Deep Hypergraph Convolutional Network for Paper Classification

Yifei Zhu[1] , **Guanzi Chen**[2]
[1]Tsinghua-Berkeley Shenzhen Institute Lab2C
[2]Tsinghua-Berkeley Shenzhen Institute Lab2C
b224hisl@yandex.com, cgz20@mails.tsinghua.edu.cn,

Abstract

Hypergraphs provide a natural representation for many complex real-world datasets such as co-authorship, co-citation, email communication, etc. Recently, many hypergraph neural networks are designed to model such data, however, most of them are shadow, limiting the ability to extract information from high-order neighbors. In this paper, we show that deep hypergraph neural networks suffer from *over-smoothing*, which means stacking more layers tends to lead to performance degradation. And inspired by the analysis of over-smoothing problem to GCN, we develop a deep hypergraph neural network called HGCNII to alleviate the problem of over-smoothing, which is an extension of the hypergraph neural network(HGNN) model with two simple yet effective techniques: *Initial residual* and *Identity mapping*. We have demonstrated the effectiveness and improved performance of HGCNII on multiple real-world datasets compared to the state-of-the-arts.

1 Introduction

Graph-based convolution neural networks [Kipf and Welling, 2016] have shown superior performance in many applications in recent years. In traditional graph convolutional neural network methods, the pairwise connections among data are employed. However, it is noted that the data structure in real practice could be beyond pairwise connections and even far more complicated. For example, in many real-world network datasets such as co-authorship, co-citation, email communication, etc., relationships are complex and go beyond pairwise association. Hypergraphs provide a flexible and natural modeling tool to model such complex relationships.

Recently, many hypergraph neural networks are proposed to model data on hypergraphs. HGNN [Feng *et al.*, 2019] has been proposed as the first deep learning method on hypergraph structure, employing hypergraph Laplacian to represent hypergraph form spectral perspective. HGHA [Bai *et al.*, 2019] has applied an attention module on hypergraph convolutional networks. DHGNN [Jiang *et al.*, 2019] proposed a dynamic hypergraph neural network. HyperGCN [Yadati *et al.*, 2019] trains a GCN for semi-supervised learning on hypergraphs. Hyper-SAGNN [Zhang *et al.*, 2019] proposed a self-attention based graph neural network applicable to homogeneous and heterogeneous hypergraphs.

Despite their enormous success, most of the current hypergraph convolutional networks are shadow, achieving their best performance with 2-layer models. Inspired by the analysis of the problem of over-smoothing to GCN models, in this paper, we aim to provide theoretical and empirical evidence to show that hypergraph convolutional networks suffer from over-smoothing, which means stacking more layers tends to lead to performance degradation. Furthermore, to tackle this issue, we develop a deep hypergraph neural network called HGCNII to alleviate the problem of over-smoothing. More specifically, this method is an extension of the hypergraph neural network(HGNN) model with two simple yet effective techniques: *Initial residual* and *Identity mapping*. At each layer, initial residual constructs a skip connection from the input layer, while identity mapping adds an identity matrix to the weight matrix. To evaluate the performance of the proposed HGCNII model, we provide theoretical analysis for HGCNII to show the effectiveness of the two techniques. In particular, we prove that a K-layer HGCNII model can express a polynomial spectral filter of order K with arbitrary coefficients. We have conducted experiments on co-authorship and co-citation networks classification tasks. The experimental results on five datasets demonstrate the two surprisingly simple techniques prevent over-smoothing and improved the performance of HGCNII consistently as we increase its network depth.

Our main contributions are summarized as follows:

1.We provide theoretical and empirical evidence to show that hypergraph convolutional networks suffer from the problem of over-smoothing.

2.We propose the HGCNII, an extension of the vanilla HGNN model with two simple yet effective techniques: *Initial residual* and *Identity mapping* to relieve the over-smoothing problem of hypergraph neural networks.

3.We have conducted experiments on co-authorship and co-citation networks classification tasks. The experimental results indicated that the performance of HGCNII consistently as we increase its network depth.

