A Generalized Neural Diffusion Framework on Graphs

Yibo Li[1], Xiao Wang[2], Hongrui Liu[3], Chuan Shi[1*]

[1]Beijing University of Posts and Telecommunications
[2]Beihang University
[3]Ant Group
{yiboL, shichuan}@bupt.edu.cn, xiao_wang@buaa.edu.cn, liuhongrui.lhr@antgroup.com

Abstract

Recent studies reveal the connection between GNNs and the diffusion process, which motivates many diffusion-based GNNs to be proposed. However, since these two mechanisms are closely related, one fundamental question naturally arises: *Is there a general diffusion framework that can formally unify these GNNs?* The answer to this question can not only deepen our understanding of the learning process of GNNs, but also may open a new door to design a broad new class of GNNs. In this paper, we propose a general diffusion equation framework with the fidelity term, which formally establishes the relationship between the diffusion process with more GNNs. Meanwhile, with this framework, we identify one characteristic of graph diffusion networks, i.e., the current neural diffusion process only corresponds to the first-order diffusion equation. However, by an experimental investigation, we show that the labels of high-order neighbors actually exhibit monophily property, which induces the similarity based on labels among high-order neighbors without requiring the similarity among first-order neighbors. This discovery motives to design a new high-order neighbor-aware diffusion equation, and derive a new type of graph diffusion network (HiD-Net) based on the framework. With the high-order diffusion equation, HiD-Net is more robust against attacks and works on both homophily and heterophily graphs. We not only theoretically analyze the relation between HiD-Net with high-order random walk, but also provide a theoretical convergence guarantee. Extensive experimental results well demonstrate the effectiveness of HiD-Net over state-of-the-art graph diffusion networks.

Introduction

Graphs, such as traffic networks, social networks, citation networks, and molecular networks, are ubiquitous in the real world. Recently, Graph Neural Networks (GNNs), which are able to effectively learn the node representations based on the message-passing manner, have shown great popularity in tackling graph analytics problems. So far, GNNs have significantly promoted the development of graph analysis towards real-world applications. e.g, node classifification (Abu-El-Haija et al. 2019; Lei et al. 2022), link prediction (Kipf and Welling 2016b; You, Ying, and Leskovec 2019), subgraph isomorphism counting (Yu et al. 2023), and graph classifification (Gao and Ji 2019; Zhang et al. 2018).

Some recent studies show that GNNs are in fact intimately connected to diffusion equations (Chamberlain et al. 2021; Wang et al. 2021; Thorpe et al. 2022), which can be considered as information diffusion on graphs. Diffusion equation interprets GNNs from a continuous perspective (Chamberlain et al. 2021) and provides new insights to understand existing GNN architectures, which motives some diffusion-based GNNs. For instance, (Wang et al. 2021) proposes continuous graph diffusion. (Thorpe et al. 2022) utilizes the diffusion process to handle oversmoothing issue. (Song et al. 2022) considers a graph as a discretization of a Riemannian manifold and studies the robustness of the information propagation process on graphs. Diffusion equation can also build a bridge between traditional GNNs and control theory (Zang and Wang 2020). Although the diffusion process and graph convolution are closely related, little effort has been made to answer: *Is there a unified diffusion equation framework to formally unify the current GNN architectures?* A well-informed answer can deepen our understanding of the learning mechanism of GNNs, and may inspire to design a broad new class of GNNs based on diffusion equation.

Actually, (Chamberlain et al. 2021) has explained GCN (Kipf and Welling 2016a) and GAT (Veličković et al. 2017) from diffusion equation. However, with more proposed GNN architectures, it is highly desired to formally revisit the relation between diffusion equation and GNNs. In this paper, we discover that many GNN architectures substantially can be unified with a general diffusion equation with the fidelity term, such as GCN/SGC [22], APPNP[14], GAT [19], AMP [16], DAGNN [15]. Basically, the diffusion equation describes that the change of a node representation depends on the movement of information on graphs from one node to its neighbors, and the fidelity term constraints that the change of a node representation depends on the difference with its initial feature. Furthermore, we show that the unified diffusion framework can also be derived from an energy function, which explains the whole framework as an energy minimization process in a global view. Compared with other unified frameworks (Zhu et al. 2021; Ma et al. 2021), our framework is from the diffusion perspective, which has many advantages. For example, diffusion-based methods are able to address the common plights of graph learning models such as

*Corresponding author.
Copyright © 2024, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

oversmoothing (Chamberlain et al. 2021; Wang et al. 2021). What's more, the diffusion equation can be seen as partial differential equations (PDEs) (Chamberlain et al. 2021), thus introducing many schemes to solve the graph diffusion equation such as explicit scheme, implicit scheme, and multi-step scheme, some of which are more stable and converge faster.

Based on the above findings, we can see that the diffusion process employed by most current GNNs just considers the first-order diffusion equation, which only diffuses messages among 1-hop neighbors. That is, the first-order diffusion has the underlying homophily assumption among 1-hop neighbors. While we empirically discover that the labels of 2-hop neighborhoods actually appear monophily property (Altenburger and Ugander 2018), i.e., nodes may have extreme preferences for a particular attribute which are unrelated to their own attribute and 1-hop neighbors' attribute, but are more likely to be similar with the attribute of their 2-hop neighbors. Simply put, monophily can induce a similarity among 2-hop neighbors without requiring similarity among 1-hop neighbors. So when the 1-hop neighbors are heterophily-dominant or have noise, the 2-hop neighbors will provide more relevant context. Therefore, a more practical diffusion process should take both the first-order and second-order neighbors into account. *How can we design a new type of graph diffusion networks satisfying the above requirement based on our framework?*

