A Terminology Explanation

Denote $\mathcal{F}: \mathbb{R}^n \to \mathbb{C}^n$ be the Discrete Fourier Transform. Applying $\mathcal{F}$ to a sequence of nodes $\mathbf{H}$ is equivalent to left-multiplying by a DFT matrix, where the rows are the Fourier basis vectors $\boldsymbol{f}_k = \left[e^{2\pi \mathrm{j}(k-1)\cdot 0} \quad \cdots \quad e^{2\pi \mathrm{j}(k-1)\cdot(n-1)}\right]^T / \sqrt{n} \in \mathbb{R}^n$, where k represents the k-th row of the DFT matrix, and j is the imaginary unit. Let $\tilde{\boldsymbol{z}} = \mathcal{F}\boldsymbol{z}$ represent the spectrum of $\boldsymbol{z}$. We can split it into two parts, as $\tilde{\boldsymbol{z}}_{dc} \in \mathbb{C}$ and $\tilde{\boldsymbol{z}}_{hc} \in \mathbb{C}^{n-1}$ take the first element and the rest elements of $\tilde{\boldsymbol{z}}$, respectively. Formally, we can define $\mathcal{DC}[\boldsymbol{z}] = \tilde{\boldsymbol{z}}_{dc}\boldsymbol{f}_1 \in \mathbb{C}^n$ as the Direct-Current (DC) component of input $\boldsymbol{z}$, and $\mathcal{HC}[\boldsymbol{z}] = \left[\boldsymbol{f}_2 \quad \cdots \quad \boldsymbol{f}_n\right]\tilde{\boldsymbol{z}}_{hc} \in \mathbb{C}^n$ as the complementary high-frequency component (HC).

B Theoretical Proofs

B.1 Proof of Theorem 1

Theorem 1. *Given $\gamma_1 > 0, \gamma_k = (-a)^k/2, a \in (0, \frac{1}{n}), k > 1$, the attention matrix $\hat{\boldsymbol{A}}$ in ParaFormer is a polynomial graph filter can function as both a low-pass and high-pass graph filter.*

Proof. As defined in prior work [51], the low frequency component in attention, e.g., direct component (DC), can be written as:

$$\mathcal{DC}[\mathbf{x}] = DFT^{-1}\mathrm{diag}(1, 0, \ldots, 0)DFT\mathbf{x} \tag{16}$$

$$= \frac{1}{n}\mathbf{1}\mathbf{1}^T\mathbf{x} \tag{17}$$

We first prove with $\gamma_k = (-a)^k/2, a \in (0, \frac{1}{n}), k > 1$, ParaFormer will only pass the high frequency information. Specifically, the low frequency information in ParaFormer can be represented as: $\mathcal{DC}\left[\sum_{k=0}^{K}\gamma_k\boldsymbol{A}^k\mathbf{H}\right] = \frac{1}{n}\mathbf{1}\mathbf{1}^T\sum_{k=0}^{K}\gamma_k\boldsymbol{A}^k\mathbf{H}$.

Define $c = \mathbf{1}^T\boldsymbol{A}$. Easily, we can have $\mathbf{1}^T\boldsymbol{A}^2 = c^{(2)}$, where $c^{(m)} = \{c_0^m, c_1^m, \ldots, c_n^m\}$. Due to the mathematical induction, we have: $\mathbf{1}^T\boldsymbol{A}^k = c^{(k)}$

$$\mathcal{DC}\left[\sum_{k=0}^{K}\gamma_k\boldsymbol{A}^k\right] = \frac{1}{n}\mathbf{1}\sum_{k=0}^{K}(-\alpha)^k\mathbf{1}^T\boldsymbol{A}^k \tag{18}$$

$$= \frac{1}{n}\mathbf{1}\sum_{k=0}^{K}(-\alpha)^k c^{(k)} \tag{19}$$

$$= \frac{1}{n}\mathbf{1}\sum_{k=0}^{K}\{(-\alpha c_1)^k, (-\alpha c_2)^k, \ldots, (-\alpha c_n)^k\} \tag{20}$$

$$= \mathbf{0} \tag{21}$$

As $\boldsymbol{A}$ is an attention matrix, $c_i < n$, where n is the number of nodes, then each value in column i of $\boldsymbol{A}$ is lower than 1, due to the Softmax. We only need to guarantee $\alpha < \frac{1}{n}$, then the Equation 20 will converge to 0. In this way, the low frequency information will be alleviated and only contain the high frequency information after the filter.

