ADAPTIVE UNIVERSAL GENERALIZED PAGERANK GRAPH NEURAL NETWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

In many important graph data processing applications the acquired information includes both node features and observations of the graph topology. Graph neural networks (GNNs) are designed to exploit both sources of evidence but they do not optimally trade-off their utility and integrate them in a manner that is also universal. Here, universality refers to independence on homophily or heterophily graph assumptions. We address these issues by introducing a new Generalized PageRank (GPR) GNN architecture that adaptively learns the GPR weights so as to jointly optimize node feature and topological information extraction, regardless of the extent to which the node labels are homophilic or heterophilic. Learned GPR weights automatically adjust to the node label pattern, irrelevant on the type of initialization, and thereby guarantee excellent learning performance for label patterns that are usually hard to handle. Furthermore, they allow one to avoid feature over-smoothing, a process which renders feature information nondiscriminative, without requiring the network to be shallow. Our accompanying theoretical analysis of the GPR-GNN method is facilitated by novel synthetic benchmark datasets generated by the so-called contextual stochastic block model. We also compare the performance of our GNN architecture with that of several state-of-the-art GNNs on the problem of node-classification, using well-known benchmark homophilic and heterophilic datasets. The results demonstrate that GPR-GNN offers significant performance improvement compared to existing techniques on both synthetic and benchmark data.

1 INTRODUCTION

Graph-centered machine learning has received significant interest in recent years due to the ubiquity of graph-structured data and its importance in solving numerous real-world problems such as semi-supervised node classification and graph classification (Zhu, 2005; Shervashidze et al., 2011; Lü & Zhou, 2011). Usually, the data at hand contains two sources of information: Node features and graph topology. As an example, in social networks, nodes represent users that have different combinations of interests and properties captured by their corresponding feature vectors; edges on the other hand document observable friendship and collaboration relations that may or may not depend on the node features. Hence, learning methods that are able to simultaneously and adaptively exploit node features and the graph topology are highly desirable as they make use of their latent connections and thereby improve learning on graphs.

Graph neural networks (GNN) leverage their representational power to provide state-of-the-art performance when addressing the above described application domains. Many GNNs use message passing (Gilmer et al., 2017; Battaglia et al., 2018) to manipulate node features and graph topology. They are constructed by stacking (graph) neural network layers which essentially propagate and transform node features over the given graph topology. Different types of layers have been proposed and used in practice, including graph convolutional layers (GCN) (Bruna et al., 2014; Kipf & Welling, 2017), graph attention layers (GAT) (Velickovic et al., 2018) and many others (Hamilton et al., 2017; Wijesinghe & Wang, 2019; Zeng et al., 2020; Abu-El-Haija et al., 2019).

However, most of the existing GNN architectures have two fundamental weaknesses which restrict their learning ability on general graph-structured data. First, most of them seem to be tailor-made to work on *homophilic (associative) graphs*. The homophily principle (McPherson et al., 2001) in the

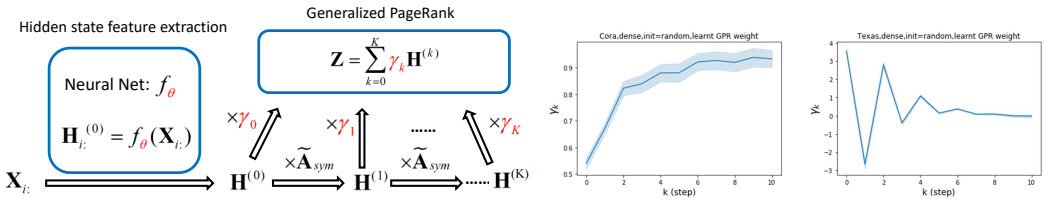
context of node classification asserts that nodes from the same class tend to form edges. Homophily is also a common assumption in graph clustering (Von Luxburg, 2007; Tsourakakis, 2015; Dau & Milenkovic, 2017) and in many GNNs design (Klicpera et al., 2018). Methods developed for homophilic graphs are nonuniversal in so far that they fail to properly solve learning problems on *heterophilic (disassortative) graphs* (Pei et al., 2019; Bojchevski et al., 2019; 2020). In heterophilic graphs, nodes with distinct labels are more likely to link together (For example, many people tend to preferentially connect with people of the opposite sex in dating graphs, different classes of amino acids are more likely to connect within many protein structures (Zhu et al., 2020) etc). GNNs model the homophily principle by aggregating node features within graph neighborhoods. For this purpose, they use different mechanisms such as averaging in each network layer. Neighborhood aggregation is problematic and significantly more difficult for heterophilic graphs (Jia & Benson, 2020).

Second, most of the existing GNNs fail to be "deep enough". Although in principle an arbitrary number of layers may be stacked, practical models are usually shallow (including 2-4 layers) as these architectures are known to achieve better empirical performance than deep networks. A widely accepted explanation for the performance degradation of GNNs with increasing depth is *feature-over-smoothing,* which may be intuitively explained as follows. The process of GNN feature propagating represents a form of random walks on "feature graphs," and under proper conditions, such random walks converge with exponential rate to their stationary points. This essentially levels the expressive power of the features and renders them nondiscriminative. This intuitive reasoning was first described for linear settings in Li et al. (2018) and has been recently studied in Oono & Suzuki (2020) for a setting involving nonlinear rectifiers.

