Two Sides of the Same Coin: Heterophily and Oversmoothing in Graph Convolutional Neural Networks

Yujun Yan[1], Milad Hashemi[2], Kevin Swersky[2], Yaoqing Yang[3], Danai Koutra[1]
[1]University of Michigan, [2]Google Research, [3]University of California, Berkeley

***Abstract*—In node classification tasks, graph convolutional neural networks (GCNs) have demonstrated competitive performance over traditional methods on diverse graph data. However, it is known that the performance of GCNs degrades with increasing number of layers (oversmoothing problem) and recent studies have also shown that GCNs may perform worse in heterophilous graphs, where neighboring nodes tend to belong to different classes (heterophily problem). These two problems are usually viewed as unrelated, and thus are studied independently, often at the graph filter level from a spectral perspective.**

We are the first to take a unified perspective to jointly explain the oversmoothing and heterophily problems at the node level. Specifically, we profile the nodes via two quantitative metrics: the relative degree of a node (compared to its neighbors) and the node-level heterophily. Our theory shows that the interplay of these two profiling metrics defines three cases of node behaviors, which explain the oversmoothing and heterophily problems jointly and can predict the performance of GCNs. Based on insights from our theory, we show theoretically and empirically the effectiveness of two strategies: structure-based edge correction, which learns corrected edge weights from structural properties (i.e., degrees), and feature-based edge correction, which learns signed edge weights from node features. Compared to other approaches, which tend to handle well either heterophily or oversmoothing, we show that our model, GGCN, which incorporates the two strategies performs well in both problems. We provide a longer version of this paper in [1] and codes on https://github.com/Yujun-Yan/Heterophily_and_oversmoothing.

I. Introduction

GCNs are effective and widely used in various applications [2, 3, 4], but their performance may degrade in some cases. Li et al. [5] pointed out the "oversmoothing problem": GCNs perform worse with increasing number of layers. It is claimed that oversmoothing could be caused by GCNs exponentially losing expressive power in the node classification task [6] and that the node representations converge to a stationary state decided by the node degrees and input features [7, 8]. These works analyze the asymptotic node representations in the limit of infinite layers, but they do not characterize how the node representations change over the layers (we call different types of changes *node behaviors*) and how different node behaviors contribute to oversmoothing. Going beyond empirical measures of oversmoothing [9], we propose theoretically-grounded node-level metrics that characterize different node behaviors across GCN layers, and show theoretically and empirically how they can explain oversmoothing and identify the nodes triggering it.

GCNs may also perform poorly on heterophilous graphs [10], which—unlike homophilous graphs—comprise many neighboring nodes that belong to different classes [11]. This is termed as the "heterophily problem". For instance, in protein networks, amino acids of different types tend to form links [12], and in transaction networks, fraudsters are more likely to connect to accomplices than to other fraudsters [13]. Most GCNs [2, 3] fail to effectively capture heterophily, so various designs have been proposed to handle it [10, 12, 14, 15]. These works take the spectral perspective and design various high-frequency graph filters to address heterophily. However, they neglect the fact that different node behaviors impact GCNs' performance under heterophily differently and need to be handled separately. In this work, we show that GCNs can perform differently on graphs that have similar graph-level heterophily but are dominated by different node behaviors.

This work. These two problems, which cause performance degradation, have mostly been studied independently. Recent work on oversmoothing [7] was shown only empirically to address heterophily, and vice versa [14]. Motivated by this empirical observation, we are the first to find a joint explanation for the two problems. Specifically, we aim to identify meaningful node-level metrics that are theoretically-grounded and their interplay can be used to characterize different node behaviors (profiles), which in turn can explain both problems. We found that the relative degree of a node (compared to its neighbors) and its node-level heterophily define three types of node behaviors, two of which are related to performance degeneration. Based on our theoretical insights, we show theoretically and empirically the effectiveness of two strategies: structure-based edge correction, which learns corrected edge weights from structural properties like node degree, and feature-based edge correction, which learns signed edge weights from node features. Signed edge weights can model both positive and negative influence from the neighbors.

In sum, we make the following contributions:

- **Theoretically-grounded Node Metrics:** We introduce two theoretically-grounded metrics, relative degree and node-level heterophily, to profile the nodes across layers in GCNs. The profiling provides a joint explanation for what triggers the heterophily and oversmoothing problems.
- **Insights:** Our theory states that under certain conditions, low-degree nodes tend to trigger the oversmoothing problem in strongly homophilous graphs, while high-degree nodes tend to cause the oversmoothing and heterophily problems in weakly homophilous (i.e., heterophilous) graphs. We also show that leveraging signed edge weights can help alleviate both problems.
- **Improved Model & Empirical Analysis:** Based on our insights, we show theoretically and empirically the effec-

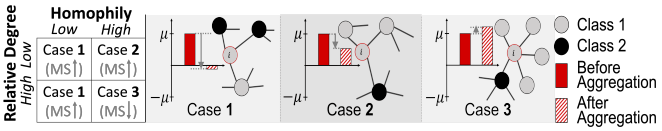

Fig. 1: Node representation dynamics during neighborhood aggregation (in 1D for illustration purposes; 'MR': misclassification rate). The expectation of node representations from class 1 & 2 are denoted by μ and $-\mu$, respectively. The bars show the expected node representations of node v_i before and after the aggregation.

tiveness of two strategies, structure-based edge correction and feature-based edge correction. Our empirical results show that our model, GGCN, which leverages the two strategies is robust to oversmoothing, achieves state-of-the-art performance on datasets with high levels of heterophily, and achieves competitive performance on homophilous datasets.