In this paper, we design a new high-order neighbor-aware diffusion equation in our proposed diffusion framework, and then derive a High-order Graph Diffusion Network (HiD-Net). Specifically, our model simultaneously combines the first-order and second-order diffusion process, then we regularize the diffusion equation by minimizing the discrepancy between the estimated and the original graph features. The whole diffusion equation is finally integrated into the APPNP architecture. With second-order diffusion equation, HiD-Net is more robust against the attacks and more general on both homophily and heterophily graphs. We theoretically prove that HiD-Net is essentially related with the second-order random walk. We also provide the convergence guarantee that HiD-Net will converge to this random walk's limit distribution as the number of layers increases, and meanwhile, the learned representations do not converge to the same vector over all nodes. The contributions of this paper are summarized as follows:

- We propose a novel generalized diffusion graph framework, consisting of diffusion equation and fidelity term. This framework, formally establishing the relation between diffusion process with a wide variety of GNNs, describes a broad new class of GNNs based on the discretized diffusion equations on graphs and provides new insight to the current graph diffusion/neural networks.
- We discover the monophily property of labels, and based on our diffusion framework, we propose a high-order graph diffusion network, HiD-Net, which is more general and robust. We theoretically build the relation between HiD-Net and second-order random walk, together with the convergence guarantee.
- Our extensive experiments on both the homophily and heterophily graphs clearly show that HiD-Net outperforms the popular GNNs based on diffusion equation.

Related Work

Graph convolutional networks. Recently, graph convolutional network (GCN) models (Bruna et al. 2013; Defferrard, Bresson, and Vandergheynst 2016; Kipf and Welling 2016a; Veličković et al. 2017; Hamilton, Ying, and Leskovec 2017) have been widely studied. Based on the spectrum of graph Laplacian, (Bruna et al. 2013) generalizes CNNs to graph signal. Then (Defferrard, Bresson, and Vandergheynst 2016) further improves the efficiency by employing the Chebyshev expansion of the graph Laplacian. (Kipf and Welling 2016a) proposes to only aggregate the node features from the one-hop neighbors and simplifies the convolution operation. (Veličković et al. 2017) introduces the attention mechanisms to learn aggregation weights adaptively. (Hamilton, Ying, and Leskovec 2017) uses various ways of pooling for aggregation. More works on GNNs can be found in surveys (Wu et al. 2020b; Zhou et al. 2020).

Diffusion equation on graphs. Graph Heat Equation (GHE) (Chung and Graham 1997), which is a well-known generalization of the diffusion equation on graph data, models graph dynamics with applications in spectral graph theory. GRAND (Chamberlain et al. 2021) studies the discretized diffusion PDE on graphs and applies different numerical schemes for their solution. GRAND++ (Thorpe et al. 2022) mitigates the oversmoothing issue of graph neural networks by adding a source term. DGC (Wang et al. 2021) decouples the terminal time and propagation steps of linear GCNs from a perspective of graph diffusion equation, and analyzes why linear GCNs fail to benefit from deep layers. ADC (Zhao et al. 2021) strategies to automatically learn the optimal diffusion time from the data. However, these works focus on specific graph diffusion network, thus there is not a framework to formally unify the GNNs.

The unified GNN framework. (Zhu et al. 2021) establishes a connection between different propagation mechanisms with a unified optimization problem, and finds out that the proposed propagation mechanisms are the optimal solution for optimizing a feature fitting function over a wide class of graph kernels with a graph regularization term. (Ma et al. 2021) establishes the connections between the introduced GNN models and a graph signal denoising problem with Laplacian regularization. It essentially is still an optimization solving framework from the perspective of signal denoising. However, our framework is based on the diffusion equation, where the advantages are two fold: one is that diffusion-based methods are able to address the oversmoothing problem (Chamberlain et al. 2021; Wang et al. 2021). The other is that the diffusion equation can be seen as partial differential equations (PDEs) (Chamberlain et al. 2021) and thus can introduce many schemes that have many good properties such as fast convergence rate and high stability.

The Generalized Diffusion Graph Framework

Notations. Consider an undirected graph as $G = (V, E)$ with adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, where V contains n nodes $\{v_1, \ldots, v_n\}$ and E is the set of edges. The initial

node feature matrix is denoted as $\mathbf{X}^{(0)} \in \mathbb{R}^{n\times q}$, where q is the dimension of node feature. We denote the neighbors of node i at exactly k hops/steps away as $N_k(i)$. For example, $N_1(i) = \{j : (i,j) \in E\}$ are the immediate neighbors of i.

Diffusion is a physical process that equilibrates concentration differences without creating or destroying mass. This physical observation can be easily cast in the diffusion equation, which is a parabolic partial differential equation. Fick's law of diffusion describes the equilibration property (Weickert 1998):

$$J = -f \cdot \nabla u, \tag{1}$$

where J is the diffusion flux, which measures the amount of substance that flows through a unit area during a unit time interval. f is the diffusivity coefficient, which can be constant or depend on time and position. ∇u is the concentration gradient. This equation states that a concentration gradient ∇u causes a flux J which aims to compensate for this gradient. The observation that a change in concentration in any part of the system is due to the influx and outflux of substance into and out of that part of the system can be expressed by the continuity equation:

$$\frac{\partial u}{\partial t} = -\operatorname{div} J, \tag{2}$$

where t denotes the time. Plugging in Fick's law (1) into the continuity equation we end up with the diffusion equation:

$$\frac{\partial u}{\partial t} = \operatorname{div}(f \cdot \nabla u). \tag{3}$$

As div is the sum of the second derivatives in all directions, please note that normal first order derivatives and second order derivatives are on continuous space and can not be generalized directly to graph which is on discrete space. As (Chamberlain et al. 2021) defined, the first derivative is the difference between the feature of a node and its neighbor. And the second derivative can be considered as the difference between the first derivatives of the node itself and its neighbors.

For better illustration, we provide an example. Consider a chain graph in Figure 1, where i, $i+1$, and $i-1$ are the indexes of the nodes. The feature of node i is denoted as x_i.