We address these two described weaknesses by combining GNNs with Generalized PageRank techniques (GPR) within a new model termed GPR-GNN. The GPR-GNN architecture is designed to first learn the hidden features and then to propagate them via GPR techniques. The focal component of the network is the GPR procedure that associates each step of feature propagation with a *learnable weight.* The weights depend on the contributions of different steps during the information propagation procedure, and they can be both positive and negative. This departures from common nonnegativity assumptions (Klicpera et al., 2018) allows for the signs of the weights to adapt to the homophily/heterophily structure of the underlying graphs. The amplitudes of the weights trade-off the degree of smoothing of node features and the aggregation power of topological features. These traits do not change with the choice of the initialization procedure and elucidate the process used to combine node features and the graph structure so as to achieve (near)-optimal predictions. In summary, the GPR-GNN method can simultaneously learn the node label patterns of disparate classes of graphs and prevent feature over-smoothing.

The excellent performance of GPR-GNN is demonstrated empirically, on real world datasets, and further supported through a number of theoretical findings. In the latter setting, we show that the GPR procedure relates to general polynomial graph filtering, which can naturally deal with both high and low frequency parts of the graph signals. In contrast, recent GNN models that utilize Personalized PageRanks (PPR) with fixed weights (Wu et al., 2019; Klicpera et al., 2018; 2019) inevitably act as low-pass filters. Thus, they fail to learn the labels of heterophilic graphs. We also establish that GPR-GNN can provably mitigate the feature-over-smoothing issue in an adaptive manner even after large-step propagation. Hence, the method is able to make use of informative large-step propagation.

To test the performance of GPR-GNN on homophilic and heterophilic node label patterns and determine the trade-off between node and topological feature exploration, we first describe the recently proposed *contextual stochastic block model* (cSBM) (Deshpande et al., 2018). The cSBM allows for smoothly controlling the "informativeness ratio" between node features and graph topology, where the graph can vary from being highly homophilic to highly heterophilic. We show that GPR-GNN outperforms all other baseline methods for the task of semi-supervised node classification on the cSBM consistently from strong homophily to strong heterophily. We then proceed to show that GPR-GNN offers state-of-the-art performance on node-classification benchmark real-world datasets which contain both homophilic and heterophilic graphs. Due to the space limit, we put all proofs, formal theorem statements, and the conclusion section in the Supplement.



(a) Illustration of our proposed GPR-GNN model. (b) Cora, $\mathcal{H}(G) = 0.656$ (c) Texas, $\mathcal{H}(G) = 0.016$

Figure 1: (a) Hidden state feature extraction is performed by a neural networks using individual node features propagated via GPR. Note that both the GPR weights γ_k and parameter set $\{\theta\}$ of the neural network are learned simultaneously in an end-to-end fashion (as indicated in red). (b)-(c) The learnt GPR weights of the GPR-GNN on real world datasets. Cora is homophilic while Texas is heterophilic (Here, $\mathcal{H}$ stands for the level of homophily defined below). An interesting trend may be observed: For the heterophilic case the weights alternate from positive to negative with dampening amplitudes (more examples are provided in Section 5). The shaded region corresponds to a 95% confidence interval.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph with nodes V and edges E. Let n denote the number of nodes, assumed to belong to one of $C \geq 2$ classes. The nodes are associated with the node feature matrix $\mathbf{X} \in \mathbb{R}^{n \times f}$, where f denotes the number of features per node. Throughout the paper, we use $\mathbf{X}_{i:}$ to indicate the ith row and $\mathbf{X}_{:j}$ to indicate the jth column of the matrix $\mathbf{X}$, respectively. The symbol δ_{ij} is reserved for the Kronecker delta function. The graph G is described by the adjacency matrix $\mathbf{A}$, while $\tilde{\mathbf{A}}$ stands for the adjacency matrix for a graph with added self-loops. We let $\tilde{\mathbf{D}}$ be the diagonal degree matrix of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{A}}_{\text{sym}} = \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}$ denote the symmetric normalized adjacency matrix with self-loops.

3 GPR-GNNs: Motivation and Contributions

Generalized PageRanks. Generalized PageRank (GPR) methods were first used in the context of unsupervised graph clustering where they showed significant performance improvements over Personalized PageRank (Kloumann et al., 2017; Li et al., 2019). The operational principles of GPRs can be succinctly described as follows. Given a seed node $s \in V$ in some cluster of the graph, a one-dimensional feature vector $\mathbf{H}^{(0)} \in \mathbb{R}^{n \times 1}$ is initialized according to $\mathbf{H}_{v:}^{(0)} = \delta_{vs}$. The GPR score is defined as $\sum_{k=0}^{\infty} \gamma_k \tilde{\mathbf{A}}_{\text{sym}}^k \mathbf{H}^{(0)} = \sum_{k=0}^{\infty} \gamma_k \mathbf{H}^{(k)}$, where the parameters γ_k, $k = 0, 1, 2, \ldots$ are referred to as the GPR weights. Clustering of the graph is performed locally by thresholding the GPR score. The work in Li et al. (2019) recently introduced and theoretically analyzed a special form of GPR termed *Inverse PR* (IPR) and showed that long random walk paths are more beneficial for clustering then previously assumed, provided that the GPR weights are properly selected (Note that IPR was developed for homophilic graphs and optimal GPR weights for heterophilic graphs remain unknown).