In another perspective, when $\{\gamma_0 = 1, \gamma_1 = 1$ and $\gamma_k = 0, k = 2, 3\ldots\}$, ParaFormer will be degraded as a vanilla Transformer, and has been proven as a low-pass filter by previous work [51].

Thus, by combining two parameters of $\{\gamma\}_{\text{high}}$ and $\{\gamma\}_{\text{low}}$ which can to function as low and high filter as $\frac{1}{2}(\{\gamma\}_{\text{high}}+\{\gamma\}_{\text{low}})$, ParaFormer can adaptively pass the low, high or both frequency information in the graph. □

B.2 Proof of Theorem 2

Theorem 2. *Give a Generalized PageRank Attention GPA and a self-attention module SA initialized with proper weight $\{\gamma_k\}$, we have: $\lambda_{GPA} \le \lambda_{SA}$.*

Proof. The proof can be directly derived from the definition of smoothing rate [51]. Specifically, we have:

$$\lambda_{\text{SA}} = \sqrt{\|\text{Softmax}(\mathbf{P})\|_1}\|\mathbf{W}_V\|_2 \tag{22}$$

$$\tag{23}$$

As $\text{Softmax}(\mathbf{P}) \in \mathbb{R}^{n\times n}$ is an attention matrix, $\sum_{i,j}\text{Softmax}(\mathbf{P})_{ij} = n$. As $\text{Softmax}(\mathbf{P})$ has n columns, we have $\|\text{Softmax}(\mathbf{P})\|_1 > 1$ if not all embeddings are totally same. Therefore, we denote $c = \|\text{Softmax}(\mathbf{P})\| > 1$.

We consider a simple initialization: $\{\gamma_0 = \frac{c-1}{2}, \gamma_1 = -\frac{1}{c}, \gamma_k = 0 | k = 2, 3, 4\cdots, K\}$. Given these conditions, we consider the smoothing rate of GPA:

$$\lambda_{\text{GPA}} = \sqrt{||\sum_{k=0}^{K}\gamma_k * \text{Softmax}(\mathbf{P})^K||_1}\|\mathbf{W}_V\|_2 \tag{24}$$

$$= \sqrt{||\frac{c-1}{2}\boldsymbol{I} - \frac{1}{c}\text{Softmax}(\mathbf{P})||_1}\|\mathbf{W}_V\|_2 \tag{25}$$

At column j, the sum of absolute value in GPA will be:

$$\sum_{i=1}^{n}\left|\frac{c-1}{2}\delta_{ij} - \frac{1}{c}p_{ij}\right|, \tag{26}$$

where δ_{ij} is the element in identity matrix $\boldsymbol{I}$.

$$\sum_{i=1}^{m}\left|\frac{c-1}{2}\delta_{ij} - \frac{1}{c}p_{ij}\right| \tag{27}$$

$$\le \frac{c-1}{2} + \frac{1}{c}\sum_{i=1}^{m}|p_{ij}| \tag{28}$$

$$\le \frac{c-1}{2} + \frac{1}{c}\times c \tag{29}$$

$$= \frac{c+1}{2} < c. \tag{30}$$

With conclusion in Equation 30, we have:

$$\lambda_{\text{GPA}} = \sqrt{||\sum_{k=0}^{K}\gamma_k * \text{Softmax}(\mathbf{P})^K||_1}\|\mathbf{W}_V\|_2 \tag{31}$$

$$= \max_j\left(\sum_{i=1}^{m}\left|\frac{c-1}{2}\delta_{ij} - \frac{1}{c}p_{ij}\right|\right)\|\mathbf{W}_V\|_2 \tag{32}$$

$$< c\|\mathbf{W}_V\|_2 = \|\text{Softmax}(\mathbf{P})\|\|\mathbf{W}_V\|_2 \tag{33}$$

$$= \lambda_{\text{SA}} \tag{34}$$

In conclusion, we have $\lambda_{\text{GPA}} < \lambda_{\text{SA}}$ with such initialization. □

