SIMPLE AND DEEP GRAPH ATTENTION NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph Attention Networks (GATs) and Graph Convolutional Neural Networks (GCNs) are two state-of-the-art architectures in Graph Neural Networks (GNNs). It is well known that both models suffer from performance degradation when more GNN layers are stacked, and many works have been devoted to address this problem. We notice that main research efforts in the line focus on the GCN models, and their techniques cannot well fit the GATs models due to the inherent difference between these two architectures. In GATs, the attention mechanism ignores the overwhelming propagation from certain nodes as the number of layers increases. To sufficiently utilize the expressive power of GATs, we propose a new version of GAT named Layer-wise Self-adaptive GAT (LSGAT), which can effectively alleviate the oversmoothing issue in deep GATs. We redesign the attention coefficients computation mechanism adaptively adjusted by layer depth, which considers both immediate neighboring and non-adjacent nodes from a global view. The experimental evaluation confirms that LSGAT consistently achieves better results on node classification tasks over relevant counterparts.

1 INTRODUCTION

Graph neural networks (GNNs) (Kipf & Welling, 2016; Veličković et al., 2017; Hamilton et al., 2017) have emerged as a promising tool for analyzing graph data, such as biochemical networks (You et al., 2020; Wang et al., 2021), social networks (Huang et al., 2019; Dong et al., 2021), and academic networks (Gao et al., 2018), etc. However, the performance of most GNNs, *e.g.,* graph convolutional networks (GCNs), would promptly degrade when stacking up the non-linear GNN layers. Towards this phenomenon, research attempts have been made to deepen current GNN architectures and understand their expressive power. Theoretical analysis investigates the deep GNNs in the perspective of expressive power (Yan et al., 2021; Li et al., 2018; Oono & Suzuki, 2019; Huang et al., 2020; Cai & Wang, 2020), training difficulty (Dasoulas et al., 2021; Huang et al., 2021; Luan et al., 2020) and generalization (Cong et al., 2021). There are three main categories (Chen et al., 2022) of deepening techniques that are based on skip connection (Xu et al., 2018), graph normalization (Zhou et al., 2020a; Dasoulas et al., 2021), and random dropping (Rong et al., 2019) respectively.

Nevertheless, existing research efforts (Yan et al., 2021; Li et al., 2018; Oono & Suzuki, 2019; Huang et al., 2020; Cai & Wang, 2020; Huang et al., 2021; Luan et al., 2020; Cong et al., 2021; Chen et al., 2022; Xu et al., 2018; Zhou et al., 2020a) mainly focus on the theoretical analysis and deepening techniques on GCNs, and few of them (Dasoulas et al., 2021) consider the attention calculation mechanism in deep GATs. Their techniques cannot well fit the GATs models due to the inherent difference between these two architectures. Under the attention calculation mechanism, GATs could compute the coefficients implicitly rather than explicitly as GCNs do, and we can use more information besides the topological information to determine each node's weight. In this paper, to alleviate the performance degradation of deep GAT, we first make a hypothesis that the oversmoothing issue happened in deep GAT is caused by the overwhelming propagation from nodes with large degrees as the number of layers increases. Specifically, the node with higher degree will be aggregated via more paths that are exponentially increased w.r.t. growing model depth, which will inevitably lead to oversmoothing problem when the number of layers is large enough. To verify the hypothesis, in Section 3, we use two intuitive but effective neural network architectures that constraint the propagation of nodes with large degree in deep layers, and compare them with vanilla GAT. As a result, after the preliminary verification of the hypothesis, the performance of the intuitive architectures is remarkable but still not satisfactory enough.

Based on the results, it remains elusive how to properly train the GAT to sufficiently utilize the expressive power of the model. Dasoulas et al. (2021) has proposed a normalization technique based on Lipschitz continuity, which alleviates the gradient vanishing/explosion in deep GAT. In spite of Lipschitz normalization, the internal computing mechanism of GAT has not changed, coefficients are always limited to adjacent nodes and layers, which means overwhelming propagation problem still exists in deep GAT with Lipschitz normalization. Particularly, without specifically designed framework, GAT is not capable of avoiding to be affected by the oversmoothing issue due to its computation mechanism, *e.g.,* overwhelming propagation from the large-degree nodes. To better address this issue, we propose a simple and deep version of GAT named Layer-wise Self-Adaptive GAT (LSGAT). LSGAT is a newly designed attention coefficients computation mechanism, which considers the influence of both adjacent and non-adjacent nodes. LSGAT adaptively adjusts the attention coefficients with the layer depth and regularizes the coefficients for nodes with large degree, which eventually alleviate potential oversmoothing issue.

Contributions. The contributions of this paper can be summarized as follow:

- We demonstrate that overwhelming propagation from large degree nodes should be considered, which could relieve the oversmoothing problem happened in deep GAT.
- Once confirmed the overwhelming propagation in deep GAT, we propose a new version of GAT named LSGAT, which considers both neighboring and non-adjacent nodes based on high-order proximity in aggregation process, to train GAT properly and alleviate the oversmoothing issue.
- The effectiveness and versatility of LSGAT have been validated in our extensive experimental results for node classification on real-world datasets. Based on GAT, compared with the methods specifically designed for deep GAT, and multiple techniques designed for deep GNNs, LSGAT consistently exhibits superior performance. Even compared with the methods designed for relieving the oversmoothing problem which equiped with GCN and ChebyNet, our LSGAT outperforms evidently.