Equivalence of the GPR method and polynomial graph filtering. If we truncate the infinite sum in the definition of GPR at some natural number K, $\sum_{k=0}^{K} \gamma_k \tilde{\mathbf{A}}_{\text{sym}}^k$ corresponds to a polynomial graph filter of order K. Thus, learning the optimal GPR weights is equivalent to learning the optimal polynomial graph filter. Note that one can approximate any graph filter using a polynomial graph filter (Shuman et al., 2013) and hence the GPR method is able to deal with a large range of different node label patterns. Also, increasing K allows one to better approximate the underlying optimal graph filter. This once again shows that large-step propagation is beneficial.

Universality with respect to node label patterns: Homophily versus heterophily. In their recent work, Pei et al. (2019) proposed an index to measure the level of homophily of nodes in a graph $\mathcal{H}(G) = \frac{1}{|V|} \sum_{v \in V} \frac{\text{Number of neighbors of } v \in V \text{ that have the same label as } v}{\text{Number of neighbors of } v}$. Note that $\mathcal{H}(G) \to 1$ corresponds to strong homophily while $\mathcal{H}(G) \to 0$ indicates strong heterophily. Figures 1 (b) and (c) plot the GPR weights learnt by our GPR-GNN method on a homophilic (Cora) and heterophilic

(Texas) dataset. The learnt GPR weights from Cora match the behavior of IPR (Li et al., 2019), which verifies that large-step propagation is indeed of great importance for homophilic graphs. The GPR weights learnt from Texas behave significantly differently from all known PR variants, taking a number of negative values. These differences in weight patterns are observed under random initialization, demonstrating that the weights are actually learned by the network and not forced by specific initialization. Furthermore, the large difference in the GPR weights for these two graph models illustrates the learning power of GPR-GNN and their universal adaptability.

The over-smoothing problem. One of the key components in most GNN models is the graph convolutional layer, described by

$$\mathbf{H}_{\text{GCN}}^{(k)} = \text{ReLU}\left(\tilde{\mathbf{A}}_{\text{sym}}\mathbf{H}_{\text{GCN}}^{(k-1)}\mathbf{W}^{(k)}\right),\ \hat{\mathbf{P}}_{\text{GCN}} = \text{softmax}\left(\tilde{\mathbf{A}}_{\text{sym}}\mathbf{H}_{\text{GCN}}^{(K-1)}\mathbf{W}^{(k)}\right),$$

where $\mathbf{H}_{\text{GCN}}^{(0)} = \mathbf{X}$ and $\mathbf{W}^{(k)}$ represents the trainable weight matrix for the kth layer. The key issue that limits stacking multiple layers is the over-smoothing phenomenon: If one were to remove ReLU in the above expression, $\lim_{k\to\infty} \tilde{\mathbf{A}}_{\text{sym}}^{k}\mathbf{H}^{(0)} = \mathbf{H}^{(\infty)}$, where each row of $\mathbf{H}^{(\infty)}$ only depends on the degree of the corresponding node, provided that the graph is irreducible and aperiodic. This shows that the model looses discriminative information provided by the node features as the number of layers increases.

Mitigating graph heterophily and over-smoothing issues with the GPR-GNN model. GPR-GNN first extracts hidden state features for each node and then uses GPR to propagate them. The GPR-GNN process can be mathematically described as:

$$\hat{\mathbf{P}} = \text{softmax}(\mathbf{Z}),\ \mathbf{Z} = \sum_{k=0}^{K}\gamma_k\mathbf{H}^{(k)},\ \mathbf{H}^{(k)} = \tilde{\mathbf{A}}_{\text{sym}}\mathbf{H}^{(k-1)},\ \mathbf{H}_{i:}^{(0)} = f_\theta(\mathbf{X}_{i:}), \tag{1}$$

where $f_\theta(.)$ represents a neural network with parameter set $\{\theta\}$ that generates the hidden state features $\mathbf{H}^{(0)}$. The GPR weights γ_k are trained together with $\{\theta\}$ in an end-to-end fashion. The GPR-GNN model is easy to interpret: As already pointed out, GPR-GNN has the ability to adaptively control the contribution of each propagation step and adjust it to the node label pattern. Examining the learnt GPR weights also helps with elucidating the properties of the topological information of a graph (i.e., determining the optimal polynomial graph filter), as illustrated in Figure 1 (b) and (c).