2 Preliminaries

Hypergraph neural networks (HGNN). Given a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, $\mathcal{V}$ is the vertex set, $\mathcal{E}$ is the hyperedge set, $\mathbf{W}$ is the weight matrix of hyperedges. The hypergraph $\mathcal{G}$ can be denoted by a $|\mathcal{V}| \times |\mathcal{E}|$ incidence matrix $\mathbf{H}$, with entries defined as

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

For a vertex $v \in \mathcal{V}$, its degree is defined as $d(v) = \sum_{e \in \mathcal{E}} w(e)h(v,e)$. For an edge $e \in \mathcal{E}$, its degree is defined as $\delta(v) = \sum_{v \in \mathcal{V}} w(e)h(v,e)$. Further, $\mathbf{D}_e$ and $\mathbf{D}_v$ denote the diagonal matrices of the edge degrees and the vertex degrees, respectively. [Zhou *et al.*, 2007] introduce the normalized hypergraph laplacian matrix as $\Delta = \mathbf{I} - P$, where $\mathbf{P} = \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}}$.

[Feng *et al.*, 2019] propose a hypergraph convolution based on the spectral convolution on the hypergraph. Δ is a symmetric positive semi-definite matrix with eigen decomposition $\Delta = \Phi\Lambda\Phi^T$. Then, the spectral convolution of a signal $x = (x_1, \cdots, x_n)$ and a filter g can be denoted as $g * x = \Phi((\Phi^T g) \odot (\Phi^T x)) = \Phi g(\Lambda)\Phi^T x$. Following the suggestion on [Defferrard *et al.*, 2016] and [Kipf and Welling, 2016], the hypergraph convolution operation can be further approximated by the K-th order polynomial of Laplacians $g*x \approx \Phi(\sum_{k=0}^{K} \theta_k \Lambda^k)\Phi^T x = \left(\sum_{k=0}^{K} \theta_k \Delta^k\right) x$. Similar to the derivation of graph convolution [Kipf and Welling, 2016], the convolution operation can be simplified to the following expression:

$$g * x = \theta \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}} x \tag{2}$$

Then, recall that $\mathbf{P} = \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}}$, we can build a hypergraph convolutional layer $f(\mathbf{X}, \mathbf{W}, \Theta)$ in the following formulation

$$\mathbf{X}^{(l+1)} = \sigma(P\mathbf{X}^{(l)}\Theta^{(l)}). \tag{3}$$

Simplifying Graph Convolutional Networks: SGC. SGC [Wu *et al.*, 2019] shows that by stacking K layers, GCN corresponds to a fixed polynomial filter of order K on the graph spectral domain. In particular, applying a K-layer GCN to a signal x corresponds to $\left(I_n - \tilde{\mathbf{L}}\right)^K x$, i.e., a polynomial filter $\left(\sum_{l=0}^{K} \theta_l \tilde{\mathbf{L}}^l\right) x$ with fixed coefficient θ, where $\tilde{\mathbf{L}}$ is the normalize Laplacian matrix of the graph.

Deep Graph Convolutional Networks: GCNII. GCNII [Chen *et al.*, 2020] is the closet GCN model, which either achieves state-of-the-art performance on graph classification task or is specifically designed for solving the over-smoothing problem of graph neural networks. GCNII extends the vanilla GCN with two simple techniques: Initial residual connection and Identity mapping. It is theoretically and empirically proved that the two simple techniques are effective to relieve over-smoothing.

3 HGCII Model

As introduced in the above section, by stacking K layers, the HGNN model simulates polynomial filter $\sum_{l=0}^{K} \theta_l \Delta^l)x$ of order K with fixed coefficients θ on the hypergraph spectral domain. The fixed coefficients limits the expressive power of a multi-layer HGNN model and thus leads to over-smoothing. Inspired by the design of GCNII [Chen *et al.*, 2020], we extend HGNN with two simple techniques: Initial residual connection and Identity mapping, to make HGNN be able to express a K order polynomial filter with arbitrary coefficients.