II. Preliminaries

We first provide the major notations and definitions that we use in the paper, and a brief background on GCNs.

Notation. We denote an unweighted and self-loop-free graph as $\mathcal{G}$ ($\mathcal{V}$, $\mathcal{E}$) and its adjacency matrix as $\mathbf{A}$. We represent the degree of node $v_i \in \mathcal{V}$ by d_i, and the degree matrix—which is a diagonal matrix whose elements are node degrees—by $\mathbf{D}$. Let $\mathcal{N}_i$ be the set of nodes directly connected to v_i, i.e., its neighbors. $\mathbf{I}$ is the identity matrix. We denote the node representations at l-th layer as $\mathbf{F}^{(l)}$, and the i-th row of $\mathbf{F}$ is $\mathbf{f}_i^{(l)}$, which is the representation of node v_i. The input node features are given by $\mathbf{F}^{(0)}$. The weight matrix and bias vector at the l-th layer are denoted as $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$, respectively.

Supervised Node Classification Task. We focus on node classification: Given a random sample of node representations $\{\mathbf{f}_1^{(0)} \dots \mathbf{f}_n^{(0)}\} \in \mathbb{R}^m$ and their labels $\{y_1 \dots y_n\} \in \mathbb{R}^n$ for training, we aim to learn a function $\mathscr{F} : \mathbb{R}^m \mapsto \mathbb{R}^n$, such that the loss $\mathbb{E}(\mathscr{L}(y_i, \hat{y}_i))$ is minimized, where $\hat{y}_i = \mathscr{F}(\mathbf{f}_i^{(0)})$ is the predicted label of v_i. The **misclassification rate** is defined as the probability $\mathbb{P}(y_i \neq \hat{y}_i)$ to misclassify an arbitrary node in the representation space.

GCNs. In node classification tasks, an L-layer GCN contains two components [16]: (1) neighborhood propagation and aggregation: $\widehat{\mathbf{f}_i^{(l)}}$ = $\texttt{AGGREGATE}(\mathbf{f}_j^{(l)}, v_j \in \mathcal{N}_i)$, and (2) combination: $\mathbf{f}_i^{(l+1)}$ = $\texttt{COMBINE}(\widehat{\mathbf{f}_i^{(l)}}, \mathbf{f}_i^{(l)})$, where $\texttt{AGGREGATE}$ and $\texttt{COMBINE}$ are learnable functions. The loss is given by $\mathscr{L}_{CE}$=$\texttt{CrossEntropy}(\texttt{Softmax}(\mathbf{f}_i^{(L)}\mathbf{W}^{(L)} + \mathbf{b}^{(L)}), y_i)$. The vanilla GCN suggests a renormalization trick on the adjacency $\mathbf{A}$ to prevent gradient explosion [2]. The $(l+1)$-th output is given by: $\mathbf{F}^{(l+1)} = \sigma(\tilde{\mathbf{A}}\mathbf{F}^{(l)}\mathbf{W}^{(l)})$, where $\tilde{\mathbf{A}} = \tilde{\mathbf{D}}^{-1/2}(\mathbf{I}+\mathbf{A})\tilde{\mathbf{D}}^{-1/2}$, $\tilde{\mathbf{D}}$ is the degree matrix of $\mathbf{I} + \mathbf{A}$, and σ is ReLU. When the non-linearities in the vanilla GCN are removed, it reduces to a linear model called SGC [17], which has competitive performance and is widely used in theoretical analyses [6, 7]. For SGC, the l-th layer representations are given by: $\mathbf{F}^{(l)} = \tilde{\mathbf{A}}^l\mathbf{F}^{(0)}$ and the last layer is a logistic-regression layer: $\hat{y}_i = \texttt{Softmax}(\mathbf{F}^{(L)}\mathbf{W}^{(L)} + \mathbf{b}^{(L)})$. We note that only one weight matrix $\mathbf{W}^{(L)}$ is learned, which is equivalent to the products of all weight matrices in a linear GCN. More related works can be found in § VI.

III. Theoretical Analysis

We now formally introduce two metrics: node-level homophily h_i and relative degree $\overline{r_i}$. We show theoretically how these metrics (1) characterize different node behaviors across GCN layers, and (2) can be used for node profiling to explain the oversmoothing and heterophily problems.

We first introduce the theoretical setup. Throughout the section, we analyze binary node classification using the typically-studied SGC model (§ II). The nodes in class 1 and class 2 are denoted as sets $\mathcal{V}_1$ and $\mathcal{V}_2$, respectively. Later, in § V, we show empirically that the insights obtained in this section are also effective for other non-linear models in multi-class classification. We give all the proofs in the long version [1].

A. Assumptions

Below, we use "i.d." to represent random variables / vectors that follow the same marginal distribution and their joint probability density function (PDF) is a permutation-invariant function $p(\mathbf{x}_1, \dots, \mathbf{x}_n) = p(\mathbf{P}(\mathbf{x}_1, \dots, \mathbf{x}_n))$, where $\mathbf{P}(\cdot)$ means permutation. We use $\mathbb{E}_{A|B}(\cdot)$ to denote the expectation taken over the randomness of A given B.

We make the following assumptions:

(A1) Random Graph: Node degrees $\{d_i\}$ are i.d. random variables, where $\{(\cdot)_i\}$ represents a set with $i = 1, \dots, |\mathcal{V}|$.