Placing GPR-GNNs in the context of related prior work. Among the methods that differ from repeated stacking of GCN layers, APPNP (Klicpera et al., 2018) represents one of the state-of-the-art GNNs that is related to our GPR-GNN approach. It can be easily seen that APPNP as well as SGC (Wu et al., 2019) are special cases of our model since APPNP fixes $\gamma_k = \alpha(1-\alpha)^k, \gamma_K = (1-\alpha)^K$, while SGC removes all nonlinearities with $\gamma_k = \delta_{kK}$, respectively. These two weight choices correspond to Personalized PageRank (PPR) (Jeh & Widom, 2003), which is known to be suboptimal compared to the IPR framework when applied to homophilic node classification (Li et al., 2019). Fixing the GPR weights makes the model unable to adaptively learn the optimal propagation rules which is of crucial importance: As we will show in Section 4, the fixed PPR weights corresponds to low-pass graph filters which makes them inadequate for learning on heterophilic graphs. The recent work (Klicpera et al., 2018) showed that fixed PPR weights (APPNP) can also provably resolve the over-smoothing problem. However, the way APPNP prevents over-smoothing is independent on the node label information. In contrast, the escape of GPR-GNN from over-smoothing is guided by the node label information (Theorem 4.2). A detailed discussion of this phenomena along with illustrative examples is delegated to the Supplement.

Among the GCN-like models, JK-Net (Xu et al., 2018) exhibits some similarities with GPR-GNN. It also aggregates the outputs of different GCN layers to arrive at the final output. On the other hand, the GCN-Cheby method (Defferrard et al., 2016; Kipf & Welling, 2017) is related to polynomial graph filtering, where each convolutional layer propagates multiple steps and the graph filter is related to Chebyshev polynomials. In both cases, the depth of the models is limited in practice (Klicpera et al., 2018) and they are not easy to interpret as our GPR-GNN method. Some prior work also emphasizes adaptively learning the importance of different steps (Abu-El-Haija et al., 2018; Berberidis et al., 2018). Nevertheless, none of the above works is applicable for semi-supervised learning with GNNs and considers heterophilic graphs.

4 THEORETICAL PROPERTIES OF GPR-GNNS

Graph filtering aspects of GPR-GNNs. As mentioned in Section 3, the GPR component of the network may be viewed as a polynomial graph filter. Let $\tilde{\mathbf{A}}_{\text{sym}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ be the eigenvalue decomposition of $\tilde{\mathbf{A}}_{\text{sym}}$. Then, the corresponding polynomial graph filter equals $\sum_{k=0}^{K} \gamma_k \tilde{\mathbf{A}}_{\text{sym}}^k = \mathbf{U} g_{\gamma,K}(\mathbf{\Lambda})\mathbf{U}^T$, where $g_{\gamma,K}(\mathbf{\Lambda})$ is applied element-wise and $g_{\gamma,K}(\lambda) = \sum_{k=0}^{K} \gamma_k \lambda^k$. We established the following result.

Theorem 4.1 (Informal)**.** *Assume that the graph G is connected. If $\gamma_k \geq 0$ $\forall k \in \{0, 1, ..., K\}$, $\sum_{k=0}^{K} \gamma_k = 1$ and $\exists k' > 0$ such that $\gamma_{k'} > 0$, then $g_{\gamma,K}(\cdot)$ is a low-pass graph filter. Also, if $\gamma_k = (-\alpha)^k, \alpha \in (0, 1)$ and K is large enough, then $g_{\gamma,K}(\cdot)$ is a high-pass graph filter.*

By Theorem 4.1 and from our discussion in Section 3, we know that both APPNP and SGC will invariably suppress the high frequency components. Thus, they are inadequate for use on heterophilic graphs. In contrast, if one allows γ_k to be negative and learned adaptively the graph filter will pass relevant high frequencies. This is what allows GPR-GNN to perform exceptionally well on heterophilic graphs (see Figure 2(c)).

GPR-GNN can escape from over-smoothing. As already emphasized, one crucial innovation of the GPR-GNN method is to make the GPR weights adaptively learnable, which allows GPR-GNN to avoid over-smoothing and trade node and topology feature informativeness. Intuitively, when large-step propagation is not beneficial, it increases the training loss. Hence, the corresponding GPR weights should decay in magnitude. This observation is captured by the following result, whose more formal statement and proof are delegated to the Supplement due to space limitations.

Theorem 4.2 (Informal)**.** *Assume the graph G is connected and the training set contains nodes from each of the classes. Also assume that k' is large enough so that the over-smoothing effect occurs for $\mathbf{H}^{(k)}, \forall k \geq k'$ which dominate the contribution to the final output $\mathbf{Z}$. Then, the gradients of γ_k and $\gamma_{k'}$ are identical in sign for all $k \geq k'$.*

Theorem 4.2 shows that as long as over-smoothing happens, $|\gamma_k|$ will approach 0 for all $k \geq k'$ when we use an optimizer such as stochastic gradient descent (SGD) which has a suitable learning rate decay. This reduces the contribution of the corresponding steps $\mathbf{H}^{(k)}$ in the final output $\mathbf{Z}$. When the weights $|\gamma_k|$ are small enough so that $\mathbf{H}^{(k)}$ no longer dominates the value of the final output $\mathbf{Z}$, the over-smoothing effect is eliminated.