Specifically, at each layer, initial residual constructs a skip connection from the input layer, while identity mapping adds an identity matrix to the weight matrix. Then, the hypergraph convolutional layer of HGCNII is defined as :

$$\mathbf{X}^{(l+1)} = \sigma\left(\left((1-\alpha_l)\mathbf{P}\mathbf{X}^{(l)} + \alpha_l\mathbf{X}^{(0)}\right)\left((1-\beta_l)\mathbf{I}_n + \beta_l\mathbf{W}^{(l)}\right)\right), \tag{4}$$

where α_l and β_l are two hyperparameters.

4 Theorem

Theorem 1 Hypergraph neural networks suffer from the problem of over-smoothing. Specifically, the Dirichlet energy of embeddings will converge to zero, resulting in the loss of discriminative power.

Proof. In this paper, we explain the phenomenon of over-smoothing based on Dirichlet energy. In mathematics, the Dirichlet energy is a measure of how variable a function is. Given a function $F = (f_1, ..., f_m)$, and the domain of definition $D \subset R^n \to R^m$, the Dirichlet energy is to measure how much information expressed by the function F over D:

$$E(F) = \int_D |dF(p)|^2 dp = \sum_{i=1}^{m} \int_D |\nabla f_i(p)|^2 dp \tag{5}$$

In many applications, they seek functions that are "as smooth as possible", amounting to seeking functions F minimizing the Dirchlet energy. But in deep learning field, we hope the model to be more expressive over the domain of definition. Intuitively, the bigger the Dirichlet energy is, The stronger the model's expression ability is. With some assumptions on the weight matrix Θ of GCN, We can prove that the Dirichlet energy decreases with the respect to the number of layers.

We prove this in three steps:
1.The Dirchlet energy of HGCN decreases when multiplying the attribute matrix of the hypergraph P.
2.The Dirchlet energy of HGCN decreases after ReLU activation.
3.The Dirchlet energy of HGCN decreases after multiplying the weight matrix.

First step We follow the definition of the Dirchlet energy E(f) in graph from other literature [Cai and Wang, 2020]. E(f) of scalar funtion $f \in R^{N*1}$ on the hypergraph is defined as:

$$E(f) = f^T \Delta f \tag{6}$$

Let us denot the eigenvalue of Δ by $\lambda_1, \lambda_2, ..., \lambda_N$ and the associated eigenvectores are $v_1, v_2, ..., v_n$, where $c_i \in R$,

$$\begin{aligned} E(Pf) &= f^T(I_N - \Delta)^T\Delta(I_N - \Delta)f \\ &= f^T(I_N - \Delta)\Delta(I_N - \Delta)f \\ &= \sum c_i^2\lambda_i(1-\lambda_i)^2 \\ &\leq (1-\lambda_{min})E(f) \leq E(f) \end{aligned} \tag{7}$$

Second step When the model is activated by ReLU function, we can prove that $E(\sigma(X)) \leq E(X)$. We first prove it holds for a scalar function f and then extend it to vector field X. We extend formul(6) by $E(f) = \sum_{i,j\in V}(\frac{f_i}{\sqrt{1+d_i}} - \frac{f_j}{\sqrt{1+d_j}})^2$, where d_i are the diagonal elements in D_v. For any hypergraph:

$$\begin{aligned} &|\frac{1}{\sqrt{1+d_i}}f_i - \frac{1}{\sqrt{1+d_j}}f_j| \\ &\geq |\sigma(\frac{1}{\sqrt{1+d_i}}f_i) - \sigma(\frac{1}{\sqrt{1+d_j}}f_j)| \\ &= |\frac{1}{\sqrt{1+d_i}}\sigma(f_i) - \frac{1}{\sqrt{1+d_j}}\sigma(f_j)| \end{aligned} \tag{8}$$

Because vector functions are superpositions of scalar functions, it's sure that it holds for vector function after completing the proof in scalar field.