(A2) Inputs: *(A2.1)* Node labels $\{y_i\}$ are i.d. Bernoulli random variables given by the ratio ρ: $\rho \equiv \frac{\mathbb{P}(y_i=1)}{\mathbb{P}(y_i=2)}, \forall i$. The event $\{y_i = y_j | v_j \in \mathcal{N}_i\}$ is independent of y_i, $\forall i, j$.

(A2.2) Initial input node features $\{\mathbf{f}_i^{(0)}\}$ are random vectors given by (PDF) $f(\mathbf{x})$, which is expressed as:

$$f(\mathbf{x}) = \begin{cases} f_1(\mathbf{x}), \text{when } y_i = 1. \\ f_2(\mathbf{x}), \text{when } y_i = 2. \end{cases}$$

$$\mathbb{E}(\mathbf{f}_i^{(0)}|y_i) = \begin{cases} \boldsymbol{\mu}, & \text{when } y_i = 1 \\ -\rho\boldsymbol{\mu}, & \text{when } y_i = 2 \end{cases}, \text{ so } \mathbb{E}(\mathbf{f}_i^{(0)})=\mathbf{0}.$$

(A3) Independence: $\mathbf{A}$ is independent of $\{y_i\}$ and $\{\mathbf{f}_i^{(0)}\}$. Also, given y_i, variables $\mathbf{f}_i^{(0)}$ and y_j are conditional independent for all i, j.

B. Node-level Metrics: Definitions

In our theoretical analysis, we consider two node-level properties: homophily or heterophily, and relative degree.

Given a set of node labels/classes, **node-level homophily** captures the tendency of a node to have the same class as its neighbors. Formally, the homophily of node v_i is defined as:

$$h_i \equiv \mathbb{P}(y_i = y_j | v_j \in \mathcal{N}_i).$$

High homophily corresponds to low **heterophily**, and vice versa, so we use these terms interchangeably.

The **relative degree** of node v_i evaluates its node degree compared to its neighbors' degrees and it is defined as:

$$\overline{r_i} \equiv \mathbb{E}_{\mathbf{A}|d_i}(\frac{1}{d_i}\sum_{j\in\mathcal{N}_i} r_{ij}|d_i), \text{where } r_{ij} \equiv \sqrt{\frac{d_i+1}{d_j+1}}.$$

When all the nodes have the same degree, $\overline{r_i} = 1$.

C. Node Profiling: Theory

In this section, we theoretically show how the two metrics can characterize different node behaviors across layers.

1) Movements of Node Representations: We monitor the node behaviors by tracking the changes of node representations across the layers. Each node representation is mapped to a point in the feature space whose coordinates are decided by the representation vector. In this way, the changes of a node's representations across the layers can be viewed as the movements of the mapped point. For example, $\mathbf{f}_i^{(l+1)} - \mathbf{f}_i^{(l)}$ is referred to as the movement of node v_i's representation at the l-th layer. Next, we show that the interplay of the two metrics relates to different types of movements.

2) Movements at the initial layer: We study how the node representations change in expectation. We assume $v_i \in \mathcal{V}_1$ and the other case can be derived similarly.

Theorem III.1. *Given $v_i \in \mathcal{V}_1$ and d_i, the conditional expectation of representation $\mathbf{f}_i^{(1)}$ is given by:*

$$\mathbb{E}_{\mathbf{A},\{y_i\},\{\mathbf{f}_i^{(0)}\}|d_i,v_i\in\mathcal{V}_1}(\mathbf{f}_i^{(1)}|v_i \in \mathcal{V}_1, d_i) \quad (1)$$
$$= \left(\frac{((1+\rho)h_i - \rho)d_i\overline{r_i}+1}{d_i+1}\right)\mathbb{E}(\mathbf{f}_i^{(0)}|v_i\in\mathcal{V}_1) \equiv \gamma_i^1\mathbb{E}(\mathbf{f}_i^{(0)}|v_i\in\mathcal{V}_1),$$

where $\gamma_i^1 \in \begin{cases}(-\infty, \frac{1}{2}], & \text{if } h_i \le \frac{\rho}{1+\rho} \quad \texttt{(case 1)}\\ (0,1], & \text{if } h_i > \frac{\rho}{1+\rho} \ \&\ \overline{r_i} \le \frac{1}{(1+\rho)h_i-\rho} \quad \texttt{(case 2)}\\ (1,\infty), & \text{otherwise} \quad \texttt{(case 3)}\end{cases}$

is a multiplicative factor. In case 1, γ_i^1 decreases as d_i increases; in case 3, γ_i^1 increases as d_i increases.

From Thm. III.1, we identify three types of movements of node representations, which are characterized by relative degree $\overline{r_i}$ and homophily level h_i. For illustration purposes, in Fig. 1, we show the three cases when we apply our theorem to 1-dimensional node representations. The bars reflect the value change of v_i's node representation. Intuitively, under heterophily (case 1), node representations tend to move closer to the representations of the other class. The higher the node degree, the more the representation moves. Under high homophily but low degrees (case 2), node representations still tend to move towards the other class, but not as much as in case 1. Only when both the homophily and the degree is high, the node representations may move away from the other class. Thus, case 3 is the only favorable case.

3) Movements at deeper layers: The scenarios at deeper layers are more complex. However, by extending the two node-level metrics (i.e., homophily and relative degree compared to neighbors) to *effective* homophily and *effective* relative degree, respectively, we can obtain a similar equation, and the extended metrics can characterize the nodes into 3 cases similar to Thm. III.1. More details of the extension and proofs can be found in the long version [1].