5 RESULTS FOR NEW CSBM SYNTHETIC AND REAL-WORLD DATASETS

Synthetic data. In order to test the ability of label learning of GNNs on graphs with arbitrary levels of homophily and heterophily, we propose to use cSBMs (Deshpande et al., 2018) to generate synthetic graphs. We consider the case with two equal-size classes. In cSBMs, the node features are Gaussian random vectors, where the mean of the Gaussian depends on the community assignment. The difference of the means is controlled by a parameter μ, while the difference of the edge densities in the communities and between the communities is controlled by a parameter λ. Hence μ and λ capture the "relative informativeness" of node features and the graph topology, respectively. Moreover, positive λ's correspond to homophilic graphs while negative λ's correspond to heterophilic graphs. The information-theoretic limits of reconstruction for the cSBM are characterized in Deshpande et al. (2018). The results show that, asymptotically, one needs $\lambda^2 + \mu^2/\xi > 1$ to ensure a vanishing ratio of the misclassified nodes and the total number of nodes, where $\xi = n/f$ and f as before denotes the dimension of the node feature vector.

Note that given a tolerance value $\epsilon > 0$, $\lambda^2 + \mu^2/\xi = 1 + \epsilon$ is an arc of an ellipsoid for which $\lambda \geq 0$ and $\mu \geq 0$. To fairly and continuously control the extent of information carried by the node features and graph topology, we introduce a parameter $\phi = \arctan(\frac{\lambda\sqrt{\xi}}{\mu}) \times \frac{2}{\pi}$. The setting $\phi = 0$ indicates that only node features are informative, while $|\phi| = 1$ indicates that only the graph topology is informative. Moreover, $\phi = 1$ corresponds to strongly homophilic graphs while $\phi = -1$ corresponds to strongly heterophilic graphs. Note that the values ϕ and $-\phi$ convey the same amount of information regarding graph topology. This is due to the fact that $\lambda^2 = (-\lambda)^2$. Ideally, GNNs that are able to optimally learn on both homophilic and heterophilic graph should

have similar performances for ϕ and $-\phi$. Due to space limitation we refer the interested reader to (Deshpande et al., 2018) for a review of all formal theoretical results and only outline the cSBM properties needed for our analysis. Additional information is also available in the Supplement.

Our experimental setup examines the semi-supervised node classification task in the transductive setting. We consider two different choices for the random split into training/validation/test samples, which we call sparse splitting ($2.5\%/2.5\%/95\%$) and dense splitting ($60\%/20\%/20\%$), respectively. The sparse splittnig is more similar to the original semi-supervised setting considered in Kipf & Welling (2017) while the dense setting is considered in Pei et al. (2019) for studying heterophilic graphs. For all datasets, we run each experiment 100 times with multiple random splits and different initializations.

Methods used for comparisons. We compare GPR-GNN with 6 baseline models: MLP, GCN (Kipf & Welling, 2017), GAT (Velickovic et al., 2018), JK-Net (Xu et al., 2018), GCN-Cheby (Defferrard et al., 2016), APPNP (Klicpera et al., 2018) and Geom-GCN (Pei et al., 2019). For all these architectures, we use the corresponding Pytorch Geometric library implementations (Fey & Lenssen, 2019). For Geom-GCN, we directly use the code provided by the authors[1]. We could not test Geom-GCN on cSBM and other datasets not originally tested in the paper due to a preprocessing subroutine that is not publicly available (Pei et al., 2019).

The GPR-GNN model setup and hyperparameter tuning. We choose random walk path lengths with $K = 10$ and use a 2-layer (MLP) with 64 hidden units for the NN component. For the GPR weights, we use different initializations including PPR with $\alpha \in \{0.1, 0.2, 0.5, 0.9\}$, $\gamma_k = \delta_{0k}$ or δ_{Kk} and the default random initialization in pytorch. Similarly, for APPNP we search the optimal α within $\{0.1, 0.2, 0.5, 0.9\}$. For other hyperparameter tuning, we optimize the learning rate over $\{0.002, 0.01, 0.05\}$ and weight decay $\{0.0, 0.0005\}$ for all models. For Geom-GCN, we use the best variants in the original paper for each dataset. Finally, we use GPR-GNN(rand) to describe the results obtained with random initialization of the GPR weights. Further experimental settings are discussed in the Supplement.

Results. We examine the robustness of all baseline methods and GPR-GNN using cSBM-generated data with $\phi \in \{-1, -0.75, -0.5, ..., 1\}$, which includes graphs across the heterophily/homophily spectrum. The results are summarized in Figure 2. For both the sparse and dense setting, GPR-GNN significantly outperforms all other baseline models whenever $\phi < 0$ (heterophilic graphs). On the other hand, all baseline GNNs can be worse then simple MLP when the graph information is weak ($\phi = 0, -0.25$). This shows that existing GNNs cannot apply to arbitrary graphs, while GPR-GNN is clearly more robust. The APPNP method has the worst performance on strongly heterophilic graphs. This confirms the result of our Theorem 4.1 since APPNP intrinsically has a low-pass filtering effect and is thus inadequate for the strong heterophily case. JKNet and GCN-Cheby are the only two baseline models that are able to learn well for the strong heterophily case when using dense splitting. This is also to be expected since JKNet is the only baseline model that combines results from different steps at the last layer, which is similar to what is done in GPR-GNN. GCN-Cheby uses multiple steps in each layers which allow it to partially adapt to heterophilic settings as each layer is related to a polynomial graph filter of higher order compared to that of GCN. Nevertheless, JKNet and GCN-Cheby are not deep in practice. Moreover, JKNet fails to learn well for sparse splitting while GCN-Cheby fails to learn well when the graph information is strong ($|\phi| \geq 0.5$) under sparse splitting.

