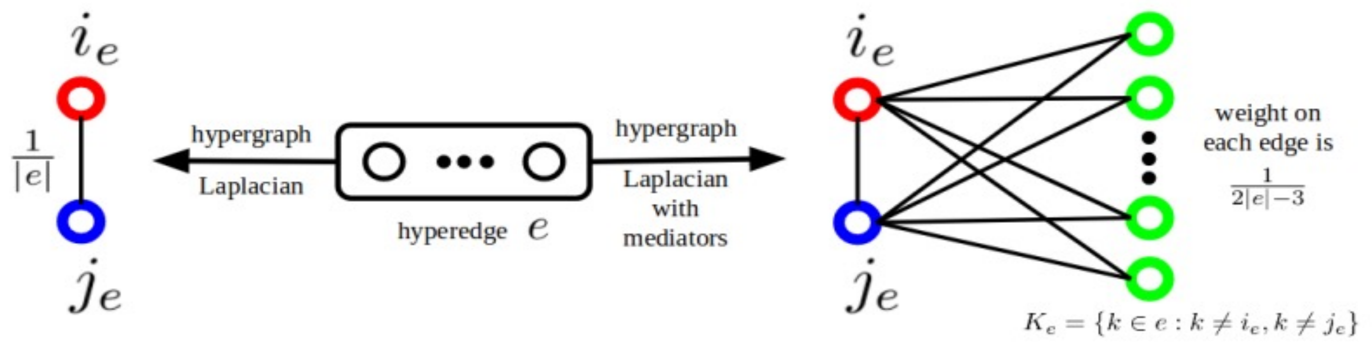


Figure 2: Hypergraph Laplacian [8] vs. the generalised hypergraph Laplacian with mediators [7]. Our approach requires at most a linear number of edges (1 and $2|e|-3$ respectively) while HGNN [17] requires a quadratic number of edges for each hyperedge.

FastHyperGCN

- use just the initial features X (without the weights) to construct the hypergraph Laplacian matrix(with mediators)
  - training time is much less than other methods.
  - the results performance is worse than HyperGCN

Model↓ Metric →	Training time	Density	Training time (DBLP)	Training time (Pubmed)
HGNN	170s	337	0.115s	0.019s
FastHyperGCN	**143**s	**352**	**0.035**s	**0.016**s

Table 1: average training time of an epoch (lower is better)

Result Comparison

Table 4: Results of SSL experiments. We report mean test error $\pm$ standard deviation (lower is better) over 100 train-test splits. Please refer to section 5 for details.

Data	Method	DBLP co-authorship	Pubmed co-citation	Cora co-authorship	Cora co-citation	Citeseer co-citation
$\mathcal{H}$	CI	54.81 ± 0.9	52.96 ± 0.8	55.45 ± 0.6	64.40 ± 0.8	70.37 ± 0.3
$\mathbf{X}$	MLP	37.77 ± 2.0	30.70 ± 1.6	41.25 ± 1.9	42.14 ± 1.8	41.12 ± 1.7
$\mathcal{H}, \mathbf{X}$	MLP + HLR	30.42 ± 2.1	30.18 ± 1.5	34.87 ± 1.8	36.98 ± 1.8	37.75 ± 1.6
$\mathcal{H}, \mathbf{X}$	HGNN	25.65 ± 2.1	29.41 ± 1.5	31.90 ± 1.9	$\mathbf{32.41 \pm 1.8}$	$\mathbf{37.40 \pm 1.6}$
$\mathcal{H}, \mathbf{X}$	1-HyperGCN	33.87 ± 2.4	30.08 ± 1.5	36.22 ± 2.2	34.45 ± 2.1	38.87 ± 1.9
$\mathcal{H}, \mathbf{X}$	FastHyperGCN	27.34 ± 2.1	29.48 ± 1.6	32.54 ± 1.8	$\mathbf{32.43 \pm 1.8}$	$\mathbf{37.42 \pm 1.7}$
$\mathcal{H}, \mathbf{X}$	HyperGCN	$\mathbf{24.09 \pm 2.0}$	$\mathbf{25.56 \pm 1.6}$	$\mathbf{30.08 \pm 1.8}$	$\mathbf{32.37 \pm 1.7}$	$\mathbf{37.35 \pm 1.6}$

Table 5: Results (lower is better) on sythetic data and a subset of DBLP showing that our methods are more effective for noisy hyperedges. η is no. of hypernodes of one class divided by that of the other in noisy hyperedges. Best result is in bold and second best is underlined. Please see Section 6.

Method	$\eta = 0.75$	$\eta = 0.70$	$\eta = 0.65$	$\eta = 0.60$	$\eta = 0.55$	$\eta = 0.50$	sDBLP
HGNN	$\mathbf{15.92 \pm 2.4}$	$\mathbf{24.89 \pm 2.2}$	$\mathbf{31.32 \pm 1.9}$	39.13 ± 1.78	42.23 ± 1.9	44.25 ± 1.8	45.27 ± 2.4
FastHyperGCN	28.86 ± 2.6	31.56 ± 2.7	33.78 ± 2.1	$\underline{33.89 \pm 2.0}$	$\mathbf{34.56 \pm 2.2}$	$\mathbf{35.65 \pm 2.1}$	$\underline{41.79 \pm 2.8}$
HyperGCN	$\underline{22.44 \pm 2.0}$	$\underline{29.33 \pm 2.2}$	$\underline{33.41 \pm 1.9}$	$\mathbf{33.67 \pm 1.9}$	$\underline{35.05 \pm 2.0}$	$\underline{37.89 \pm 1.9}$	$\mathbf{41.64 \pm 2.6}$

Conclusion

- This paper introduces a new Hypergraph convlutional network traing method and it’s variations.
- the main idea is to simplify the hypergraph to a simple graph, and then use Laplacian construction for performing GCN.
- For saving the training time, try to use FastHyperGCN, the trainning time is much less than HyperGCN

Questions

- How to use the meditors?
    - store the signals in feature matrix? or weights matrix?
- what is Y represent in cross-entropy.

Thank you