4) Movements & Misclassification Rate: In the long version [1], we also prove that under certain condition, the movement of representations towards the other class by a non-zero step increases the misclassification rate, causing performance degradation. We note that the condition is important to explain why recent works [18, 19] find that GCNs can sometimes perform well in heterophilous graphs (e.g., bipartite graphs) because the representations of opposite classes swap places.

5) Explanation for Heterophily and Oversmoothing: Oversmoothing problem: Nodes with low homophily (case 1) and nodes with high homophily but low degrees (case 2) cannot benefit from message aggregation. Their representations tend to move towards the other class. Under certain conditions, their misclassification rate is increased via message aggregation. GCNs' performance on node classification degrades each time the message aggregation is applied, which explains oversmoothing in homophilous and heterophilous graphs.

Heterophily problem: In heterophilous graphs, nodes from case 1 (and sometimes case 2) dominate. The performance degradation occurs at the first layer, which explains why GCNs may perform worse than MLP in heterophilous graphs.

Relation between the problems: (1) In heterophilous graphs, both problems are caused by nodes from case 1 and case 2. Message aggregation makes the representations of these nodes (esp. case 1) less distinguishable. (2) In homophilous graphs, we can decompose the oversmoothing process into two stages, where the node behaviors in the second stage resemble those in the heterophily problem. *Initial Stage.* At shallow layers, nodes of case 3 dominate initially, GCNs benefit from graph convolution. *Developing Stage.* Nodes of case 2 and case 1 cannot benefit from message aggregation and their misclassification rate increases with more layers. In deep layers, they are misclassified and are wrongly viewed by their neighbors as coming from a different class. Thus, nodes of case 2 and case 1 cause low effective homophily of their neighbors. At last, most nodes have low effective homophily in deep layers and are transformed to case 1, which resembles the phenomenon in heterophilous graphs ("pseudo-heterophily").

D. Node Profiling With Signed Edges

In this section, we discuss how the interplay of the two metrics changes when allowing signed edges. We provide theory to show when signed edges can help enhance the performance in heterophilous graphs and alleviate oversmoothing. Due to limited space, we only show the effect of signed edges at the initial layer; similar results can be derived in deeper layers.

Setup. Each edge is assigned a positive or a negative sign. Messages passing through a signed edge are multiplied by its sign. Ideally, we would like to assign the positive signs to homophilous edges (i.e, edges connecting nodes from the same class) and negative signs to heterophilous edges. In reality, we cannot access the nodes' ground-truth labels and cannot know whether the edges are homophilous or heterophilous. Thus we learn the signs, which introduces errors. For node v_i, we define m_i^l as the *ratio of neighbors that send incorrect messages* at the l-th layer because we wrongly assign a negative (positive) sign to a homophilous (heterophilous) edge that connects them. We define the l-th layer *error rate* as $e_i^l = \mathbb{E}(m_i^l)$, where the expectation is over the randomness of the neighbors that send incorrect messages. We assume that m_i^l is independent of $\{d_i\}$, $\{y_i\}$ and $\{\mathbf{f}_i^0\}$.

Theorem III.2. ***[Signed Edges]*** *By allowing signed edges, the movements of representations will* ***be less affected*** *by the initial*

homophily level h_i, and will be dependent on the ***error rate*** *e_i^0. The multiplicative factor γ_i^1 at the first layer is:*

$$\mathbb{E}_{\mathbf{A},\{y_i\},\{\mathbf{f}_i^{(0)}\}|d_i,v_i\in\mathcal{V}_1}(\mathbf{f}_i^{(1)}|d_i,v_i\in\mathcal{V}_1) \tag{2}$$
$$=\left(\frac{(1-2e_i^0)(\rho+(1-\rho)h_i)d_i\overline{r_i}+1}{d_i+1}\right)\mathbb{E}(\mathbf{f}_i^{(0)}|v_i\in\mathcal{V}_1),$$

where

$$\gamma_i^1\in\begin{cases}(-\infty,\frac{1}{2}], & \text{if } e_i^0\geq 0.5\\ (0,1], & \text{if } e_i^0<0.5 \;\&\; \overline{r_i}\leq\frac{1}{(1-2e_i^0)(\rho+(1-\rho)h_i)}\\ (1,\infty), & \text{otherwise.}\end{cases} \tag{3}$$

From Eq. 3, we see that when using signed edges, to benefit from case 3 ($\gamma_i^1>1$), the minimum relative degree satisfies: $\overline{r_i}>\frac{1}{(1-2e_i^0)(\rho+(1-\rho)h_i)}$. Given $h_i\leq 1$, if the error rate is low ($e_i^0\ll 0.5$), we get: $\frac{1}{(1-2e_i^0)(\rho+(1-\rho)h_i)}\leq\frac{1}{(1+\rho)h_i-\rho}$, and $\frac{1}{(1+\rho)h_i-\rho}$ is the minimum relative degree required when not using signed edges. This implies that more nodes can benefit from using signed edges. We note that if low error rate cannot be guaranteed, signed edges may hurt the performance.

IV. Model Design

Based on our theoretical analysis, we propose two new, simple mechanisms to address both the heterophily and oversmoothing problems: structure-based edge correction and feature-based edge correction. We integrate these mechanisms, along with a decaying combination of the current and previous node representations [7], into a generalized GCN model, GGCN, whose effectiveness we show empirically in § V.

A. Structure-based Edge Correction

Our analysis in § III-C and § III-D highlights that, when the homophily level is high (or error rate is low), oversmoothing is initially triggered by low-degree nodes. Thus, we aim to compensate for low degrees by learning new edge weights. Unlike attention which encodes similarity of features, these weights only contain structural information (i.e., node degrees).