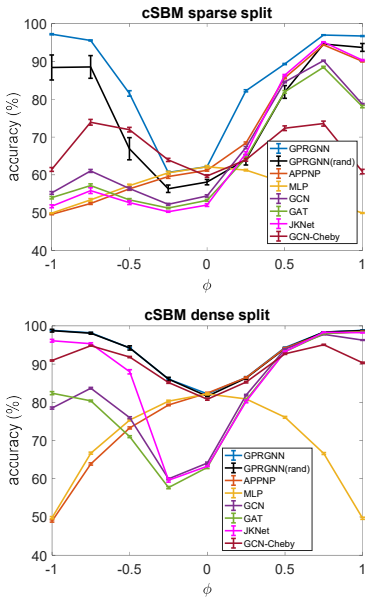
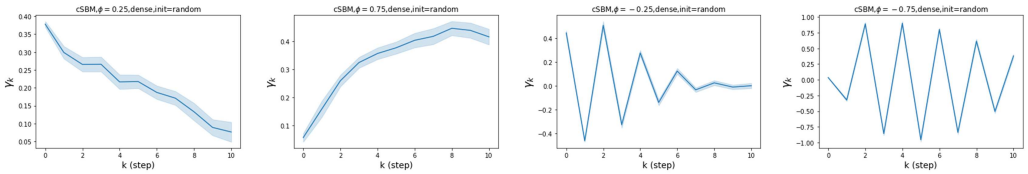


Figure 2: Accuracy of tested models on cSBM. Error bars indicate 95% confidence interval.

[1]https://github.com/graphdml-uiuc-jlu/geom-gcn



(a) $\phi = 0.25$, ($\mathcal{H}(G) = 0.189$) (b) $\phi = 0.75$, ($\mathcal{H}(G) = 0.688$) (c) $\phi = -0.25$, ($\mathcal{H}(G) = 0.029$) (d) $\phi = -0.75$, ($\mathcal{H}(G) = 0.002$)

Figure 3: Figure (a)-(d) shows the learnt GPR weights by GPR-GNN with random initialization on cSBM, dense split. The shaded region indicates 95% confidence interval.

Also, we observe that random initialization of our GPR weights only results in slight performance drops under dense splitting (The drop is more evident for sparse splitting setting but our method still outperforms baseline models by a large margin for strongly heterophilic graphs). This is also to be expected as we have less label information in the sparse splitting setting where the implicit bias provided by good GPR initialization is helpful. The implicit bias becomes irrelevant for the dense splitting setting, since the available label information is sufficiently rich.

Besides the strong performance of GPR-GNN, the other benefit is its interpretability. In Figure 3, we demonstrate the learnt GPR weights by our GPR-GNN on cSBM with random initialization. When the graph is weak homophilic ($\phi = 0.25$), the learnt GPR weights are decreasing. This is similar to the PPR weights used in APPNP, despite that the decaying speed is different. When the graph is strong homophilic ($\phi = 0.75$), the learnt GPR weights are increasing which is significantly different from the PPR weights. This result matches the recent finding in Li et al. (2019) and behave similar to IPR proposed by the authors. On the other hand, the learnt GPR weights have zig-zag shape when the graph is heterophilic. This again validates Theorem 4.1 as GPR weights with alternating signs correspond to a high-pass filter. Notably, when $\phi = -0.25$ the magnitude of learnt GPR weight is decreasing. This is because the graph information is weak and the node feature information is more important in this case. It makes sense that the learnt GPR weight focus on the first few steps. Hence, we have validated the interpretablity of GPR-GNN. In practice, one can use the learnt GPR weights to better understand the graph structured data at hand. We showcase this benefit in the results of real world benchmark datasets.

Real world benchmark datasets. We use 5 homophilic benchmark datasets available from the Pytorch Geometric library, including the citation graphs Cora, CiteSeer, PubMed (Sen et al., 2008; Yang et al., 2016) and the Amazon co-purchase graphs Computers and Photo (McAuley et al., 2015; Shchur et al., 2018). We also use 5 heterophilic benchmark datasets tested in Pei et al. (2019), including Wikipedia graphs Chameleon and Squirrel, the Actor co-occurrence graph, and webpage graphs Texas and Cornell from WebKB[2]. We summarize the dataset statistics in Table 1.

Table 1: Benchmark dataset properties and statistics.