Roadmap. The rest of the paper is organized as follows: The preliminaries are introduced in Section 2. We then introduce and verify the overwhelming propagation problem in Section 3. The details of our proposed novel LSGAT is introduced in Section 4. The experimental results are reported in Section 5. Section 6 presents the related works, and Section 7 concludes the paper.

2 PRELIMINARIES

Notations and setup. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ denotes the number of vertices, where each node $v_i \in \mathcal{V}$ is associated with a feature vector $h_i \in \mathbb{R}^d$, a layer outputs a new set of node representations $h_i' \in \mathbb{R}^{d'}$, and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, where $(v_j, v_i) \in \mathcal{E}$ denotes an edge from node v_j to node v_i, and its node degree represents as d_i. Let $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ denote the adjacent matrix, and $\hat{\boldsymbol{A}} \in \mathbb{R}^{n \times n}$ denote the adjacent matrix with self-loop, *i.e.,* $\hat{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_n$.

2.1 GRAPH ATTENTION NETWORKS

Different from GCN and many other popular GNN architectures (Hamilton et al., 2017; Chen et al., 2020) that weigh all neighbors with equal importance (*e.g.,* mean and max-pooling as aggregation) , GAT (Veličković et al., 2017) enables (implicitly) specifying different weights to different nodes in a neighborhood. A scoring function e: $\mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}$ computes a score for every edge (v_j, v_i), which indicates the importance of the features of the neighbor v_j to node v_i:

$$e\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right) = \text{LeakyReLU}\left(\boldsymbol{a}^{\mathrm{T}} \cdot [\boldsymbol{W}\boldsymbol{h}_i || \boldsymbol{W}\boldsymbol{h}_j]\right), \tag{1}$$

where $\boldsymbol{a} \in \mathbb{R}^{2d'}$, $\boldsymbol{W} \in \mathbb{R}^{d' \times d}$ are trainable parameters, and $||$ denotes vector concatenation. These attention scores are normalized across all neighbors using softmax, and the attention function is defined as:

$$\alpha_{ij} = \text{softmax}_j\left(e\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right) = \frac{exp\left(e\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right)}{\sum_{v_{j'} \in \mathcal{N}_i} exp\left(e\left(\boldsymbol{h}_i, \boldsymbol{h}_{j'}\right)\right)} \tag{2}$$

We denote the coefficient matrix, whose entries are α_{ij}, if $(v_i, v_j) \in \mathcal{E}$, and 0 otherwise, as $\boldsymbol{C} \in \mathbb{R}^{n \times n}$. Then, GAT computes a weighted average of the transformed features of the neighbor nodes followed by a non-linearity σ as the new representation of v_i, using the normalized attention coefficients:

$$\boldsymbol{h}_i' = \sigma\left(\sum_{v_j \in \mathcal{N}_i} \alpha_{ij} \cdot \boldsymbol{W}\boldsymbol{h}_j\right), \tag{3}$$

where $\boldsymbol{h}_i'$ denotes the representation of node v_i in the next layer. In this paper, we refer to Equation 1 to 3 as the computation of each layer in an L-layer GAT.

3 Overwhelming Propagation Problem in Deep GAT

In this section, we define and investigate the *overwhelming propagation problem* in the node classification task of deep GAT. GAT computes the attention coefficients α_{ij} for node v_i only based on the feature and topological relation with its neighbor nodes $v_j \in \mathcal{N}_i$. It is well known that the performance of GAT will decrease as the number of layers increases, and the phenomenon could be attributed to oversmoothing problem and overcorrelation problem (Jin et al., 2022). Specifically in deeper GAT, the unique characteristics of the attention coefficients computation mechanism, which have not been considered in the previous work: the attention mechanism ignores the overwhelming propagation from nodes with large degree as model depth increases. That is, the node with higher degree will be aggregated via more paths that are exponentially increased *w.r.t.* growing model depth, and hence GAT is more likely to suffer from oversmoothing issue. Here, we hypothesize that the oversmoothing problem happened in deep GAT could be relieved by extra consideration to constraint the propagation of nodes with large degree in the coefficients calculation mechanism, and we empirically verify that below.

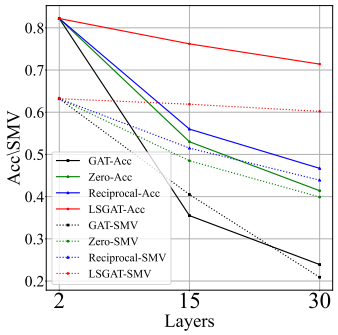


Figure 1: Test accuracy and SMV between different architectures based on Cora dataset and equipped with GAT. Zero means Zero-GAT, Reciprocal means Reciprocal-GAT. The smaller SMV is, the smoother the node representations are.

Experimental Setup. We conduct experiments to observe the test accuracy and SMV (Liu et al., 2020). Specifically, SMV uses normalized node representations to compute their Euclidean distance, and measures the oversmoothing problem in GNNs, the smaller SMV is, the smoother the node representations are. To validate whether we can relieve the oversmoothing problem in GAT by constraining the propagation of the nodes with large degree, our LSGAT and two intuitive GAT variants are compared here: Zero-GAT and Reciprocal-GAT. Zero-GAT means that the top 20% largest degree nodes will not be aggregated any more from the third layer. Reciprocal-GAT means that the Equation 1 will be updated as $e\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right) = \text{LeakyReLU}\left(\boldsymbol{a}^{\mathrm{T}} \cdot \frac{1}{d_j} \cdot [\boldsymbol{W}\boldsymbol{h}_i || \boldsymbol{W}\boldsymbol{h}_j]\right)$, if the given node v_i is linked with node v_j, which belongs to the top 20% largest degree nodes from the third layer. Dataset used here is Cora (Sen et al., 2008), which follows the data split way used in Kipf & Welling (2016). The experimental results are the average of five random experiments, and the parameter tuning space is unified.