Third step Expand E(XΘ) in matrix form,

$$\begin{aligned} E(X\Theta) &= tr(\Theta^T X^T \Delta X \Theta) \\ &= tr(X^T\Delta X\Theta\Theta^T) \\ &\leq tr(X^T\Delta X)\sigma_{max}(\Theta\Theta^T) \\ &= E(X)||\Theta^T||_2^2 \end{aligned} \tag{9}$$

Where σ_{max} denotes the largest eigenvalue.
Therefore, by these three steps, we can conclude that for any layer l$\in N_+$,

$$\begin{aligned} E(f_l(X)) &= E(\sigma P(\sigma P(\sigma P(...)\Theta^{l-2})\Theta^{l-1})\Theta^l) \\ &\leq E(\sigma * (1-\lambda_{min})^l X)\prod_1^l ||\Theta^l||_2^2 \end{aligned} \tag{10}$$

when $l \to \infty$, the left side in (10) will converge to zero. Now we have proved that the Dirichlet energy will decrease when stacking more layers, which explains the phenomenon of over-smoothing.

Theorem 2 A K-layer HGCNII can express a K order polynomial filter $\sum_{k=0}^K \theta_k\Delta^k$ with arbitrary coefficients θ.

Proof. Similar to the derivation in SGC [Wu *et al.*, 2019], we can obtain that by stacking K layers, the HGNN model simulates a polynomial filter$\sum_{k=0}^K \theta_k\Delta^k$ with **fixed** coefficients θ. Next, we prove that by the two simple techniques: Initial residual and Identity mapping, a $K-$ layer HGCNII model can express a K order polynomial filter with **arbitrary** coefficients.

For simplicity, we assume the signal vector x to be non-negative. Note that we can convert x into a non-negative input layer $\mathbf{X}^{(0)}$ by a linear transformation $\mathbf{X}^{(0)} = \mathbf{P}x$. We consider a weaker version of HGCNII by fixing $\alpha_l = 0.5$ and fixing the weight matrix $(1-\beta_l)\mathbf{I}_n + \beta_l\mathbf{W}^{(l)}$ to be $\gamma_l\mathbf{I}_n$, where γ_l is a learnable parameter. We have

$$\mathbf{X}^{(l+1)} = \frac{1}{2}\sigma\left(P(\mathbf{X}^{(l)} + x)\gamma_l\mathbf{I}_n\right) \tag{11}$$

Since the input feature x is non-negative, we can remove coefficient $\frac{1}{2}$ and ReLU operation for simplicity:

$$\mathbf{X}^{(l+1)} = \gamma_l P(\mathbf{X}^{(l)} + x) \quad = \gamma_l\left((\mathbf{I}_n - \Delta)(\mathbf{X}^{(l)} + x)\right) \tag{12}$$

Consequently, we can express the final representation after K layers HGCNII as:

$$\mathbf{X}^{(l+1)} = \left(\sum_{l=0}^{K-1}\left(\prod_{k=K-l-1}^{K-1}\gamma_k\right)(\mathbf{I}_n - \Delta)^l\right)x. \tag{13}$$

On the other hand, as introduced in the begining, a polynomial filter of $K-$layer HGCNII can be expressed as:

$$\begin{aligned} \left(\sum_{k=0}^{K-1}\theta_k\Delta^k\right)x &= \left(\sum_{k=0}^{K-1}\theta_k\left(\mathbf{I}_n - (\mathbf{I}_n - \Delta)\right)^k\right)x \\ &= \left(\sum_{k=0}^{K-1}\theta_k\left(\sum_{l=0}^{k}(-1)^l\binom{k}{l}(\mathbf{I}_n - \Delta)^l\right)\right)x \\ &= \left(\sum_{l=0}^{K-1}\left(\sum_{k=l}^{K-1}\theta_k(-1)^l\binom{k}{l}(\mathbf{I}_n - \Delta)^l\right)\right)x \end{aligned} \tag{14}$$

To show that HGCNII can express an arbitrary K-order polynomial filter, we need to prove that there exists a solution γ_l, l=0, $\cdots$,$K-1$ such that the corresponding coefficients of $(\mathbf{I}_n - \Delta)^l$ in (13) and (14) are equivalent. More precisely, we need to show the following equation system has a solution γ_l, l=0, $\cdots$,$K-1$.