B.3 Proof of Theorem 3

Theorem 3. *Suppose K is sufficiently large, for PageRank-enhanced attention, $\hat{\mathbf{A}}^k\mathbf{V}, \forall k \geq k'$, will be over-smoothed. When over-smoothing happens in ParaFormer, the learnable γ_k will converge to 0 with appropriate learning rate.*

Proof: First, we prove the first half of this proposition. Specifically, we aim to demonstrate that for sufficiently large k', the expression $\hat{\mathbf{A}}^k\mathbf{V}, \forall k \geq k'$, will be over-smoothed. This phenomenon can be interpreted through random walk theory. The attention matrix $\hat{\mathbf{A}}$ is normalized via Softmax, thereby allowing us to interpret $\hat{\mathbf{A}}$ as the probability transition matrix of a Markov chain. Each entry within this matrix is strictly positive due to the Softmax normalization, which implies the Markov chain is irreducible. Irreducibility ensures that the Markov chain converges to a unique stationary distribution denoted by $\boldsymbol{\pi}$. Let $\boldsymbol{a}_i$ represent the i-th row of $\boldsymbol{A}$. It follows that for all $i = 1, \cdots, n$ we have $\|(\boldsymbol{a}_i)^T\boldsymbol{A} - \boldsymbol{\pi}^T\| \leq \lambda\|\boldsymbol{a}_i - \boldsymbol{\pi}\|$, where $\lambda \in (0, 1)$ is the mixing rate of the transition matrix $\boldsymbol{A}$. Consequently, we arrive at the conclusion that, $\lim_{l\to\infty} \boldsymbol{A}^l = \mathbf{1}\boldsymbol{\pi}^T$, which yields a low-pass filter. In the case of ParaFormer, when k is sufficiently large, $\mathbf{H}_k = \hat{\mathbf{A}}^k\mathbf{V} = \mathbf{1}\boldsymbol{\pi}^T\mathbf{V}$. The rank of $\mathbf{H}_k$ will collapse to 1, and all nodes will have the same representation, resulting over-smoothing phenomenon.

We will now proceed to demonstrate the subsequent component of our analysis. As established in the preceding paragraph, with sufficiently large $k', \forall k \geq k'$, the representations $\mathbf{H}_k$ exhibit over-smoothing. In this section, we aim to show that within the framework of gradient descent, the representations $\mathbf{H}_k$ will have minimal influence on the final learned representation $\mathbf{H}$ unless the over-smoothed representations align with the final predictions.

Formally, the label matrix is denoted as $\mathbf{Y} \in \mathbb{R}^{n\times C}$, where each row of $\mathbf{Y}$ represents a one-hot vector. The predicted probability matrix is denoted as $\hat{\mathbf{P}} \in \mathbb{R}^{n\times C}$. Accordingly, the cross-entropy loss can be defined as follows:

$$L = -\sum_{i\in V} \mathbf{Y}_{i:} \log(\hat{\mathbf{P}}_{i:}) \tag{35}$$

For those smoothed representations, *e.g.*, when k is sufficiently large, we have the following limit: $\lim_{l\to\infty} \mathbf{H}_k = \hat{\mathbf{A}}^k\mathbf{V} = \mathbf{1}\boldsymbol{\pi}^T\mathbf{H}_0$. This can also be expressed as: $\mathbf{H}_k = \mathbf{1}\boldsymbol{\pi}^T\mathbf{H}_0 + o_k(1)$. It follows that we can derive the prediction matrix as: $\hat{\mathbf{P}} = \text{Softmax}_\eta(\mathbf{H})$, where $\mathbf{H} = \sum_{k=0}^{K} \gamma_k\mathbf{H}_k$. The function Softmax_η incorporates an additional smoothing parameter η. when $\eta = 1$, Softmax_η reduces to the standard Softmax function. We denote $\boldsymbol{\beta}^T = \boldsymbol{\pi}^T\mathbf{H}_0$. Consequently, we can compute the gradient with respect to γ_k:

$$\begin{aligned}\frac{\partial L}{\partial \gamma_k} &= \sum_{i\in\mathcal{T}} \eta(\hat{\mathbf{P}}_{i:} - \mathbf{Y}_{i:})\mathbf{H}_{k,i:} \\ &= \sum_{i\in\mathcal{T}} \eta(\hat{\mathbf{P}}_{i:} - \mathbf{Y}_{i:})\boldsymbol{\beta}^T + o_k(1)\end{aligned} \tag{36}$$

Here, we also introduce a lemma and the definition defined in GPRGNN [8].