Overwhelming Propagation Problem. From Figure 1, we can see how seriously GAT is prone to be oversmoothing by the nodes with large degree in deeper layers. Compared with the performance with two layers, both the test accuracy and SMV of GAT are dramatically decreased as the number of layers increases. The intuitive architectures named Zero-GAT and Reciprocal-GAT both evidently improve the performance of basic GAT. Specifically, compared with vanilla GAT, the test accuracy and SMV of Reciprocal-GAT are both improved by nearly 80% (in percentage) as shown in Figure 1. The experiments based on intuitive models further verify the hypothesis. The performance of Zero-GAT and Reciprocal-GAT is remarkable but still not satisfactory enough. Followed this line, how to better redesign the coefficients calculation mechanism in GAT is a critical challenge.

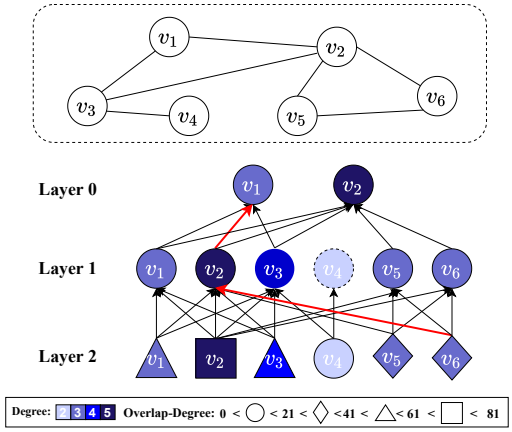


Figure 2: The computation trees rooted at v_1 and v_2 in a 2-layer GNN. Color and shape suggest the degree and *overlap-degree* of each node.

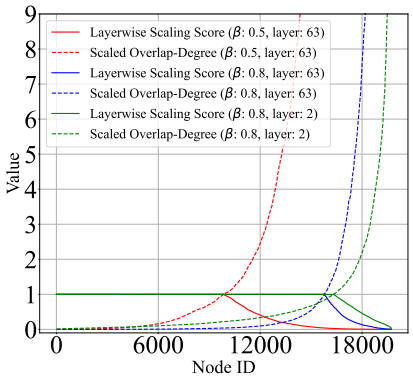


Figure 3: The layer-wise scaling score (lss) and scaled overlap-degree (d^*_{over}) varying number of layers l and hyperparamter β for nodes in Pubmed dataset.

4 Layer-wise Self-adaptive GAT

We first introduce two important definitions to demonstrate our proposed LSGAT.

Definition 1. *(Cong et al., 2021) Let $\mathcal{T}_i^L$ denote the L-layer computation tree rooted at node v_i, which represents the structured L-hop neighbors of node v_i, where the children of any node v_j in the tree are the nodes in $\mathcal{N}(i)$.*

Figure 2 illustrates an example of computation trees $\mathcal{T}_1^L$ and $\mathcal{T}_2^L$ rooted at nodes v_1 and v_2 with $L = 2$. Please note that these computation trees consider the self-loop of nodes. The colors of nodes in Figure 2 indicate the degree of nodes, *i.e.,* darker color suggests higher degree. The shape of each node illustrate the *overlap-degree* of the corresponding node which is defined as follows:

Definition 2. *Overlap-degree $d^{(l)}_{over}(j)$ is the the number of paths that will be affected by the node v_j in layer l through GNNs propagation processes, i.e., $\boldsymbol{D}^{(l)}_{over} = \sum_{i'=1}^{n} \hat{\boldsymbol{A}}^{l}_{i'j} = \left[d^{(l)}_{over}(j)\right]_{n\times 1}$.*

4.1 Proposed Technique for Addressing Oversmoothing

It is well known that the information can also be aggregated to the non-adjacent nodes through layers of GNNs, *e.g.,* in Figure 2, v_6 is not the neighbor of v_1, but its information is propagated to v_1 after two layers via the computation path $v_6 \rightarrow v_2 \rightarrow v_1$. The overlap-degree indicates the number of such computation paths, which exponentially grows *w.r.t.* node degree and increasing layers. For example, the overlap-degree of v_2 is 66 after two layers. The nodes with larger overlap-degree tend to be aggregated to more parent nodes via more paths in the computation tree. This phenomenon makes the representations of nodes be similar, and eventually degrade deep GNN's performance. The similar analysis and experimental results can also be found in (Chen et al., 2020; Liu et al., 2020; Li et al., 2018).