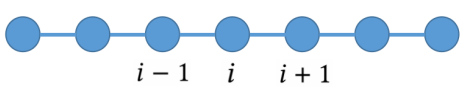


Figure 1: Chain graph

The first order derivatives on node i is defined as $x_{i+1} - x_i$ and $x_i - x_{i-1}$. The diffusion flux from node j to node i at time t on a graph is:

$$J_{ij}^{(t)} = -f \cdot (\nabla x)_{ij}^{(t)} = -f(x_j^{(t)} - x_i^{(t)}). \tag{4}$$

The second order derivative is the difference of first order derivatives: $(x_{i+1} - x_i) - (x_i - x_{i-1}) = (x_{i+1} - x_i) + (x_{i-1} - x_i)$. We notice that the chain graph only has one dimension, so the divergence of node i on a chain graph is equal to its second order derivative: $\operatorname{div}(\nabla x_i) = (x_{i+1} - x_i) + (x_{i-1} - x_i)$.

Thus, on a normal graph, we have the generalized form: $\operatorname{div}(\nabla x_i) = \sum_{j\in N_1(i)}(x_j - x_i)$.

Here we normalize the diffusion process utilizing the degree of the nodes to down-weight the high-degree neighbors, and we have $\operatorname{div}(\nabla x_i) = \sum_{j\in N_1(i)} \frac{\tilde{A}_{ij}}{\sqrt{\tilde{d}_i}\sqrt{\tilde{d}_j}}(x_j - x_i)$, where $\tilde{A}_{ij}$ is the element of $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, and $d_i = \sum_j \tilde{A}_{ij}$.

So the diffusion equation on node i can be defined as:

$$\begin{aligned}\frac{\partial x_i^{(t)}}{\partial t} &= -\operatorname{div} J_{ij}^{(t)} = \operatorname{div}[f(\nabla x)_{ij}^{(t)}] \\ &= f \sum_{j\in N_1(i)} \frac{\tilde{A}_{ij}}{\sqrt{\tilde{d}_i}\sqrt{\tilde{d}_j}}(x_j^{(t)} - x_i^{(t)}).\end{aligned} \tag{5}$$

The diffusion equation models the change of representation $x_i^{(t)}$ with respect to t, which depends on the difference between the nearby nodes, implying that the greater the difference between a node and its neighbors, the faster it changes.

However, how fast $x_i^{(t)}$ changes should not only depend on the representation difference between node i and its neighbors, otherwise, it will cause oversmoothing issue, i.e., as the diffusion process goes by, the nodes are not distinguishable. Based on this phenomenon, we think that the representation change of $x_i^{(t)}$ should be also related with the node feature $x_i^{(0)}$ itself, i.e., if the difference between $x_i^{(t)}$ and $x_i^{(0)}$ is small, the change of $x_i^{(t)}$ should also be small. Then we add another fidelity term and obtain our general graph diffusion framework as follows:

$$\frac{\partial x_i^{(t)}}{\partial t} = \alpha(x_i^{(0)} - x_i^{(t)}) + \beta \operatorname{div}(f(\nabla x)_{ij}^{(t)}), \tag{6}$$

where α, β are coefficients.

Remark 1. (6) can be derived from the energy function:

$$E(x) := \int_\Omega \left(\alpha \cdot (x_i - x_i^{(0)})^2 + \beta \cdot |f(\nabla x)_{ij}|^2\right) d\theta, \tag{7}$$

where θ represents the position of the nodes, and Ω represents the entire graph domain. The corresponding Euler–Lagrange equation, which gives the necessary condition for an extremum of (7), is given by:

$$0 = \alpha(x_i - x_i^{(0)}) + \beta \operatorname{div}(f(\nabla x)_{ij}). \tag{8}$$

(8) can also be regarded as the steady-state equation of (6). Based on the energy function, we can see that (6) constrains space variation and time variation of the diffusion process, indicating that the representations of the graph nodes will not change too much between nearby nodes, as well as not change too much from the initial features.

Remark 2. The framework (6) is closely related to many GNNs, such as GCN/SGC (Wu et al. 2019), APPNP (Klicpera, Bojchevski, and Günnemann 2018), GAT (Veličković et al. 2017), AMP (Liu et al. 2021), DAGNN (Liu, Gao, and Ji 2020), as demonstrated by the following propositions. We provide the proofs of all the subsequent propositions in Appendix.

Proposition 1. With $\alpha = 0$, $\beta = 1$, $\Delta t = 1$ and $f = 1$ in (6), the diffusion process in SGC/GCN is:

$$\frac{\partial x_i^{(t)}}{\partial t} = \operatorname{div}((\nabla x)_{ij}^{(t)}). \tag{9}$$

Proposition 2. Introducing η as coefficient, with $\alpha = 1$, $\beta = 1 - \frac{1}{\eta}$, $\Delta t = 1$ and $f = 1$ in (6), the diffusion process in APPNP is:

$$\frac{\partial x_i^{(t)}}{\partial t} = (x_i^{(0)} - x_i^{(t)}) + (1 - \frac{1}{\eta}) \operatorname{div}((\nabla x)_{ij}^{(t)}). \quad (10)$$

Proposition 3. With $\alpha = 0$, $\beta = 1$, $\Delta t = 1$ and the learned similarity coefficient $f_{ij}^{(t)}$ between nodes i and j at time t in (6), the diffusion process in GAT is:

$$\frac{\partial x_i^{(t)}}{\partial t} = \operatorname{div}(f_{ij}^{(t)} (\nabla x)_{ij}^{(t)}). \quad (11)$$

Proposition 4. With stepsize ϵ and coefficient λ, $\beta_i^{(t)} = \max\left(1 - \frac{\epsilon\lambda}{\left\|(1-2\epsilon(1-\lambda))\mathbf{X}_i^{(t)} + 2\epsilon(1-\lambda)\tilde{\mathbf{A}}\mathbf{X}_i^{(t)} - \left(\mathbf{X}^{(0)}\right)_i\right\|_2}, 0\right)$. Let $\alpha = 1 - \beta_i^{(t)}$, $\beta = 2\epsilon(1-\lambda)\beta_i^{(t)}$, $\Delta t = 1$ and $f = 1$ in (6), the diffusion process in AMP is:

$$\frac{\partial x_i^{(t)}}{\partial t} = (1-\beta_i^{(t)})(x_i^{(0)} - x_i^{(t)}) + 2\epsilon(1-\lambda)\beta_i^{(t)} \operatorname{div}((\nabla x)_{ij}^{(t)}). \quad (12)$$