Lemma 1. *For any real vector $\boldsymbol{\beta} \in \mathbb{R}^C$ and and sufficiently large $\eta > 0$, the following holds: $\text{softmax}_\eta(\boldsymbol{\beta}) = \mathbf{1}[\boldsymbol{\beta}] + o_\eta(1)$, where $\mathbf{1}[\boldsymbol{\beta}] \in \mathbb{R}^C$ is defined such that $\mathbf{1}[\boldsymbol{\beta}]_{argmax(\boldsymbol{\beta})} = 1$, and takes the value 0 at all other indices.*

Definition 2 (The over-smoothing phenomenon). *To enhance clarity in the notation, in the subsequent proof we introduce a new variable to denote the final representation: $\mathbf{Z} = \sum_k \gamma_k\mathbf{H}^{(k)}$. If over-smoothing occurs in the ParaFormer for K sufficiently large, we have $\mathbf{Z} = c_0\mathbf{1}\boldsymbol{\beta}^T$, $\forall j \in [C]$ for some $c_0 > 0$ if $\gamma_k > 0$ and $\mathbf{Z} = -c_0\mathbf{1}\boldsymbol{\beta}^T$, $\forall j \in [C]$ for some $c_0 > 0$ if $\gamma_k < 0$.*

Given the definition and of over-smoothing, the gradient can be reformulated as follows:

$$\begin{aligned}\frac{\partial L}{\partial \gamma_k} &= \sum_{i\in\mathcal{T}} \eta(\hat{\mathbf{P}}_{i:} - \mathbf{Y}_{i:})\boldsymbol{\beta}^T + o_k(1) \\ &= \sum_{i\in\mathcal{T}} \eta\Big(\frac{e^{\eta\mathbf{Z}_{i:}}}{\sum_{j\in[C]} e^{\eta\mathbf{Z}_{ij}}} - \mathbf{Y}_{i:}\Big)\boldsymbol{\beta}^T + o_k(1)\end{aligned} \tag{37}$$

From Definition 2, we first discuss the situation of $\gamma_k > 0$. We can simplify the Softmax term via 1 and calculate the gradient as:

$$\begin{aligned}\frac{\partial L}{\partial \gamma_k} &= \sum_{i\in\mathcal{T}} \eta\Big(\frac{e^{\eta c_0\boldsymbol{\beta}^T}}{\sum_{j\in[C]} e^{\eta\mathbf{Z}_{ij}}} - \mathbf{Y}_{i:}\Big)\boldsymbol{\beta}^T + o_k(1) \\ &= \sum_{i\in\mathcal{T}} \eta(\mathbf{1}[c_0\boldsymbol{\beta}^T] - \mathbf{Y}_{i:})\boldsymbol{\beta}^T + o_k(1) + o_\eta(1) \\ &= \sum_{i\in\mathcal{T}} \eta\Big(\max_{j\in[C]} \boldsymbol{\beta}_j^T - \boldsymbol{\beta}_{\mathbf{1}[\mathbf{Y}_{i:}]}^T\Big) + o_k(1) + o_\eta(1)\end{aligned} \tag{38}$$