Zero-GAT directly enforces the top 20% largest degree nodes not to be aggregated any more from the third layer, which overly restricts the propagation of the nodes with large degree. The potential disadvantages exist in Reciprocal-GAT will be discussed in Section 4.2. In this work, we aim to relieve oversmoothing problem and reduce the redundant information that come from large degree nodes by regularizing their attention coefficients. The regularization is adaptively adjusted with the depth of model to set higher importance to shallow layers. Here, we first give a self-adaptive threshold value τ to identify the nodes with higher overlap-degree.

$$\tau = \boldsymbol{D}^{(l)}_{over}{}^{(\eta^{(l)})\text{-}th} \tag{4}$$

Here, τ is $\eta^{(l)}$-th value of $\boldsymbol{D}_{over}^{(l)}$ in an ascending order. $\eta^{(l)}$ is defined as $\eta^{(l)} = \beta n + \frac{(1-\beta)n}{e^l}$ where $\beta \in [0, 1]$ is a hyperparameter that determines the proportion of nodes to be regularized. Then, the scaled overlap-degree matrix is computed as follows:

$$\boldsymbol{D}_{over}^{(l)*} = \boldsymbol{D}_{over}^{(l)} / \tau \tag{5}$$

With the scaled overlap-degree matrix, the layer-wise scaling score, denoted as lss, is computed with the following equation:

$$lss_{over}^{(l)}(j) = \begin{cases} 1 & d_{over}^{(l)*}(j) \leq 1 \\ 1/d_{over}^{(l)*}(j) & d_{over}^{(l)*}(j) > 1 \end{cases} \tag{6}$$

The layer-wise scaling scores for nodes in Pubmed dataset varying β and layer l are illustrated in Figure 3. Then the layer-wise scaling scores are multiplied with the scores computed in Equation 1, *i.e.,* $\boldsymbol{a}^{\mathrm{T}} \cdot [\boldsymbol{W}\boldsymbol{h}_i || \boldsymbol{W}\boldsymbol{h}_j]$, followed by LeakyReLU and a Softmax function. As a result, the computation of attention coefficient in LSGAT is formulated as follows:

$$\alpha_{ij}^* = \frac{exp\left(\mathrm{LeakyReLU}\left(lss_{over}^{(l)}(j) \cdot \boldsymbol{a}^{\mathrm{T}} \cdot [\boldsymbol{W}\boldsymbol{h}_i || \boldsymbol{W}\boldsymbol{h}_j]\right)\right)}{\sum_{v_{j'} \in \mathcal{N}_i} exp\left(\mathrm{LeakyReLU}\left(lss_{over}^{(l)}(j') \cdot \boldsymbol{a}^{\mathrm{T}} \cdot [\boldsymbol{W}\boldsymbol{h}_i || \boldsymbol{W}\boldsymbol{h}_{j'}]\right)\right)} \tag{7}$$

In this paper, we denote the matrix whose entries are the attention coefficients α_{ij}^* as $\boldsymbol{C}^{*(l)} \in \mathbb{R}^{n \times n}$ at l-th layer. Finally, the representation computation of node v_i in each layer with LSGAT is as:

$$\boldsymbol{h}_i' = \sigma\left(\sum_{v_j \in \mathcal{N}_i} \alpha_{ij}^* \cdot \boldsymbol{W}\boldsymbol{h}_j\right) \tag{8}$$

In matrix format, LSGAT can also be formulated as $\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{C}^{*(l)} \boldsymbol{W}^{(l)} \boldsymbol{H}^{(l)})$.

4.2 Discussion

In this section, we would like to provide further discussions about the characteristics of LSGAT and the comparisons with the existing works.

Choice and utilization of overlap-degree. One may wonder why the overlap-degree is chosen in our model rather than degree. The layer-wise aggregations in GNNs can enlarge the influence of the nodes from deeper layers. If only consider degree of nodes layer-by-layer, the global high-order information cannot be fully utilized in the computation. As the number of layers increases, the extent of overwhelming propagation influence is divergent and should be treated differently and adaptively with the number of layers, and $\eta^{(l)}$ is used in LSGAT. Furthermore, the layer-wise scaling scores based on the overlap-degree could be smoother in terms of continuity as shown in Figure 3 than that based on degree. For example, the proportion of the largest group in Pubmed dataset (Sen et al., 2008) with same degree is 46%, which means the large number of nodes will share the same scaling score in each aggregation. As a comparison, the layer-wise scaling scores based on the overlap-degree are different with each other and helpful for training a discriminative model. Besides, the layer-wise scaling scores can also be directly multiplied with the attention coefficient, *i.e.,* $\alpha_{ij}^* = \alpha_{ij} \cdot lss_{over}^{(l)}(j)$. However, we found that this variant of our model would significantly ignore the information from the nodes with large overlap-degree which are typically of great importance in the graph. Therefore, we provide a "soft" regulation of the attention coefficients based on overlap-degree for these nodes as shown in Equation 7. The comparison in Figure 1 also verifies that the performance of LSGAT is much better than the performance of Reciprocal-GAT.

Comparison with prior works. Zhou et al. (2020a) shared the similar intuition as ours. They claim that most studies focus on performance degradation problem based on immediate neighboring relationship, but ignore the global graph structural information in each layer. Zhou et al. (2020a) tackles the over-smoothing problem by making the representations similar for nodes that are in the same class and differentiating that are not. In GNNs, the information of nodes in a fully connected

graph could be aggregated to all nodes if the network is deep enough, regardless the nodes belong to the same class or not. Thus, in our work, we utilize the global information to regularize the coefficient during aggregation.

The method in Rong et al. (2019) is based on the random drop of edges, which effectively prevents over-smoothing in deep GNNs. Compared with the random dropout, LSGAT is based on the premise of retaining as much information of nodes as possible, while weakens the feature expression of the nodes that are more likely to suffer from the over-smoothing problem, and hence improves the generalization ability of the model.