Proposition 5. With $\alpha + \beta = 1$, $\Delta t = 1$ and the learned coeffiecient $f(t)$ at time t satisfying $\sum_{t=0}^{T} (\beta f(t))^t = \frac{1}{\alpha}$ in (6), the diffusion process in DAGNN is:

$$\frac{\partial x_i^{(t)}}{\partial t} = \alpha(x_i^{(0)} - x_i^{(t)}) + \beta \operatorname{div}(f(t)((\nabla x)_{ij}^{(t)})). \quad (13)$$

The High-order Graph Diffusion Network

High-order Graph Diffusion Equation

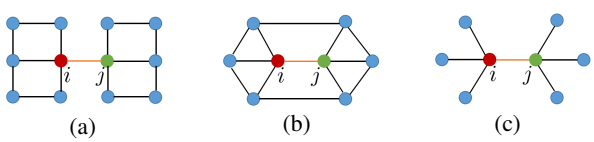


Figure 2: Illustration of the same node pair in different contexts.

In the first-order diffusion process, the diffusion equation only considers 1-hop neighbors. As shown in Figure 2, the nodes in (a), (b), and (c) are the same, but the structures are different. The first-order diffusion flux from node j to node i will be the same, even if the local structure of node i and j is very different. Based on the specific local environments, the diffusion flux should be either different, so as to provide more additional information and make the learned representations more discriminative. To better understand the effect of local structures, we conduct an experiment on six widely used graphs to evaluate the effect of 2-hop neighbors. First, we have the following definition of k-hop neighbor similarity score.

Definition 1. Let y_i be the label of node i, the k-hop neighbor similarity score $h_k = \frac{|\sum_{i \in V} \mathbf{1}_{y_i = \mathbf{O}(\{y_j, j \in N_k(i)\})}|}{|N_k(i)|}$, and $h_{a+b} = \frac{|\sum_{i \in V} \mathbf{1}_{y_i = \mathbf{O}(\{y_j, j \in \{N_a(i), N_b(i)\}\})}|}{|\{N_a(i), N_b(i)\}|}$, where $\mathbf{O}(\{y_j, j \in N_k(i)\})$ represents the element with the highest frequency in $\{y_j, j \in N_k(i)\}$.

The similarity score is based on node labels, and higher similarity score implies the labels of a node and its k-hop neighbors are more consistent. The scores of the six graphs are shown in Table 1. Interestingly, we find that the labels of 2-hop neighbors show monophily property (Altenburger and Ugander 2018), i.e., as can be seen from both the homophily graphs (Cora, Citeseer, Pubmed) and heterophily graphs (Chameleon, Squirrel, Actor), without requiring the similarity among first-order neighbors, the second-order neighbors are more likely to have the same labels.

	cora	citeseer	pubmed	chameleon	squirrel	actor
h_1	0.8634	0.7385	0.7920	0.2530	0.1459	0.2287
h_2	0.8696	0.8476	0.7885	0.3131	0.1600	0.3716
h_{1+2}	0.8737	0.8206	0.7880	0.3070	0.1530	0.3363

Table 1: The similarity scores of six graphs.

To take advantage of 2-hop neighbors, we regularize the gradient ∇x_i utilizing the average gradient of 1-hop neighbors:

$$\overline{(\nabla x)_j} = \operatorname{avg}(\nabla x_{jk}) = \sum_{k \in N_1(j)} \frac{\tilde{A}_{jk}}{\sqrt{\tilde{d}_j}\sqrt{\tilde{d}_k}} (x_k - x_j). \quad (14)$$

We propose the high-order graph diffusion equation:

$$\frac{\partial x_i^{(t)}}{\partial t} = \alpha(x_i^{(0)} - x_i^{(t)}) + \beta \operatorname{div}(f(\nabla x_{ij}^{(t)}) + \gamma \overline{(\nabla x)_j^{(t)}}), \quad (15)$$

where γ is the parameter of the regularization term. The iteration step of (15) is:

$$\begin{aligned} x_i^{t+\Delta t} &= \alpha \Delta t x_i^{(0)} + (1 - \alpha \Delta t) x_i^{(t)} \\ &\quad + \beta \Delta t \operatorname{div}(f(\nabla x_i^{(t)})) + \beta\gamma\Delta t \operatorname{div}(\overline{(\nabla x)_j^{(t)}}), \end{aligned} \quad (16)$$

which is the diffusion-based message passing scheme (DMP) of our model. We can see that DMP utilizes the 2-hop neighbors' information, where the advantages are two-fold: one is that the 2-hop neighbors capture the local environment around a node, even if there are some abnormal features among 1-hop neighbors, their negative effect can still be alleviated by considering a larger neighborhood size, making the learning process more robust. The other is that the monophily property of 2-hop neighbors provides additional stronger correlation with labels, thus even if the 1-hop neighbors may be heterophily, DMP can still make better predictions with information diffused from 2-hop neighbors.

Comparison with other GNN models. Though existing GNN iteration steps can capture high-order connectivity through iterative adjacent message passing, they still have their limitations while having the same time complexity as DMP. DMP is superior because it can utilize the monophily property, adjust the balance between first-order and second-order neighbors, and is based on diffusion equation which has some unique characteristics. More comparisons are discussed in Appendix.

Theoretical Analysis

Next, we theoretically analyze some properties of our diffusion-based message passing scheme.