Similarly, we can have when $\gamma_k < 0$, the gradient can be reformulated as:

$$\begin{aligned}\frac{\partial L}{\partial \gamma_k} &= \sum_{i\in\mathcal{T}} \eta\Big(\frac{e^{\eta c_0\boldsymbol{\beta}^T}}{\sum_{j\in[C]} e^{\eta\mathbf{Z}_{ij}}} - \mathbf{Y}_{i:}\Big)\boldsymbol{\beta}^T + o_k(1) \\ &= \sum_{i\in\mathcal{T}} \eta(\mathbf{1}[c_0\boldsymbol{\beta}^T] - \mathbf{Y}_{i:})\boldsymbol{\beta}^T + o_k(1) + o_\eta(1) \\ &= \sum_{i\in\mathcal{T}} \eta\Big(\min_{j\in[C]} \boldsymbol{\beta}_j^T - \boldsymbol{\beta}_{\mathbf{1}[\mathbf{Y}_{i:}]}^T\Big) + o_k(1) + o_\eta(1)\end{aligned} \tag{39}$$

In conclusion, when $\gamma_k > 0$, $\frac{\partial L}{\partial \gamma_k}$ will also be positive, disregarding the $O(1)$, and vice versa. In the framework of gradient descent optimization, the γ_k will decrease if it is greater than 0 and increase if it is less than 0. With an appropriately chosen learning rate, γ_k can can converge towards 0 to alleviate the over-smoothing effect.

C Details of Datasets

In this section, we will introduce the detailed information of these datasets.

Cora, Citeseer and Pubmed [45] represent scientific publications as nodes, with citations between them forming edges. Node features consist of bag-of-words representations of the publication's abstract, while node labels denote the publication's subject category. Cora comprises 2,708 publications in seven computer science subfields, CiteSeer encompasses 3,327 publications across six computer science categories, and PubMed includes 19,717 publications from the PubMed database, classified into three medical categories. We employ the full-supervised settings following [19], which employs a random 60%/20%/20% train/valid/test split.

Film [48] dataset encompasses a network of 7,600 actors (nodes) connected by 29,926 edges representing shared appearances on Wikipedia pages. Each actor's node is characterized by keywords

extracted from their corresponding Wikipedia entry, serving as features. The dataset aims to categorize these actors into five distinct classes based on the content of their Wikipedia profiles. Notably, this graph exhibits low homophily, indicating a tendency for actors within the same category to be less interconnected compared to real-world social networks.

The Squirrel and Chameleon datasets [41], derived from Wikipedia, represent page-page networks centered around specific topics. Nodes correspond to individual pages, interconnected by edges indicating mutual links. Each node's features comprise a set of informative nouns extracted from the page content. The task associated with these datasets involves classifying nodes into five categories based on their average monthly traffic. Notably, Squirrel and Chameleon exhibit significant label heterophily, meaning linked nodes often belong to different traffic categories. This inherent characteristic makes accurate classification more complex, demanding models capable of discerning subtle patterns within the network structure and node attributes. A recent work [34] demonstrates there are overlapping nodes between training and testing in original Squirrel and Chameleon datasets, which will result data leakage. Furthermore, this work solve this problem and propose new splits. In our work, we thus follow the updated splits.

The Deezer-Europe [42] dataset, derived from the Deezer music streaming platform, presents a user-user friendship network encompassing European users. Nodes in this network symbolize individual users, with edges denoting reciprocal friendships. User preferences, specifically the artists they have 'liked', constitute the node features. This dataset poses the challenge of gender prediction, aiming to classify users based on their musical tastes and connections. Following established benchmarks [29], the dataset is partitioned into a 50%/25%/25% split for training, validation, and testing, respectively.

The ogbn-arxiv [20] dataset presents a comprehensive citation network of Computer Science (CS) research papers on arXiv. This network models each paper as a node, with edges representing citations between them. Each node is characterized by a 128-dimensional feature, derived by averaging the word embeddings of the paper's title and abstract. These embeddings are generated using the WORD2VEC model. The dataset's primary task is classifying papers into one of 40 specific CS subject areas, enabling the exploration of topical relationships within the field. Following the established split [20], we train models on papers published before 2017, validate on those from 2018, and test on papers published from 2019 onwards.

The Amazon2M [30] dataset, constructed from the Amazon co-purchasing network, models product relationships as a graph. Nodes in this graph symbolize individual products, while edges signify frequent co-purchasing patterns. Each product node is characterized by a bag-of-words representation derived from its textual description. Product categorization, specifically the top-level category a product belongs to, serves as the node label. Following established practices [54], we employed a 50%/25%/25% random split for training, validation, and testing.