Versatility of our proposed mechanism. Our proposed LSGAT conducts the regulation on the aggregation process without modifying the model architecture, which allows LSGAT to be integrated with various existing GNN deepening techniques such as graph normalization (Zhou et al., 2020a), GAT-Lip (Dasoulas et al., 2021), and random dropping (Rong et al., 2019) to further improve the performance of deep GNNs. In our experiments, we show that with skip-connection and identity mapping (He et al., 2016b; Chen et al., 2020), LSGAT achieves the state-of-the-art performance for node classification task.

5 EXPERIMENT

In this section, we evaluate the performance of our LSGAT in real-world benchmarks under semi-supervised learning. We first introduce the experimental settings in Section 5.1. In Section 5.2, we compare LSGAT with existing techniques, which are specifically designed for deep GATs on node classification tasks. In Section 5.3, we compare LSGAT with proposed deep GNNs methods and multi-hop works for node classification tasks, which are equipped with GAT model. In Section 5.4, we report the state-of-the-art performance of LSGAT with skip-connection and identity mapping, which validates the versatility of LSGAT. The hyper-parameter study of LSGAT has been reported in Section 5.5. Furthermore, the comparison between LSGAT and other deep GNNs methods, which are also designed to relieve the oversmoothing problem based on GCN and ChebyNet has been put in Appendix A.1. The combination of LSGAT and GAT-Lip has been reported in Appendix A.2.

5.1 EXPERIMENT SETUP

Datasets. Joining the practice of previous work, we evaluate GNN models by performing the node classification tasks on six real-world datasets: Cora, Citeseer, Pubmed (Sen et al., 2008), CoauthorPhysics, CoauthorCS (Shchur et al., 2018), and Ogbn-Arxiv (Hu et al., 2020). The statistics of these datasets and data splits could be found in Appendix B.1.

Baseline Methods and Models. To our best knowledge, GAT-Lip is the only work proposed to improve the performance in deep GAT. The methods try to relieve oversmoothing problem in deep GNNs include: PairNorm (Zhao & Akoglu, 2019), BatchNorm (Ioffe & Szegedy, 2015), DGN (Zhou et al., 2020a), and DropEdge (Rong et al., 2019). As a variant of GATs, MAGNA (Wang et al., 2020) incorporates multi-hop context information into every layer of attention computation. DeCorr (Jin et al., 2022) is proposed to help enable deeper GNNs from feature overcorrelation perspective.

We consider three basic GNN models, GAT (Veličković et al., 2017), GCN (Kipf & Welling, 2016), and ChebyNet (Defferrard et al., 2016). Besides, as one of the state-of-the-art GNN variants, it is meaningful to see whether GCNII will perform better based on our LSGAT. Specifically, in GCNII, the computation in l-th layer is defined as $\boldsymbol{H}^{(l+1)} = \sigma\left(\left((1-\alpha_l)\,\hat{\boldsymbol{A}}_{\odot}\boldsymbol{H}^{(l)} + \alpha_l\boldsymbol{H}^{(0)}\right)\left((1-\beta_l)\,\boldsymbol{I}_n + \beta_l\boldsymbol{W}^{(l)}\right)\right)$, where $\hat{\boldsymbol{A}}_{\odot} = \hat{\boldsymbol{D}}^{-\frac{1}{2}}(\boldsymbol{A}+\boldsymbol{I}_n)\hat{\boldsymbol{D}}^{-\frac{1}{2}}$, α_l and β_l are two hyperparameters that adjust initial residual and identity mapping respectively.

Implementations. We have strictly followed the experiment settings of previous works in each comparison section. Specifically, in Section 5.2, we follow the setting used in Dasoulas et al. (2021), and we directly report the best performance of GAT and GAT-Lip shown in its work. The results of Section 5.3 and Appendix A.1 are based on the best performance and setting reported in (Jin et al., 2022; Zhou et al., 2020a; Wang et al., 2020). Furthermore, the techniques such as ResNet (He et al., 2016a) and LayerNorm (Ba et al., 2016) which were used in MAGNA are removed for a fair comparison. In Section 5.4, we follow the setting used in Chen et al. (2020) and report the best results shown in the work. Detailed parameter settings can be found in Appendix B.2.

Table 1: Summary of classification accuracy $(\%)$ results among deep GAT-based methods. The value of "Variation" shows the accuracy gap between models with 2 and 30 layers.

Dataset	Method	Layers 2	5	10	15	20	25	30	Variation
Cora	GAT	**82.2**	78.9	57.8	35.5	32.2	30.0	**23.9**	58.3
	GAT-Lip	**82.2**	83.3	80.7	78.8	76.6	71.6	**68.8**	13.4
	LSGAT	**82.2**	79.1	77.5	76.4	76.2	74.8	**73.5**	8.7
Citeseer	GAT	**67.6**	65.0	62.9	61.2	60.9	59.9	**59.3**	7.50
	GAT-Lip	**67.1**	65.9	62.6	62.1	60.1	60.9	**59.4**	8.00
	LSGAT	**67.6**	65.5	63.8	60.9	62.4	62.5	**61.2**	6.40
Pubmed	GAT	**76.3**	78.1	64.5	57.4	51.5	48.8	**29.5**	46.8
	GAT-Lip	**77.6**	80.9	75.4	72.4	73.2	67.7	**65.0**	12.6
	LSGAT	**76.5**	77.0	76.9	77.6	77.4	76.2	**72.8**	3.70
Physics	GAT	**93.2**	91.0	88.3	77.0	50.0	15.3	**13.6**	79.6
	GAT-Lip	**93.7**	91.6	90.4	84.2	72.6	71.7	**63.9**	29.8
	LSGAT	**93.2**	92.1	91.7	91.5	91.2	91.0	**87.0**	6.20
Ogbn-arxiv	GAT	**72.2**	72.5	67.8	59.5	53.9	52.9	**31.4**	40.6
	GAT-Lip	**72.0**	72.3	72.4	69.7	67.3	66.8	**62.2**	9.80
	LSGAT	**72.2**	72.7	71.8	70.5	67.9	67.3	**64.4**	7.80

Table 2: Comparison results of test accuracy (%) between GCNII and LSGAT-GCNII.