Based on Eq. (3), we require that the node degrees satisfy $\overline{r_i}>\frac{1}{(1-2e_i^0)(\rho+(1-\rho)h_i)}$ to prevent oversmoothing. Since the node degrees cannot be modified, our strategy is to *rescale* or correct the edge weights by multiplying them with scalars τ_{ij}^l:

$$(\tilde{A}\mathbf{F}^{(l)})[i,:]=\frac{\mathbf{F}^{(l)}[i,:]}{d_i+1}+\boxed{\sum_{v_j\in\mathcal{N}_i}\frac{\mathbf{F}^{(l)}[j,:]}{\sqrt{d_i+1}\sqrt{d_j+1}}}$$
$$\implies\sum_{v_j\in\mathcal{N}_i}\frac{\tau_{i,j}^l\,\mathbf{F}^{(l)}[j,:]}{\sqrt{d_i+1}\sqrt{d_j+1}}. \tag{4}$$

This multiplication is equivalent to changing the ratio r_{ij} in Thm. III.1 to $\sqrt{\frac{(\tau_{ij}^l)^2(d_i+1)}{d_j+1}}$. That is, a larger τ_{ij}^l increases the effective $\overline{r_i}$ at layer l. Training independent $\tau_{i,j}^l$ is not practical because it would require $O(|\mathcal{V}|^2)$ additional parameters per layer, which can lead to overfitting. Moreover, low-rank parameterizations suffer from unstable training dynamics. Intuitively, when r_{ij} is small, we would like to compensate for it via a larger $\tau_{i,j}^l$. Thus, we set $\tau_{i,j}^l$ to be a function of r_{ij}:

$$\tau_{ij}^l=\text{softplus}\left(\lambda_0^l\left(\frac{1}{r_{ij}}-1\right)+\lambda_1^l\right), \tag{5}$$

where λ_0^l and λ_1^l are learnable parameters. We subtract 1 so that when $r_{ij}=1$ (i.e., $d_i=d_j$), then $\tau_{ij}^l=\text{softplus}(\lambda_1^l)$ is a constant bias.

Let $\boldsymbol{\mathcal{T}}^{(l)}$ be a matrix with elements τ_{ij}^l. Our model GGCN learns a corrected adjacency matrix at l-th layer: $\widehat{\tilde{\mathbf{A}}^l}=\tilde{\mathbf{A}}\odot\boldsymbol{\mathcal{T}}^{(l)}$, where $\odot$ is element-wise multiplication.

B. Feature-based Edge Correction

Theorem III.2 points out the importance of signed edges in tackling the heterophily and oversmoothing problems. Inspired by this, we aim to learn the signed edge weights based on node features. Unlike attention weights that are usually nonnegative, we allow the edge weights to be negative.

For expressiveness, as in GCN [2], we first perform a learnable linear transformation of each node's representation at the l-th layer: $\widehat{\mathbf{F}^{(l)}}=\mathbf{F}^{(l)}\mathbf{W}^{(l)}+\mathbf{b}^{(l)}$. Then, we define a sign function to be multiplied with the messages exchanged between neighbors. To allow for backpropagation of the gradient information, we approximate the sign function with cosine similarity. Denote $\mathbf{S}^l$ as the matrix which stores the sign information about the edges, defined as: $\mathbf{S}^{(l)}[i,j]=\texttt{Cosine}(\mathbf{f}_i^{(l)},\mathbf{f}_j^{(l)})$ if $(i\neq j)$ & $(v_j\in\mathcal{N}_i)$; 0 otherwise.

In order to separate the contribution of similar neighbors (likely in the same class) from that of dissimilar neighbors (unlikely to be in the same class), we split $\mathbf{S}^{(l)}$ into a positive matrix $\mathbf{S}_{\text{pos}}^{(l)}$ and a negative matrix $\mathbf{S}_{\text{neg}}^{(l)}$. Thus, our proposed GGCN model learns a weighted combination of the self-representations, the positive, and the negative messages:

$$\mathbf{F}^{(l+1)}=\sigma\Big(\hat{\alpha}^l(\hat{\beta}_0^l\,\widehat{\mathbf{F}^{(l)}}+\hat{\beta}_1^l(\mathbf{S}_{\text{pos}}^{(l)}\odot\widehat{\tilde{\mathbf{A}}^l})\widehat{\mathbf{F}^{(l)}}+\hat{\beta}_2^l(\mathbf{S}_{\text{neg}}^{(l)}\odot\widehat{\tilde{\mathbf{A}}^l})\widehat{\mathbf{F}^{(l)}})\Big),$$

where $\hat{\beta}_0^l$, $\hat{\beta}_1^l$ and $\hat{\beta}_2^l$ are scalars obtained by applying softmax to the learned scalars β_0^l, β_1^l and β_2^l; the non-negative scaling factor $\hat{\alpha}^l=\text{softplus}(\alpha^l)$ is derived from the learned scalar α^l; and σ is the nonlinear function Elu. We note that we learn *different* α and β parameters *per layer* for flexibility. We also require the combined weights, $\hat{\alpha}^l\hat{\beta}_x^l$, to be non-negative so that they do not negate the intended effect of the signed information.

C. Decaying Aggregation

Besides our two proposed mechanisms that are theoretically grounded in our analysis (§ III), we also incorporate into GGCN an existing design—decaying aggregation of messages—that empirically increases performance. However, we note that, even without this design, our GCN architecture still performs well under heterophily and is robust to oversmoothing [1].