$$\prod_{k=K-l-1}^{K-1}\gamma_k = \sum_{l=0}^{k}\theta_k(-1)^l\binom{k}{l}, k = 0, \cdots, K-1. \tag{15}$$

Note that we can solve the equation system by

$$\gamma_{K-l-1} = \sum_{k=l}^{K-1}\theta_k(-1)^l\binom{k}{l} \Big/ \sum_{k=l-1}^{K-1}\theta_k(-1)^{l-1}\binom{k}{l-1} \tag{16}$$

for l=0, $\cdots$,$K-2$ and $\gamma_{K-1l} = \sum_{k=0}^{K-1}\theta_k$. This prove that a K-layer HGCNII can express the K-th order polynomial filter with arbitrary coefficients.

5 Experiments

In this section, we evaluate the performance of HGCNII against the state-of-art hypergraph neural network models on a wide variety of open hypergraph datasets.

Dataset and experimental setup. In this experiment, the task is to classify co-authorship and co-citation data. We use three co-citation network datasets: Cora, Pubmed and Citeseer, and two co-authorship network datasets: Cora, DBLP, which are used in recent hypergraph neural network [Yadati *et al.*, 2019]. In these co-citation datasets, nodes correspond to documents, and edges correspond to citation; each node feature corresponds to the bag-of-words representation of the document and belongs to one of the academic topics. And in the co-authorship datasets, edges indicate there exists co-authors among those documents. Statistics of the datasets are summarized in Table 1.

For baselines, We compare HGCNII with state-of-the-art hypergraph neural networks: HGNN [Feng *et al.*, 2019] and HyperGCN [Yadati *et al.*, 2019], and a backbone Multilayer perceptron (MLP) that does not use the hypergraph structure to make predictions.

We use the Adam SGD optimizer with a learning rate of 0.01 and early stopping with a patience of 200 epochs. We set $\alpha_l = 0.3$, and $\beta_l \approx \frac{\lambda}{l}$ with $\lambda = 1.5$.

Results on Node Classification. Table 2 shows that on the five datasets, HGCNII consistently improves as we increase the number of layers, while HGNN and HyperGCN drop rapidly when the number of layers exceeds 2. It is noted that MLP has always got low accuracy, indicating that hypergraph structure is effective to make predictions. And HGCNII outperform SOTA HyperGCN on Cora, Pubmed and DBLP datasets.

Hypernode Visualizaiton. We also draw t-SNE and feature maps for hypernode representations. Figure 1 is the t-SNE visualization of learned node representations, which include original features, different layers of HGNN and different layers of HGCNII on Cora dataset. Different colors represent different node classes. As we can observe that the classes are separated to a good extend on 2-layer HGNN, indicating that it learns meaningful embeddings which distinguish the different classes. And it shows the effectiveness of hypergraph convolutional networks. However, as the number of layers increases, the embeddings of different classes also tend to be similar and lead to indistinguishable features. Instead, HGCNII consistently get separate results as we increase the number of layers. Overall, the results suggest that hypergraph neural networks suffer from severe over-smoothing problems, while HGCNII can effectively relieve the over-smoothing problem and extend the HGNN into a truly deep model. Figure 2 is the feature map of learned embedding of the first 50 hypernodes. It can come to the same conclusion.

6 Conclusion

We analyze the over-smoothing problem of hypergraph neural networks. We propose HGCNII, a simple and deep hypergraph neural network model that can prevent the problem of over-smoothing, by initial residual connection and identity mapping. Experiments show that the deep HGCNII model relieve the problem of over-smoothing and achieves SOTA results on node classificaiton problem on hypergraphs.

Table 1: Summary of classification accuracy(%) results with various depths.(Red is the best, and blue is the second.