The Pokec [25] dataset offers a rich snapshot of social network interactions, comprising detailed user profiles with attributes such as geographic location, age, date of registration, and declared interests.

Table 9: Frequently Used Notations

Notation	Description
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	a graph
$\mathbf{A}$	the adjacency matrix
λ	the smoothing rate
γ_k	the weights of GPR
K	the number of layers
$\mathbf{Q}, \mathbf{K}, \mathbf{V}$	the query, key, value matrices
$\hat{\mathbf{A}}$	the attention matrix
$\mathbf{H}/\mathbf{X}$	the token/node features

This dataset leverages these features to predict user gender, providing a challenging classification task. Following previous work [55], we randomly partition the dataset into training (10%), validation (10%), and testing (80%) sets.

The arXiv-year [29] dataset, derived from the ogbn-arxiv network, offers a unique perspective on scientific publications by employing publication year as the target label instead of subject areas. This dataset comprises arXiv papers as nodes, with directed edges indicating citation relationships. Each node's features are represented by averaged word2vec embeddings of the paper's title and abstract, capturing semantic information. The dataset is carefully partitioned into five classes based on publication year ranges (pre-2014, 2014-2015, 2016-2017, 2018, and 2019-2020) to ensure balanced class distribution. For the split, we employed a 50%/25%/25% random partition for training, validation, and testing.

D Implementation Details

In this section, we will provide more detailed information about the implementation of ParaFormer for reproducibility.

D.1 Notation & Algorithm

Table 9 provides the frequently-used notations throughout the paper. Algorithm 1 presents the forward pass algorithm of computing the scalability GPA. Here, K attention blocks share the parameter matrices $\mathbf{W}_Q$, $\mathbf{W}_K$ $\mathbf{W}_V$. The matrix $\mathbf{M}$ caches the intermediate result of $(\hat{\mathbf{K}}^T\hat{\mathbf{Q}})^{(k-1)}(\hat{\mathbf{K}}^T\mathbf{V})$.

D.2 ParaFormer

To ensure a fair comparison with SGFormer, the implementation of ParaFormer$_{\text{GCN}}$ maintains consistency in the GCN architecture employed for each dataset. This encompasses utilizing the same GCN layer, hidden dimension, and the weighting assigned to the graph representation's contribution to the final hidden dimension. For these detailed hyper-parameters, please refer to the original paper and repository of SGFormer.

In the implementation of ParaFormer$_{\text{GCN}}$, we fix the hyperparameter K as 10 across all datasets. The remaining hyperparameters are determined through a grid search strategy within the following search space:

- learning rate within $\{0.005, 0.01, 0.05, 0.1\}$.
- hidden dimension within $\{64, 96, 128, 256\}$.
- dropout rate within $\{0.3, 0.4, 0.5, 0.6, 0.7\}$.

In the implementation of ParaFormer$_{\text{GPRGNN}}$, we set the K as 10 both for ParaFormer and GPRGNN. In our implementation, instead of paralellizing the ParaFormer and GNN block, we turn to

another more natural approach to integrate the structural information. Specifically, we sum up the GPR enhanced attention matrix with GPR enhanced adjacency matrix as: $\mathbf{H} = \sum_{k=0}^{10}(\lambda_k \hat{\mathbf{A}}^k + \lambda_k^{'} \mathbf{A}^k)\mathbf{V}$. As other hyper-parameters, we search the hyper-parameters within the space:

- learning rate within $\{0.001, 0.005, 0.01, 0.05\}$.
- weight decay within $\{0.0, 1e-4, 5e-4, 1e-3\}$.
- hidden dimension within $\{64, 128, 256, 512\}$.
- dropout rate within $\{0.3, 0.5, 0.7, 0.8\}$.
- graph weight within $\{0.3, 0.5, 0.6, 0.7, 0.8, 0.9\}$.

Following the SGFormer, we adopt a full-batch training approach for the medium-sized graphs, ogbn-arxiv and arxiv-year. For the larger datasets, Amazon2M and pokec, a mini-batch training method is utilized. The batch size is set as 0.1M, which is the same as SGFormer. Furthermore, we also utilize the graph partitioning strategy employed in SGFormer.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009