Decaying aggregation was introduced in [7] as a way to slow down the convergence rate of node representations. Inspired by this work, we modify the decaying function, $\hat{\eta}$, and integrate it to our GGCN model:

$$\mathbf{F}^{(l+1)}=\mathbf{F}^{(l)}+\hat{\eta}\Big(\sigma\Big(\hat{\alpha}^l(\hat{\beta}_0^l\widehat{\mathbf{F}^{(l)}}+\hat{\beta}_1^l(\mathbf{S}_{\text{pos}}^{(l)}\odot\tilde{\mathbf{A}}\odot\boldsymbol{\mathcal{T}}^{(l)})\widehat{\mathbf{F}^{(l)}}$$
$$+\hat{\beta}_2^l(\mathbf{S}_{\text{neg}}^{(l)}\odot\tilde{\mathbf{A}}\odot\boldsymbol{\mathcal{T}}^{(l)})\widehat{\mathbf{F}^{(l)}})\Big)\Big). \tag{6}$$

In practice, we found that the following decaying function works well: $\hat{\eta}\equiv\ln(\frac{\eta}{l^k}+1)$, iff $l\geq l_0$; $\hat{\eta}=1$, otherwise. The hyperparameters k, l_0, η are tuned on the validation set.

V. Experiments

We focus on three questions: (**Q1**) How does GGCN perform on homophilous and heterophilous graphs? (**Q2**) How robust is it against oversmoothing under homophily and heterophily? (**Q3**) How can we verify the correctness of our theorems about oversmoothing on real datasets? We also give an ablation study for our proposed edge correction mechanisms in [1].

A. Experimental Setup

Datasets. We evaluate the performance of our GGCN model and existing GNNs in node classification on various real-world datasets [10]. We provide their summary statistics in Table I, where we compute the homophily level h of a graph as the average of h_i of all nodes $v_i \in \mathcal{V}$. For all benchmarks, we use the feature vectors, class labels, and 10 random splits (48%/32%/20% of nodes for train/validation/test[1]) from [10].

Baselines. For baselines we use **(1)** classic GNN models for node classification: vanilla GCN [2], GAT [3] and GraphSage [20]; **(2)** recent models tackling heterophily: Geom-GCN [10], H2GCN [12], FAGCN [15] and GPRGNN [14]; **(3)** models tackling oversmoothing: PairNorm [21] and GCNII [7] (state-of-the-art); and **(4)** 2-layer MLP (with dropout and Elu non-linearity). We use the original codes provided by the authors for GCN, PairNorm, Geom-GCN, GCNII, H2GCN, and GPRGNN; we use the codes from a well-accepted Github repository[2] for GAT. We report the results of GraphSage and FAGCN from [12, 22], which use the same data and splits. We choose the best variant per dataset and denote them as [model]* for the baselines with multiple variants (Geom-GCN, GCNII, H2GCN). We give the hyperparameters in [1].

Machine. We ran our experiments on Nvidia V100 GPU.

B. (Q1) Performance Under Homophily & Heterophily

Table I provides the test accuracy of different GNNs on the supervised node classification task over datasets with varying graph homophily levels (arranged from low homophily to high homophily). We report the best performance of each model across different layers.

GGCN performs the best in terms of average rank (1.78) across all datasets, which suggests its strong adaptability to graphs of various homophily levels. In particular, it achieves the highest accuracy in 5 out of 6 heterophilous graphs (low h) and improves the accuracy by up to 6%. We note that MLP is a good baseline for heterophilous data and usually outperforms models not tailored to heterophily. Our GGCN model is the only model that always performs better than MLP. For homophilous data (high h), GGCN maintains its competitive performance, which is within 1% of the best model.

C. (Q2) Oversmoothing

To test how robust the models are to oversmoothing, we measure the supervised node classification accuracy for 2 to 64 layers. Table II presents the results for both homophilous and heterophilous datasets. Per model, we also report the layer at which the best performance is achieved (column 'Best').

[1][10] claims that the ratios are 60%/20%/20%, which is different from the actual data splits shared on GitHub.

[2]https://github.com/Diego999/pyGAT

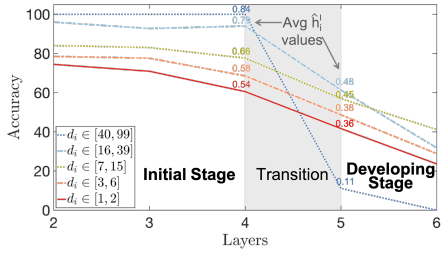

Fig. 2: Accuracy of nodes grouped by degree d_i on Citeseer. Initial stage: when mean effective homophily $\widehat{h}_i^l$ (ratio of a node's neighbors in the same class–§ III-C3) is high, the accuracy increases as the degree increases. Developing stage: when $\widehat{h}_i^l$ is low, the accuracy of high-degree nodes drops more sharply.

As shown in Table II, GGCN and GCNII* achieve **increase** in accuracy when stacking more layers, while GPRGNN and PairNorm exhibit robustness against oversmoothing. Models that are not designed for oversmoothing have various issues. The performance of GCN and Geom-GCN* drops rapidly with more layers; H2GCN* concatenates all the intermediate outputs and quickly reaches memory capacity; GAT needs careful initialization when stacking many layers as it may suffer from numerical instability in sparse tensor operations.

D. (Q3) Empirical Verification of the Two Stages

Using the vanilla GCN model [2], we validate our theorems by measuring the test accuracy and effective homophily for different node degrees (binned logarithmically) on real datasets. We estimate the effective homophily as the portion of the same-class neighbors that are correctly classified *before* the last propagation. Figure 2 shows the results for Citeseer. In the initial stage (high $\widehat{h}_i^l$), the accuracy increases with the degree, but in the developing stage, the trend changes, with high-degree nodes being impacted the most, as predicted by our theorems.