Dataset	Method	Layers					
		2	4	8	16	32	64
Cora (co-authorship)	MLP	35.16	35.04	34.37	34.37	34.41	33.25
	HyperGCN	68.81	57.87	15.5	OOM	OOM	OOM
	HGNN	68.96	65.89	29.95	29.98	29.95	29.98
	HGCNII	72.04	74.45	74.34	74.96	73.87	74.03
DBLP (co-authorship)	MLP	47.31	47.19	47.16	46.57	OOM	OOM
	HyperGCN	68.76	54.36	15.95	OOM	OOM	OOM
	HGNN	88.32	87.72	83.67	27.61	27.63	OOM
	HGCNII	88.56	89.42	89.4	89.5	OOM	OOM
Cora (co-citation)	MLP	35.16	35.05	34.37	34.37	34.41	33.25
	HyperGCN	68.11	57.75	7.63	OOM	OOM	OOM
	HGNN	47.2	47.16	14.56	13.16	16.12	13.2
	HGCNII	68.15	69.12	69.67	70.05	70.48	70.17
Pubmed (co-citation)	MLP	44.48	41.52	43.33	40.24	43	42.34
	HyperGCN	76.57	52.52	20.94	OOM	OOM	OOM
	HGNN	30.82	31.76	24.06	23.25	24.29	21.75
	HGCNII	76	76.34	76.46	76.79	77.45	77.57
Citeseer (co-citation)	MLP	35.15	35.59	34.98	34.88	34.66	34.92
	HyperGCN	64.4	51.04	18.27	OOM	OOM	OOM
	HGNN	41.49	41.43	35.44	21.61	21.39	21.39
	HGCNII	62.07	61.97	63.3	62.1	62.2	63.11

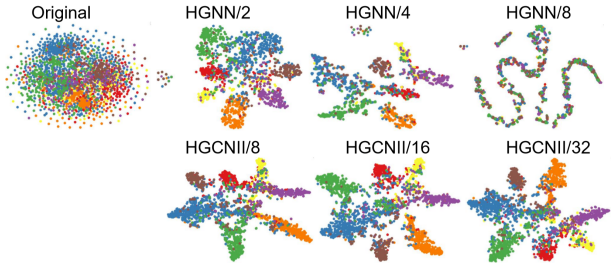

Figure 1: t-SNE visualization of the hypernode representations on Cora. Colors represent node classes.

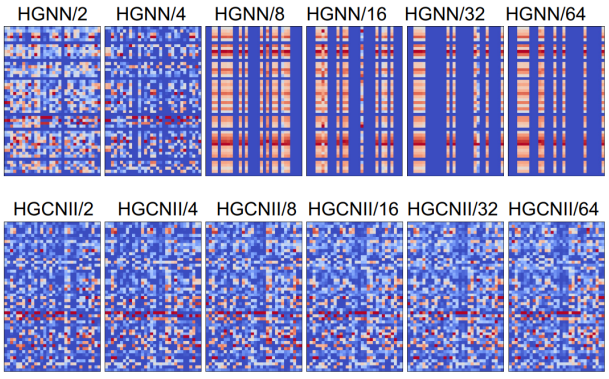

Figure 2: Feature maps of hypernode representations on Cora.

References

[Bai *et al.*, 2019] Song Bai, Feihu Zhang, and Philip HS Torr. Hypergraph convolution and hypergraph attention. *arXiv preprint arXiv:1901.08150*, 2019.

[Cai and Wang, 2020] Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

[Chen *et al.*, 2020] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, pages 1725–1735. PMLR, 2020.

[Defferrard *et al.*, 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29:3844–3852, 2016.

[Feng *et al.*, 2019] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3558–3565, 2019.

[Jiang *et al.*, 2019] Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. Dynamic hypergraph neural networks. In *IJCAI*, pages 2635–2641, 2019.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Wu *et al.*, 2019] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. Simplifying graph convolutional networks. *arXiv preprint arXiv:1902.07153*, 2019.

[Yadati *et al.*, 2019] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. In *Advances in Neural Information Processing Systems*, pages 1511–1522, 2019.

[Zhang *et al.*, 2019] Ruochi Zhang, Yuesong Zou, and Jian Ma. Hyper-sagnn: a self-attention based graph neural network for hypergraphs. *arXiv preprint arXiv:1911.02613*, 2019.

[Zhou *et al.*, 2007] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in neural information processing systems*, pages 1601–1608, 2007.