Definition 2. Consider a surfer walks from node j to i with probability P_{ij}. Let $\mathbb{X}_t$ be a random variable representing the node visited by the surfer at time t. The probability P_{ij} can be represented as a conditional probability $\mathbb{P}\left[\mathbb{X}_t = i \mid \mathbb{X}_{t-\Delta t} = j\right]$. Let

$$P_{ij} = \begin{cases} 1-(\alpha+\beta)\Delta t, & i = j \\ (\beta-\beta\gamma)\Delta t \hat{\tilde{A}}_{ij}, & j \in N_1(i) \\ \beta\gamma\Delta t B_{ij}, & j \in N_2(i) \\ \alpha\Delta t, & restart, \end{cases} \tag{17}$$

where $\hat{\tilde{\mathbf{A}}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$, B_{ij} is the element of $\mathbf{B} = \hat{\tilde{\mathbf{A}}}^2$, and restart means that the node i will teleport back to the initial root node i. Based on the definition, we have the following propositions.

Proposition 6. Given the probability $\mathbb{H}_{ij}^{(t)} = \mathbb{P}\left[\mathbb{X}_t = i \mid \mathbb{X}_0 = j\right]$, DMP (16) is equivalent to the second-order random walk with the transition probability P_{ij} in (17):

$$x_i^{(t)} = \sum_{j \in V} \mathbb{H}_{ij}^{(t)} x_j^{(0)}. \tag{18}$$

Proposition 7. With $f = 1$, $\alpha, \beta, \gamma, \Delta t \in (0, 1]$, DMP (16) converges, i.e., when $t \to \infty$, $\mathbf{X}^{(\infty)} = \alpha((\alpha+\beta)\mathbf{I} - \beta(1-\gamma)\hat{\tilde{\mathbf{A}}} - \beta\gamma\hat{\tilde{\mathbf{A}}}^2)^{-1}\mathbf{X}^{(0)}$.

Proposition 8. When $t \to \infty$, the representations of any two nodes on the graph will not be the same as long as the two nodes have different initial features, i.e., $\forall i, j \in V$, if $x_i^{(0)} \neq x_j^{(0)}$, then $x_i^{(t)} \neq x_j^{(t)}$ as $t \to \infty$.

The proofs of the above propositions are in Appendix.

Our Proposed HiD-Net

To incorporate the high-order graph diffusion DMP (16) into deep neural networks, we introduce High-Order Graph Diffusion Network (HiD-Net). In this work, we follow the decoupled way as proposed in APPNP (Klicpera, Bojchevski, and Günnemann 2018):

$$\mathbf{Y}' = \mathbf{DMP}\left(l_\omega\left(\mathbf{X}^{(0)}\right), t, \Delta t, \alpha, \beta, \gamma\right), \tag{19}$$

where l_ω is a representation learning model such as an MLP network, ω is the learnable parameters in the model. The training objective is to minimize the cross entropy loss defined by the final prediction $\mathbf{Y}'$ and labels for training data. Because of DMP, HiD-Net is more robust and works well on both homophily and heterophily graphs in comparison with other graph diffusion networks.

Time complexity. The time complexity of HiD-Net can be optimized as $\mathcal{O}(n^2\zeta)$, which is the same as the propagation step of GCN, where n is the number of the nodes, ζ is the dimension of the feature vector. We provide the proof in Appendix.

Experiments

Node Classification

Datasets. For comprehensive comparison, we use seven real-world datasets to evaluate the performance of node classification. They are three citation graphs, i.e., Cora, Citeseer, Pubmed (Kipf and Welling 2016a), two Wikipedia networks, i.e., Chameleon and Squirrel (Pei et al. 2020), one Actor co-occurrence network Actor (Pei et al. 2020), one Open Graph Benchmark(OGB) graph ogbn-arxiv(Hu et al. 2020). Among the seven datasets, Cora, Citeseer, Pubmed and ogbn-arxiv are homophily graphs, Chameleon, Squirrel, and Actor are heterophily graphs. Details of datasets are in Appendix.

Baselines. The proposed HiD-Net is compared with several representative GNNs, including three traditional GNNs: GCN (Kipf and Welling 2016a), GAT (Veličković et al. 2017), APPNP (Klicpera, Bojchevski, and Günnemann 2018), and four graph diffusion networks: GRAND (Chamberlain et al. 2021), GRAND++ (Thorpe et al. 2022), ADC (Zhao et al. 2021), DGC (Wang et al. 2021). They are implemented based on their open repositories, where the code can be found in Appendix.

Experimental setting. We perform a hyperparameter search for HiD-Net on all datasets and the details of hyperparameter can be seen in Appendix. For other baseline models: GCN, GAT, APPNP, GRAND, GRAND++, DGC, and ADC, we follow the parameters suggested by (Kipf and Welling 2016a; Veličković et al. 2017; Klicpera, Bojchevski, and Günnemann 2018; Chamberlain et al. 2021; Thorpe et al. 2022; Wang et al. 2021; Zhao et al. 2021) on Cora, Citeseer, and Pubmed, and carefully fine-tune them to get optimal performance on Chameleon, Squirrel, and Actor. For all methods, we randomly run 5 times and report the mean and variance. More detailed experimental settings can be seen in Appendix.

Results. Table 2 summarizes the test results. Please note that OGB prepares standardized evaluators for testing results and it only provides accuracy metric for ogbn-arxiv. As can be seen, HiD-Net outperforms other baselines on seven datasets. Moreover, in comparison with the graph diffusion networks, our HiD-Net is generally better than them with a large margin on heterophily graphs, which indicates that our designed graph diffusion process is more practical for different types of graphs.