VI. Related Work

Early works [2, 3] propose GCNs based on spectral convolution, followed by variants that target applications in computer vision [23], biology [4], algorithmic tasks [24], and more.

Oversmoothing. There are various empirical solutions for the oversmoothing problem [5] of GCNs: residual connections and dilated convolutions [25]; new normalization strategies [21] and aggregations [7]; and edge dropout [26].

Heterophily & GCNs. Heterophily has recently been recognized as an important issue for GCNs [10]. Zhu et al. [12] identified a set of effective designs that allow GCNs to generalize to heterophilous data and handle adversarial attacks [27], and [28] introduced a new GCN model that leverages ideas from belief propagation. Though recent work [7] focused on tackling oversmoothing, it also empirically showed improvements on heterophilous data; also, a PageRank-based model has been shown to perform well under heterophily and alleviate the oversmoothing problem [14]. These empirical observations formed the basis of our work. However, they view the two problems independently and provide analysis from an asymptotic spectral perspective. On the other hand, our work studies the representation dynamics and unveils the connections between oversmoothing and heterophily theoretically and empirically.

TABLE I: Real data: mean accuracy ± stdev over different data splits. Per GNN model, we report the best performance across different layers. Best model per benchmark highlighted in gray. The "†" results (GraphSAGE and FAGCN) are obtained from [12, 22].

	Texas	**Wisconsin**	**Actor**	**Squirrel**	**Chameleon**	**Cornell**	**Citeseer**	**Pubmed**	**Cora**	
Hom. level h	**0.11**	**0.21**	**0.22**	**0.22**	**0.23**	**0.3**	**0.74**	**0.8**	**0.81**	**Avg Rank**
#Nodes	183	251	7,600	5,201	2,277	183	3,327	19,717	2,708	
#Edges	295	466	26,752	198,493	31,421	280	4,676	44,327	5,278	
#Classes	5	5	5	5	5	5	7	3	6	
GGCN (**ours**)	84.86±4.55	86.86±3.29	37.54±1.56	55.17±1.58	71.14±1.84	85.68±6.63	77.14±1.45	89.15±0.37	87.95±1.05	1.78
GPRGNN	78.38±4.36	82.94±4.21	34.63±1.22	31.61±1.24	46.58±1.71	80.27±8.11	77.13±1.67	87.54±0.38	87.95±1.18	5.67
FAGCN†	77.56±6.11	79.41±6.55	34.85±1.61	30.59±1.22	46.44±2.81	78.64±5.47	74.01±1.85	76.57±1.88	86.34±0.67	8.11
H2GCN*	84.86±7.23	87.65±4.98	35.70±1.00	36.48±1.86	60.11±2.15	82.70±5.28	77.11±1.57	89.49±0.38	87.87±1.20	3.89
GCNII*	77.57±3.83	80.39±3.4	37.44±1.30	38.47±1.58	63.86±3.04	77.86±3.79	77.33±1.48	90.15±0.43	88.37±1.25	3.67
Geom-GCN*	66.76±2.72	64.51±3.66	31.59±1.15	38.15±0.92	60.00±2.81	60.54±3.67	78.02±1.15	89.95±0.47	85.35±1.57	6.67
PairNorm	60.27±4.34	48.43±6.14	27.40±1.24	50.44±2.04	62.74±2.82	58.92±3.15	73.59±1.47	87.53±0.44	85.79±1.01	8.44
GraphSAGE†	82.43±6.14	81.18±5.56	34.23±0.99	41.61±0.74	58.73±1.68	75.95±5.01	76.04±1.30	88.45±0.50	86.90±1.04	6.00
GCN	55.14±5.16	51.76±3.06	27.32±1.10	53.43±2.01	64.82±2.24	60.54±5.3	76.50±1.36	88.42±0.5	86.98±1.27	7.00
GAT	52.16±6.63	49.41±4.09	27.44±0.89	40.72±1.55	60.26±2.5	61.89±5.05	76.55±1.23	86.33±0.48	87.30±1.10	7.67
MLP	80.81±4.75	85.29±3.31	36.53±0.70	28.77±1.56	46.21±2.99	81.89±6.40	74.02±1.90	87.16±0.37	75.69±2.00	7.11

TABLE II: Model performance for different layers: mean accuracy ± stdev over different data splits. Per dataset and GNN model, we also report the layer at which the best performance (given in Table I) is achieved. 'OOM': out of memory; 'INS': numerical instability.