Dataset	Cora	Citeseer	PubMed	Computers	Photo	Chameleon	Squirrel	Actor	Texas	Cornell
Classes	7	6	5	10	8	5	5	5	5	5
Features	1433	3703	500	767	745	2325	2089	932	1703	1703
Nodes	2708	3327	19717	13752	7650	2277	5201	7600	183	183
Edges	5278	4552	44324	245861	119081	31371	198353	26659	279	277
$\mathcal{H}(G)$	0.656	0.578	0.644	0.272	0.459	0.024	0.055	0.008	0.016	0.137

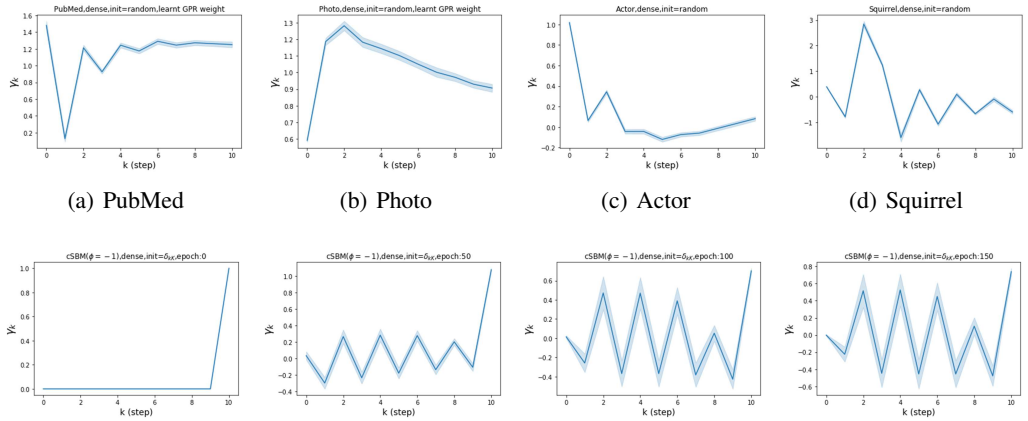
Results on real-world datasets. We use accuracy (the micro-F1 score) as the evaluation metric along with a 95% confidence interval. The relevant results are summarized in Table 2. For homophilic datasets, we provide results for sparse splitting which is more aligned with the original setting used in Kipf & Welling (2017); Shchur et al. (2018). For the heterophilic datasets, we adopt dense splitting which is used in Pei et al. (2019).

Table 2 shows that, in general, GPR-GNN outperforms all tested methods. On homophilic datasets, GPR-GNN achieves the state-of-the-art performance. On heterophilic datasets, GPR-GNN significantly outperforms all the other baseline models. For fairness, we also repeated the experiment

[2]http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/wwkb

Table 2: Results on real world benchmark datasets: Mean accuracy (%) $\pm$ 95% confidence interval. Boldface letters are used to mark the best results while underlined boldface letters indicate results within the given confidence interval of the best result.

	Cora	Citeseer	PubMed	Computers	Photo	Chameleon	Actor	Squirrel	Texas	Cornell
GPRGNN	**79.51±0.36**	67.63±0.38	**85.07±0.09**	**82.90±0.37**	**91.93±0.26**	**67.48±0.40**	**39.30±0.27**	**49.93±0.53**	**92.92±0.61**	**91.36±0.70**
APPNP	**79.41±0.38**	**68.59±0.30**	**85.02±0.09**	81.99±0.26	91.11±0.26	51.91±0.56	38.86±0.24	34.77±0.34	91.18±0.70	**91.80±0.63**
MLP	50.34±0.48	52.88±0.51	80.57±0.12	70.48±0.28	78.69±0.30	46.72±0.46	38.58±0.25	31.28±0.27	92.26±0.71	**91.36±0.70**
GCN	75.21±0.38	67.30±0.35	84.27±0.01	82.52±0.32	90.54±0.21	60.96±0.78	30.59±0.23	45.66±0.39	75.16±0.96	66.72±1.37
GAT	76.70±0.42	67.20±0.46	83.28±0.12	81.95±0.38	90.09±0.27	63.9±0.46	35.98±0.23	42.72±0.33	78.87±0.86	76.00±1.01
JKNet	73.22±0.64	60.85±0.76	82.91±0.11	77.80±0.97	87.70±0.70	62.92±0.49	33.41±0.25	44.72±0.48	75.53±1.16	66.73±1.73
GCN-Cheby	71.39±0.51	65.67±0.38	83.83±0.12	82.41±0.28	90.09±0.28	59.96±0.51	38.02±0.23	40.67±0.31	86.08±0.96	85.33±1.04
GeomGCN	20.37±1.13	20.30±0.90	58.20±1.23	NA	NA	61.06±0.49	31.81±0.24	38.28±0.27	58.56±1.77	55.59±1.59



(a) PubMed (b) Photo (c) Actor (d) Squirrel

(e) Epoch= 0 (f) Epoch= 50 (g) Epoch= 100 (h) Epoch= 150

Figure 4: Figures (a)-(d) show the learned GPR weights of our GPR-GNN method with random initialization on various datasets, for dense splitting. Figures (e)-(f) show the learned weights of our GPR-GNN method with initialization δ_{kK} on cSBM($\phi = -1$), for dense splitting. The shaded region indicates a 95% confidence interval.

involving GeomGCN on homophilic datasets using a dense split - the observed performance pattern tends to be similar which can be found in Supplement.