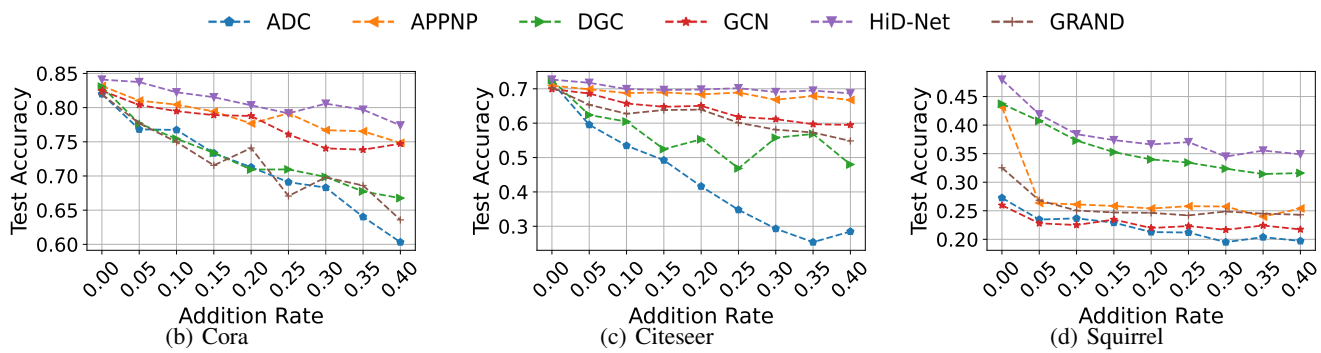
Robustness Analysis

Utilizing the information from 2-hop neighbors, our model is more robust in abnormal situations. We comprehensively evaluate the robustness of our model on three datasets (Cora, Citeseer and Squirrel) in terms of attacks on edges and features, respectively.

Attacks on edges. To attack edges, we adopt random edge deletions or additions following (Chen, Wu, and Zaki 2020; Franceschi et al. 2019). For edge deletions and additions, we randomly remove or add 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40% of the original edges, which retains the connectivity of the attacked graph. Then we perform node classification task. All the experiments are conducted 5 times and we report the average accuracy. The results are plotted in Figure 3 and Figure 4. From the figures, we can see

Datasets	Metric	GCN	GAT	APPNP	GRAND	GRAND++	DGC	ADC	**HiD-Net**
Cora	F1-macro	81.5±0.6	79.7±0.4	82.2 ±0.5	79.4±2.4	81.3±3.3	82.1±0.1	80.0±1.0	**82.8±0.6**
	F1-micro	82.5 ±0.6	80.1±0.8	83.2±0.2	80.1±2.7	82.95 ±1.4	83.1±0.1	81.0±0.7	**84.0±0.6**
	AUC	97.3 ±0.1	96.4 ±0.5	97.5±0.1	96.0±0.3	97.3±0.5	97.2±0.0	97.1±0.1	**97.6±0.0**
Citeseer	F1-macro	66.4 ±0.4	68.5 ±0.3	67.7±0.6	64.9±1.5	66.4±2.6	68.3±0.4	47.0±1.4	**69.5±0.6**
	F1-micro	69.9 ±0.5	72.2 ±0.3	71.0 ±0.4	68.6±1.7	70.9±2.3	72.5±0.4	53.7±1.5	**73.2±0.2**
	AUC	89.9 ±0.4	90.2 ±0.1	90.3 ±0.0	89.5±0.8	91.2±2.8	91.0 ±0.0	87.1±1.1	**91.5±0.1**
Pubmed	F1-macro	78.4 ±0.2	76.7 ±0.5	79.3±0.2	77.5±3.2	78.9±2.5	78.4±0.1	73.7±2.3	**80.1±0.1**
	F1-micro	79.1 ±0.4	77.3 ±0.4	79.9±0.3	78.0±3.2	79.8±1.6	79.2±0.1	74.3±2.3	**81.1±0.1**
	AUC	91.2 ±0.2	90.3±0.5	**92.2±0.1**	90.7±1.6	91.5±2.2	92.0±0.0	89.1±1.9	**92.2±0.1**
Chameleon	F1-macro	38.5 ±2.1	45.0 ±1.0	57.5±1.0	35.7±1.8	46.3±2.4	58.0±0.1	32.6±0.6	**61.0±0.3**
	F1-micro	41.8 ±1.2	44.4 ±1.9	57.1±1.4	37.7±1.5	45.7±3.4	58.2±0.1	33.2±0.5	**60.8±0.7**
	AUC	69.8±0.5	75.5 ±1.0	85.0±0.6	69.0 ±1.3	74.8 ±2.8	82.4±0.0	63.7±0.8	**85.2±0.3**
Squirrel	F1-macro	25.2 ±1.2	26.5 ±1.3	41.1±1.1	24.7±2.0	30.5±3.7	42.1±0.4	24.7±1.2	**47.5±0.9**
	F1-micro	25.8 ±0.8	27.3. ±0.7	43.2±1.0	28.6 ±1.0	34.6 ±2.5	43.1±0.3	25.4±1.0	**48.4±0.8**
	AUC	57.5 ±0.5	58.2±1.1	78.9±0.3	60.2±1.0	65.6±1.4	74.3±0.0	55.0±2.4	**79.4±0.3**
Actor	F1-macro	21.5±0.4	19.7 ±0.8	30.3±4.7	28.0±1.1	30.4±1.1	**31.6±0.0**	20.0±0.6	25.7±0.44
	F1-micro	29.2 ±0.6	27.1 ±0.5	33.2±0.6	32.5±1.0	33.7±2.3	34.1±0.0	25.5±0.5	**34.7±0.4**
	AUC	58.0 ±0.5	55.8±0.4	64.8±0.1	56.2±2.0	60.8.2±0.6	64.7±0.0	53.8±0.1	**68.1±0.2**
ogbn-arxiv	Accuracy	71.5±0.3	71.6±0.5	71.2 ±0.3	71.7 ±0.1	71.9 ±0.6	70.9±0.2	70.0±0.1	**72.2±0.1**

Table 2: Quantitative results (%$\pm\sigma$) on node classification. (bold: best)



(b) Cora (c) Citeseer (d) Squirrel

Figure 3: Results of different models under random edge addition.

that as the addition or deletion rate rises, the performances on three datasets of all the models degenerate, and HiD-Net consistently outperforms other baselines.