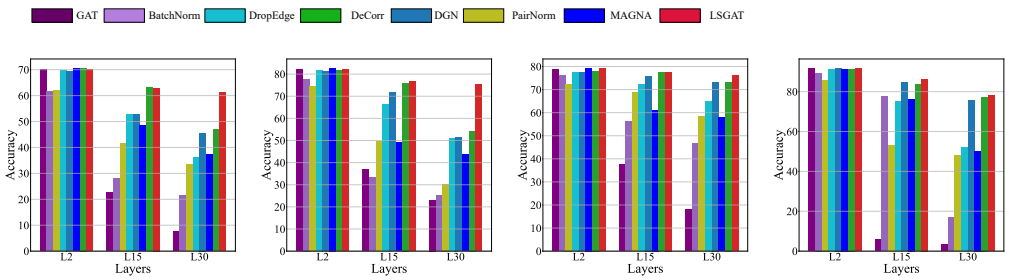
Dataset	Method	Layers 2	4	8	16	32	64	Best
Cora	GCNII	82.2	82.6	84.2	84.6	85.4	85.5	$85.5 \pm 0.5(64)$
	LSGAT-GCNII	83.8	83.9	84.5	85.0	85.6	85.5	$85.6 \pm 0.7(32)$
	Improvement	$+1.6$	$+1.3$	$+0.3$	$+0.4$	$+0.2$	$+0.0$	$+0.1$
Citeseer	GCNII	68.2	68.9	70.6	72.9	73.4	73.4	$73.4 \pm 0.6(32)$
	LSGAT-GCNII	71.6	72.3	73.3	73.2	73.4	72.4	$73.4 \pm 0.8(32)$
	Improvement	$+3.4$	$+3.4$	$+2.7$	$+0.3$	$+0.0$	-1.0	$+0.0$
Pubmed	GCNII	78.2	78.8	79.3	80.2	79.8	79.7	$80.2 \pm 0.4(16)$
	LSGAT-GCNII	79.2	79.3	79.4	79.6	79.8	80.4	$80.4 \pm 0.5(64)$
	Improvement	$+1.0$	$+0.5$	$+0.1$	-0.6	$+0.0$	$+0.7$	$+0.2$

5.2 Comparison with GAT-based Algorithms.

We evaluate the performance of the proposed LSGAT and existing deep GATs methods *w.r.t.* the increasing number of layers for node classification on real-world datasets. The results are shown

Table 3: Test accuracies (%) of LSGAT based on different β

Dataset	Method	Layers 5	10	20	30
Cora	LSGAT 0.2	79.1 ± 1.3	77.5 ± 0.6	75.8 ± 1.8	73.5 ± 0.8
	LSGAT 0.4	78.9 ± 1.0	77.2 ± 0.5	75.5 ± 0.4	69.2 ± 1.6
	LSGAT 0.6	78.7 ± 1.0	77.2 ± 1.6	76.2 ± 0.8	71.2 ± 3.0
	LSGAT 0.8	78.6 ± 0.7	77.2 ± 0.7	75.9 ± 0.6	71.4 ± 2.1
Pubmed	LSGAT 0.2	76.1 ± 0.7	76.3 ± 0.5	76.9 ± 0.9	72.8 ± 1.4
	LSGAT 0.4	76.9 ± 1.1	75.9 ± 1.5	77.4 ± 0.7	73.2 ± 8.0
	LSGAT 0.6	76.7 ± 0.8	76.6 ± 1.1	77.0 ± 1.3	72.1 ± 2.4
	LSGAT 0.8	77.0 ± 0.8	76.9 ± 1.2	76.7 ± 1.0	71.3 ± 3.4



(a) Citeseer (b) Cora (c) Pubmed (d) CoauthorCS

Figure 4: The comparison of test accuracies between LSGAT and other general Deep Graph Neural Network methods, which are equipped with Graph Attention Networks (GAT).

in Table 1, where the "Variation" denotes the accuracy gap between models with 2 and 30 layers. Please note that due to space limit, the standard deviations are reported in Appendix A.3. From Table 1, it can be observed from variation that LSGAT has remarkably alleviated the over-smoothing issue happened in GAT. The accuracy of LSGAT is greatly higher than other methods as depth equals to 30 in all datasets. Particularly, on CoauthorPhysics dataset, the accuracy of our method LSGAT is 87.0% which is much better than the previous best performance by GAT-Lip (63.9%). Furthermore, LSGAT performs the best through almost all numbers of layers, specifically in Citeseer, Pubmed, CoauthorPhysics, and Ogbn-Arxiv datasets. In conclusion, considering both the stability and accuracy through all layers, LSGAT significantly outperforms the baseline methods, especially when the number of layers is large enough. What is more, as a new version of GAT, we can see that the performance gap between LSGAT and GAT has dramatically increased as the number of layers increases.