We also examined the learned GPR weights on real datasets in Figure 4. Due to space limitations, a more comprehensive GPR weight analysis for other datasets is deferred to the Supplement. We can see that learned GPR weights are all positive for homophilic datasets (PubMed and Photo). In contrast, some GPR weights learned from heterophilic datasets (Actor and Squirrel) are negative. These results agree with the patterns observed on cSBMs. Interestingly, the learned weight γ_0 has the largest magnitude for the Actor dataset. This indicates that most of the information is contained in node features. From Table 2 we can also see that MLPs indeed outperforms most baseline GNNs (this is similar to the case of cSBM($\phi = -0.25$)). On the other hand, GPR weights learned from Squirrel have a zig-zag pattern. This implies that graph topology is more informative for Squirrel compared to Actor. From Table 2 we also see that baseline GNNs also outperform MLPs on Squirrel.

Escaping from over-smoothing and dynamics of learning GPR weights. To demonstrate the ability of GPR-GNNs to escape from over-smoothing, we choose the initial GPR weights to be $\gamma_k = \delta_{kK}$. This ensures that over-smoothing effects are present with high probability at the very beginning of the learning process. On cSBM($\phi = -1$) with dense splitting, we find that for 96 out of 100 runs, GPR-GNN predicts the same labels for all nodes at epoch 0, which implies that over-smoothing indeed occurs immediately. The final prediction is 98.79% accurate which is much larger than the initial accuracy of 50.07% at epoch 0. Similar results can be observed for other datasets and this verifies our theoretical findings. We plot the dynamics of the learned GPR weights in Figure 4(e)-(h), which shows that the peak at last step is indeed reduced while the GPR weights for other steps are significantly increased in magnitude. More results on the dynamics of learning GPR weights may be found in the Supplement.

REFERENCES

Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alexander A Alemi. Watch your step: Learning node embeddings via graph attention. In *Advances in Neural Information Processing Systems*, pp. 9180–9190, 2018.

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International Conference on Machine Learning*, pp. 21–29, 2019.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Dimitris Berberidis, Athanasios Nikolakopoulos, and Georgios B Giannakis. Adaptive diffusions for scalable learning over graphs. In *Mining and Learning with Graphs Workshop @ ACM KDD 2018*, pp. 1, 8 2018.

Aleksandar Bojchevski, Johannes Klicpera, Bryan Perozzi, Martin Blais, Amol Kapoor, Michal Lukasik, and Stephan Günnemann. Is pagerank all you need for scalable graph neural networks? In *ACM KDD, MLG Workshop*, 2019.

Aleksandar Bojchevski, Johannes Klicpera, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann. Scaling graph neural networks with approximate pagerank. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2464–2473, 2020.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, pp. http–openreview, 2014.

Hoang Dau and Olgica Milenkovic. Latent network features and overlapping community discovery via boolean intersection representations. *IEEE/ACM Transactions on Networking*, 25(5):3219–3234, 2017.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pp. 3844–3852, 2016.

Yash Deshpande, Subhabrata Sen, Andrea Montanari, and Elchanan Mossel. Contextual stochastic block models. In *Advances in Neural Information Processing Systems*, pp. 8581–8593, 2018.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1263–1272. JMLR. org, 2017.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pp. 1024–1034, 2017.

Glen Jeh and Jennifer Widom. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*, pp. 271–279, 2003.

Junteng Jia and Austin Benson. Outcome correlation in graph neural network regression. *arXiv preprint arXiv:2002.08274*, 2020.

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. In *International Conference on Learning Representations (ICLR)*, 2017.

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations*, 2018.

Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. In *Advances in Neural Information Processing Systems*, pp. 13333–13345, 2019.

Isabel M Kloumann, Johan Ugander, and Jon Kleinberg. Block models and personalized pagerank. *Proceedings of the National Academy of Sciences*, 114(1):33–38, 2017.

Pan Li, I Chien, and Olgica Milenkovic. Optimizing generalized pagerank methods for seed-expansion community detection. In *Advances in Neural Information Processing Systems*, pp. 11705–11716, 2019.

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.

Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170, 2011.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 43–52, 2015.

Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 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. *arXiv preprint arXiv:1811.05868*, 2018.

Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(77):2539–2561, 2011.

D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

Charalampos Tsourakakis. Provably fast inference of latent features from networks: With applications to learning social circles and multilabel classification. In *Proceedings of the 24th International Conference on World Wide Web*, pp. 1111–1121, 2015.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lia, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=rJXMpikCZ`.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

WOK Asiri Suranga Wijesinghe and Qing Wang. Dfnets: Spectral cnns for graphs with feedback-looped filters. In *Advances in Neural Information Processing Systems*, pp. 6007–6018, 2019.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International Conference on Machine Learning*, pp. 6861–6871, 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, 2018.

Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, pp. 40–48, 2016.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graph-SAINT: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=BJe8pkHFwS`.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Generalizing graph neural networks beyond homophily. *arXiv preprint arXiv:2006.11468*, 2020.

Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.