Attacks on features. To attack features, we inject random perturbation into the node features as in (Wu et al. 2020a). Firstly, we sample a noise matrix $\mathbf{M} \in \mathbb{R}^{n\times q}$, where each entry in $\mathbf{M}$ is sampled from the normal distribution $N(0,1)$. Then, we calculate reference amplitude r, which is the mean of the maximal value of each node's feature. We add Gaussian noise $\mu \cdot r \cdot \mathbf{M}$ to the original feature matrix, and get the attacked feature matrix, where μ is the noise ratio. The results are reported in Figure 5. Again, HiD-Net consistently outperforms all other baselines under different perturbation rates by a margin for three datasets.

Non-over-smoothing with Increasing Steps

To demonstrate that our model solves the oversmoothing problem compared with other graph diffusion networks, we test different graph diffusion models with increasing propagation step k from 2 to 20. Baselines include DGC, ADC and GRAND. The results are plotted in Figure 7. We can see that with the increase of k, HiD-Net consistently performs better than other baselines.

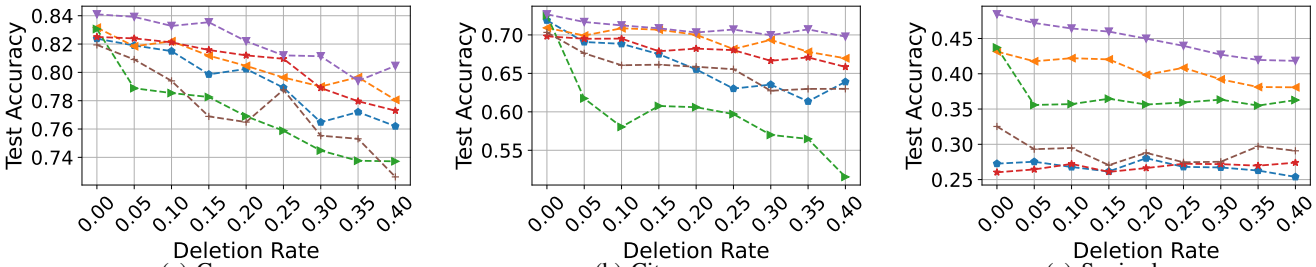
Parameter Study

In this section, we investigate the sensitivity of parameters on all datasets.

Analysis of α. We test the effect of α in (16), and vary it from 0 to 1. From Figure 8(a) we can see that with the increase of α, the performances of citation graphs rise first and then start to drop slowly, the performances of Chameleon and Squirrel have not changed too much, and the performance of Actor first rises and then remains unchanged. As citation graphs are more homophily, we need to focus less on the node itself, implying a small α, while on heterophily graphs, we need to focus more on the node itself.

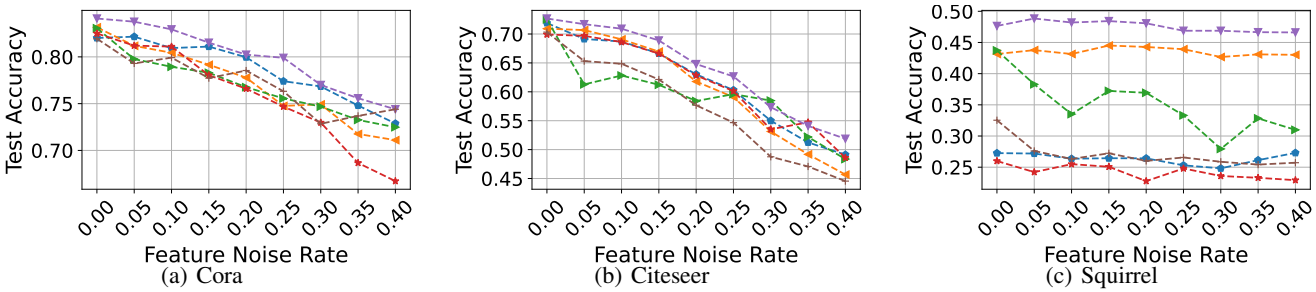
Analysis of β. In order to check the impact of the diffusion term, we study the performance of HiD-Net with β varying from 0 to 1. The results are shown in Figure 8(b). We can see that as the value of β increases, the accuracies generally increase, while the accuracy on Actor remains relatively stable, implying that the features diffused from 1-hop and 2-hop neighbors are very informative.

Analysis of γ. Finally we test the effect of γ in (16) and vary it from 0 to 0.6. With the increase of γ, the accuracies on different datasets do not change much, so we just separately plot each dataset for a clearer illustration here. As can be



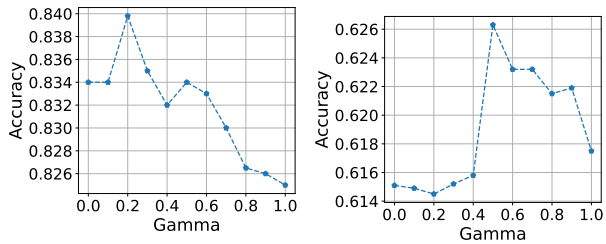
(a) Cora (b) Citeseer (c) Squirrel

Figure 4: Results of different models under random edge deletion.



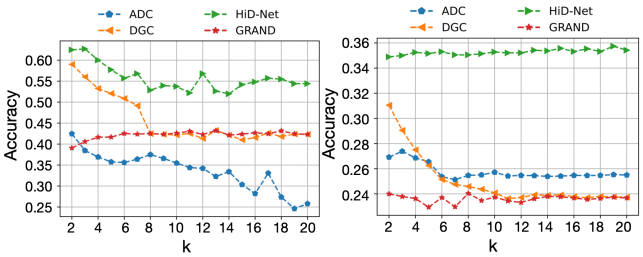
(a) Cora (b) Citeseer (c) Squirrel

Figure 5: Results of different models under random feature perturbation.