5.3 Comparison with other Deep GNN Algorithms

In this subsection, we analysis the comparison between the state-of-the-art algorithms designed for deep GNNs and our LSGAT based on GAT to further verify the performance. As shown in Figure 4, LSGAT consistently exhibits superior performance. Especially the results with mainly compared layers: fifteen layers and thirty layers, among sixty-four cases based on eight algorithms, our LSGAT achieves the best performance in sixty-three cases. The comparison implies that with the proper consideration of overwhelming propagation, GAT itself could be effective both in shallow and deep layers. Furthermore, the comparison between LSGAT and other deep GNNs methods, which tried to relieve the oversmoothing problem based on GCN and ChebyNet has been put in Appendix A.1.

5.4 Combining with Other Deep GNN Methods

Combining with GCNII. As one of the state-of-the-art deep GCN variants, GCNII suggested to have a try of new version of GCNII, which includes attention mechanism. In this section, we further investigate the performance of LSGAT with initial residual and identity mapping introduced in GCNII, named as LSGAT-GCNII. The results are demonstrated in Table 2, where parentheses include the number of layers of the model that achieves the best performance. As shown in Table 2, LSGAT-GCNII further enhances the performance with new state-of-the-art results on several datasets. As highlighted in Section 3 and mentioned in GCNII, the nodes with large degrees are more likely to suffer from oversmoothing problem. With redesigned self-attention mechanism, the component $\hat{\boldsymbol{A}}_{\odot}\boldsymbol{H}^{(l)}$ in GCNII is replaced by $\boldsymbol{C}^{*}\boldsymbol{H}^{(l)}$ in LSGAT-GCNII, which is much more sensitive to the nodes with high overlap-degrees during aggregation, and better relieve the oversmoothing problem. LSGAT-GCNII fills the gap and improves the performance of GCNII based on the theory that nodes with high degrees are more likely to lead to the oversmoothing problem, which is not addressed in GCNII. Therefore, through all datasets, LSGAT-GCNII generally outperforms GCNII. Specifically, the improvement brought by LSGAT can be up to 3.4% on Citeseer. Notably, with no more than sixteen layers, the improvement of LSGAT-GCNII is significant. As number of layers increases, the improvement generally shrinks, whose reason is in the computation process of GCNII. As introduced in Section 5.1, because of the initial residual connection, the fraction of information from l-th

layer, *i.e.,* $\boldsymbol{C}^*\boldsymbol{H}^{(l)}$ in LSGAT-GCNII, dramatically reduces when l increases. The performance improvement brought by LSGAT reduces accordingly.

5.5 Parameter Study

In this subsection, we take a deeper look at the proposed LSGAT to find whether there exists the best proportion (β in Section 4) between large degree nodes and small degree nodes in each setting. As shown in Table 3, the performances based on different proportion are all good enough compared with GAT ($\beta = 0.0$). This further verifies that the performance of deep GAT could be improved by constraining the nodes with a large degree in deeper layers. Because of the limited space, detailed results have been put in Appendix A.2.

6 Related Work

In this section, we introduce the related works. Specifically, the deep graph neural network methods and graph attention network-based works are introduced.

Deep Graph Neural Networks. There has been an important line of research works that aim to relieve the over-smoothing issue. Inspired by ResNets (He et al., 2016a), the methods that are based on skip-connection are proposed in Li et al. (2019); Zhang et al. (2020); Luan et al. (2019); Xu et al. (2018) to exploit node representations from the preceding layers. Specifically, Chen et al. (2020) improves the capacity of APPNP (Klicpera et al., 2018) by using initial residual and identity mapping in each layer. Another line is using normalization to re-scale node representations to constrain pairwise node distance (Ioffe & Szegedy, 2015; Zhao & Akoglu, 2019; Zhou et al., 2020b). In particularly, Zhou et al. (2020a) normalises representations for nodes within the same group separately, and isolates node distributions among distinct groups to prevent over-smoothing. Random dropout methods (Huang et al., 2021; 2020) are connectivity-aware and graph-adaptive sampling approaches, which could address over-smoothing and over-fitting problems. Recently, Jin et al. (2022) try to relieve the performance degradation problem in deep GNNs from a new perspective named overcorrelation. A series of works (Loukas, 2019; Zeng et al., 2020; Li et al., 2020; Cong et al., 2021; Huang et al., 2020) have explored the underlying reasons for performance degradation towards mitigation solutions.

Graph Attention Networks. Various research works focus on designing the attention mechanism on graph neural networks for specific tasks and applications. In a synthetic issue requiring dynamic node selection, Brody et al. (2021) developed a dynamic graph attention alternative which is strictly more expressive than GAT. In recommender systems, Wu et al. (2019) developed dual graph attention networks to cooperatively learn representations for two-fold social impacts. Park et al. (2020) proposed a new spatio-temporal graph attention paradigm with spatial attention and temporal attention for capturing the spatio-temporal dynamics in road networks. Wang et al. (2020) developed multi-hop attention graph neural network, which calculated the attention between the given node and its multi-hop neighbors. Dasoulas et al. (2021) tried to relieve the gradient explosion problem in deep GAT by introducing the Lipschitz normalization.