Layers	**2**	**4**	**8**	**16**	**32**	**64**	**Best layer**	**2**	**4**	**8**	**16**	**32**	**64**	**Best layer**
	Cora (h=0.81)							**Cornell** (h=0.3)						
GGCN (**ours**)	87.00±1.15	87.48±1.32	87.63±1.33	87.51±1.19	87.95±1.05	87.28±1.41	32	83.78±6.73	83.78±6.16	84.86±5.69	83.78±6.73	83.78±6.51	84.32±5.90	6
GPRGNN	87.93±1.11	87.95±1.18	87.87±1.41	87.26±1.51	87.18±1.29	87.32±1.21	4	76.76±8.22	77.57±7.46	80.27±8.11	78.38±6.04	74.59±7.66	70.00±5.73	8
H2GCN*	87.87±1.20	86.10±1.51	86.18±2.10	OOM	OOM	OOM	2	81.89±5.98	82.70±6.27	80.27±6.63	OOM	OOM	OOM	1
GCNII*	85.35±1.56	85.35±1.48	86.38±0.98	87.12±1.11	87.95±1.23	88.37±1.25	64	67.57±11.34	64.59±9.63	73.24±5.91	77.84±3.97	75.41±5.47	73.78±4.37	16
PairNorm	85.79±1.01	85.07±0.91	84.65±1.09	82.21±2.84	60.32±8.28	44.39±5.60	2	50.27±7.17	53.51±8.00	58.38±5.01	58.38±3.01	58.92±3.15	58.92±3.15	32
Geom-GCN*	85.35±1.57	21.01±2.61	13.98±1.48	13.98±1.48	13.98±1.48	13.98±1.48	2	60.54±3.67	23.78±11.64	12.97±2.91	12.97±2.91	12.97±2.91	12.97±2.91	2
GCN	86.98±1.27	83.24±1.56	31.03±3.08	31.05±2.36	30.76±3.43	31.89±2.08	2	60.54±5.30	59.19±3.30	58.92±3.15	58.92±3.15	58.92±3.15	58.92±3.15	2
GAT	87.30±1.10	86.50±1.20	84.97±1.24	INS	INS	INS	2	61.89±5.05	58.38±4.05	58.38±3.86	INS	INS	INS	2

VII. Conclusion

Our work provides the first theoretical and empirical analysis that unveils the connections between the oversmoothing and heterophily problems for GCNs. Specifically, by theoretically analyzing the interplay of two metrics—the relative degree of a node and heterophily level in its neighborhood—, we introduce a unified perspective to jointly explain the oversmoothing and heterophily problems. We also propose two theory-grounded strategies, structure-based and feature-based edge corrections, to improve GCN robustness against these problems and show that they consistently achieve strong performance on real data.

Acknowledgements

This material is based upon work supported by the NSF under CAREER Grant No. IIS 1845491 and IIS 2212143, and AWS Cloud Credits for Research.

References

[1] Y. Yan, M. Hashemi, K. Swersky, Y. Yang, and D. Koutra, "Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks," *arXiv:2102.06462*, 2021.

[2] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ICLR*, 2017.

[3] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," *ICLR*, 2018.

[4] Y. Yan, J. Zhu, M. Duda, E. Solarz, C. Sripada, and D. Koutra, "GroupINN: Grouping-based interpretable neural network for classification of limited, noisy brain data," in *SIGKDD*, 2019.

[5] Q. Li, Z. Han, and X. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," *AAAI*, 2018.

[6] K. Oono and T. Suzuki, "Graph neural networks exponentially lose expressive power for node classification," *ICLR*, 2019.

[7] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *ICML*, 2020.

[8] G. Wang, R. Ying, J. Huang, and J. Leskovec, "Improving graph attention networks with large margin-based constraints," *arXiv preprint arXiv:1910.11945*, 2019.

[9] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," *AAAI*, 2020.

[10] H. Pei, B. Wei, K. C.-C. Chang, Y. Lei, and B. Yang, "Geom-gcn: Geometric graph convolutional networks," *ICLR*, 2019.

[11] M. E. Newman, "Assortative mixing in networks," *Physical review letters*, vol. 89, no. 20, 2002.

[12] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," *NeurIPS*, 2020.

[13] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos, "Netprobe: a fast and scalable system for fraud detection in online auction networks," in *WWW*, 2007.

[14] E. Chien, J. Peng, P. Li, and O. Milenkovic, "Adaptive universal generalized pagerank graph neural network," in *ICLR*, 2021.

[15] D. Bo, X. Wang, C. Shi, and H. Shen, "Beyond low-frequency information in graph convolutional networks," in *AAAI*, vol. 35, 2021.

[16] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," *ICML*, 2017.

[17] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *ICML*, 2019.

[18] S. Luan, C. Hua, Q. Lu, J. Zhu, M. Zhao, S. Zhang, X.-W. Chang, and D. Precup, "Is heterophily a real nightmare for graph neural networks to do node classification?" *arXiv:2109.05641*, 2021.

[19] Y. Ma, X. Liu, N. Shah, and J. Tang, "Is homophily a necessity for graph neural networks?" *ICLR*, 2022.

[20] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *NeurIPS*, 2017.

[21] L. Zhao and L. Akoglu, "Pairnorm: Tackling oversmoothing in gnns," *ICLR*, 2019.

[22] J. Chen, W. Liu, and J. Pu, "Memory-based message passing: Decoupling the message for propagation from discrimination," in *ICASSP*, 2022.

[23] V. G. Satorras and J. B. Estrach, "Few-shot learning with graph neural networks," *ICLR*, 2018.

[24] Y. Yan, K. Swersky, D. Koutra, P. Ranganathan, and M. Hashemi, "Neural execution engines: Learning to execute subroutines," *NeurIPS*, 2020.

[25] G. Li, M. Muller, A. Thabet, and B. Ghanem, "Deepgcns: Can gcns go as deep as cnns?" in *ICCV*, 2019.

[26] Y. Rong, W. Huang, T. Xu, and J. Huang, "Dropedge: Towards deep graph convolutional networks on node classification," *ICLR*, 2019.

[27] J. Zhu, J. Jin, D. Loveland, M. T. Schaub, and D. Koutra, "How Does Heterophily Impact the Robustness of Graph Neural Networks? Theoretical Connections and Practical Implications," in *KDD*, 2022.

[28] J. Zhu, R. A. Rossi, A. Rao, T. Mai, N. Lipka, N. K. Ahmed, and D. Koutra, "Graph neural networks with heterophily," in *AAAI*, 2021.