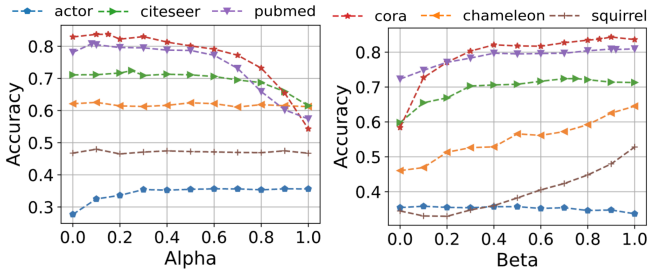
(a) Cora (b) Chameleon

Figure 6: Analysis of parameter γ.



(a) Chameleon (b) Actor

Figure 7: Non-over-smoothing with increasing steps.



(a) α (b) β

Figure 8: Analysis of parameter α and β.

seen in Figure 6, with the increase of γ, the performance on Cora and Chameleon rises first and then drops, and different graphs have different best choices of γ. The results on other datasets are shown in Appendix.

Conclusion

In this paper, we propose a generalized diffusion graph framework, which establishes the relation between diffusion equation with different GNNs. Our framework reveals that current graph diffusion networks mainly consider the first-order diffusion equation, then based on our finding of the monophily property of labels, we derive a novel high-order diffusion graph network (HiD-Net). HiD-Net is more robust and general on both homophily and heterophily graphs. Extensive experimental results verify the effectiveness of HiD-Net. One potential issue is that our model utilizes a constant diffusivity coefficient, and a future direction is to explore a learnable diffusivity coefficient depending on time and space. Our work formally points out the relation between diffusion equation with a wide variety of GNNs. Considering that previous GNNs are designed mainly based on spatial or spectral strategies, this new framework may open a new path to understanding and deriving novel GNNs. We believe that more insights from the research community on the diffusion process will hold great potential for the GNN community in the future.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (No. U20B2045, 62192784, U22B2038, 62002029, 62172052, 62322203).

References

Abu-El-Haija, S.; Perozzi, B.; Kapoor, A.; Alipourfard, N.; Lerman, K.; Harutyunyan, H.; Ver Steeg, G.; and Galstyan, A. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, 21–29. PMLR.

Altenburger, K. M.; and Ugander, J. 2018. Monophily in social networks introduces similarity among friends-of-friends. *Nature human behaviour*, 2(4): 284–290.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Chamberlain, B.; Rowbottom, J.; Gorinova, M. I.; Bronstein, M.; Webb, S.; and Rossi, E. 2021. Grand: Graph neural diffusion. In *International Conference on Machine Learning*, 1407–1418. PMLR.

Chen, Y.; Wu, L.; and Zaki, M. 2020. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in Neural Information Processing Systems*, 33: 19314–19326.

Chung, F. R.; and Graham, F. C. 1997. *Spectral graph theory*. 92. American Mathematical Soc.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Franceschi, L.; Niepert, M.; Pontil, M.; and He, X. 2019. Learning discrete structures for graph neural networks. In *International conference on machine learning*, 1972–1982. PMLR.

Gao, H.; and Ji, S. 2019. Graph u-nets. In *international conference on machine learning*, 2083–2092. PMLR.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33: 22118–22133.

Kipf, T. N.; and Welling, M. 2016a. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N.; and Welling, M. 2016b. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Klicpera, J.; Bojchevski, A.; and Günnemann, S. 2018. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*.

Lei, R.; Wang, Z.; Li, Y.; Ding, B.; and Wei, Z. 2022. EvenNet: Ignoring Odd-Hop Neighbors Improves Robustness of Graph Neural Networks. In *NeurIPS*.

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

Liu, X.; Ding, J.; Jin, W.; Xu, H.; Ma, Y.; Liu, Z.; and Tang, J. 2021. Graph Neural Networks with Adaptive Residual. *Advances in Neural Information Processing Systems*, 34.

Ma, Y.; Liu, X.; Zhao, T.; Liu, Y.; Tang, J.; and Shah, N. 2021. A unified view on graph neural networks as graph signal denoising. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 1202–1211.

Pei, H.; Wei, B.; Chang, K. C.-C.; Lei, Y.; and Yang, B. 2020. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*.

Song, Y.; Kang, Q.; Wang, S.; Kai, Z.; and Tay, W. P. 2022. On the robustness of graph neural diffusion to topology perturbations. *arXiv preprint arXiv:2209.07754*.

Thorpe, M.; Nguyen, T. M.; Xia, H.; Strohmer, T.; Bertozzi, A.; Osher, S.; and Wang, B. 2022. GRAND++: Graph neural diffusion with a source term. In *International Conference on Learning Representations*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, Y.; Wang, Y.; Yang, J.; and Lin, Z. 2021. Dissecting the diffusion process in linear graph convolutional networks. *Advances in Neural Information Processing Systems*, 34.

Weickert, J. 1998. *Anisotropic diffusion in image processing*, volume 1. Teubner Stuttgart.

Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*, 6861–6871. PMLR.

Wu, T.; Ren, H.; Li, P.; and Leskovec, J. 2020a. Graph information bottleneck. *Advances in Neural Information Processing Systems*, 33: 20437–20448.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020b. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1): 4–24.

You, J.; Ying, R.; and Leskovec, J. 2019. Position-aware graph neural networks. In *International Conference on Machine Learning*, 7134–7143. PMLR.

Yu, X.; Liu, Z.; Fang, Y.; and Zhang, X. 2023. Learning to count isomorphisms with graph neural networks. *arXiv preprint arXiv:2302.03266*.

Zang, C.; and Wang, F. 2020. Differential Deep Learning on Graphs and its Applications.

Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018. An end-to-end deep learning architecture for graph classification. In *Thirty-second AAAI conference on artificial intelligence*.

Zhao, J.; Dong, Y.; Ding, M.; Kharlamov, E.; and Tang, J. 2021. Adaptive Diffusion in Graph Neural Networks. *Advances in Neural Information Processing Systems*, 34.

Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; and Sun, M. 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1: 57–81.

Zhu, M.; Wang, X.; Shi, C.; Ji, H.; and Cui, P. 2021. Interpreting and unifying graph neural networks with an optimization framework. In *Proceedings of the Web Conference 2021*, 1215–1226.