7 Conclusion

In this paper, we investigated that oversmoothing problem happened in deep GAT could be relieved by considering overwhelming propagation caused by the nodes with large degree. Then, we propose a novel and versatile coefficient computation mechanism LSGAT to properly train GAT. This mechanism could rescale the propagation influence based on overlap-degree from adjacent and non-adjacent nodes adaptively with the number of layers, and specifically limit the propagation of nodes with large degrees to relieve the oversmoothing problem in deep GAT. Specifically, LSGAT does not change the architecture of GAT, and our layer-wise scaling scores could be calculated offline and easily applied to GAT in the training phase. The results of extensive experiments on various real-world datasets show the advantage of our proposed method over the baselines. Specifically, with initial residual and identity mapping, our proposed LSGAT achieves the state-of-the-art performance as a deep GNN.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

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

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

Tianlong Chen, Kaixiong Zhou, Keyu Duan, Wenqing Zheng, Peihao Wang, Xia Hu, and Zhangyang Wang. Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Weilin Cong, Morteza Ramezani, and Mehrdad Mahdavi. On provable benefits of depth in training graph convolutional networks. *Advances in Neural Information Processing Systems*, 34, 2021.

George Dasoulas, Kevin Scaman, and Aladin Virmaux. Lipschitz normalization for self-attention layers with application to graph neural networks. In *International Conference on Machine Learning*, pp. 2456–2466. PMLR, 2021.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

Xinzhou Dong, Beihong Jin, Wei Zhuo, Beibei Li, and Taofeng Xue. Improving sequential recommendation with attribute-augmented graph neural networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 373–385. Springer, 2021.

Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1416–1424, 2018.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016b.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Wei Huang, Yayong Li, Weitao Du, Richard Yi Da Xu, Jie Yin, Ling Chen, and Miao Zhang. Towards deepening graph neural networks: A gntk-based optimization perspective. *arXiv preprint arXiv:2103.03113*, 2021.

Wenbing Huang, Yu Rong, Tingyang Xu, Fuchun Sun, and Junzhou Huang. Tackling over-smoothing for general graph convolutional networks. *arXiv preprint arXiv:2008.09864*, 2020.

Xiao Huang, Qingquan Song, Yuening Li, and Xia Hu. Graph recurrent networks with attributed random walks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 732–740, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Wei Jin, Xiaorui Liu, Yao Ma, Charu Aggarwal, and Jiliang Tang. Feature overcorrelation in deep graph neural networks: A new perspective. *arXiv preprint arXiv:2206.07743*, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018.

Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9267–9276, 2019.

Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. Deepergcn: All you need to train deeper gcns. *arXiv preprint arXiv:2006.07739*, 2020.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*, 2018.

Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 338–348, 2020.

Andreas Loukas. What graph neural networks cannot learn: depth vs width. *arXiv preprint arXiv:1907.03199*, 2019.

Sitao Luan, Mingde Zhao, Xiao-Wen Chang, and Doina Precup. Break the ceiling: Stronger multi-scale deep graph convolutional networks. *Advances in neural information processing systems*, 32, 2019.

Sitao Luan, Mingde Zhao, Xiao-Wen Chang, and Doina Precup. Training matters: Unlocking potentials of deeper graph convolutional neural networks. *arXiv preprint arXiv:2008.08838*, 2020.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2019.

Cheonbok Park, Chunggi Lee, Hyojin Bahng, Yunwon Tae, Seungmin Jin, Kihwan Kim, Sungahn Ko, and Jaegul Choo. St-grat: A novel spatio-temporal graph attention networks for accurately forecasting dynamically changing road speed. In *Proceedings of the 29th ACM International conference on information & knowledge management*, pp. 1215–1224, 2020.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *Relational Representation Learning Workshop, NeurIPS 2018*, 2018.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec. Multi-hop attention graph neural network. *arXiv preprint arXiv:2009.14332*, 2020.

Hanchen Wang, Defu Lian, Ying Zhang, Lu Qin, and Xuemin Lin. Gognn: graph of graphs neural network for predicting structured entity interactions. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 1317–1323, 2021.

Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In *The World Wide Web Conference*, pp. 2091–2102, 2019.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pp. 5453–5462. PMLR, 2018.

Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. *arXiv preprint arXiv:2102.06462*, 2021.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33: 5812–5823, 2020.

Hanqing Zeng, Muhan Zhang, Yinglong Xia, Ajitesh Srivastava, Andrey Malevich, Rajgopal Kannan, Viktor Prasanna, Long Jin, and Ren Chen. Deep graph neural networks with shallow subgraph samplers. *arXiv preprint arXiv:2012.01380*, 2020.

Hongwei Zhang, Tijin Yan, Zenjun Xie, Yuanqing Xia, and Yuan Zhang. Revisiting graph convolutional network on semi-supervised node classification from an optimization perspective. *arXiv preprint arXiv:2009.11469*, 2020.

Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in gnns. *arXiv preprint arXiv:1909.12223*, 2019.

Kaixiong Zhou, Xiao Huang, Yuening Li, Daochen Zha, Rui Chen, and Xia Hu. Towards deeper graph neural networks with differentiable group normalization. *Advances in neural information processing systems*, 33:4917–4928, 2020a.

Kuangqi Zhou, Yanfei Dong, Kaixin Wang, Wee Sun Lee, Bryan Hooi, Huan Xu, and Jiashi Feng. Understanding and resolving performance degradation in graph convolutional networks. *arXiv preprint arXiv:2006.07107*, 2020b.