APPENDIX

Anonymous authors
Paper under double-blind review

A. PROOFS OF THE DEFINITIONS

Definition 2 Vanilla GCNs. *Let $\bar{\boldsymbol{x}}(i)_{i\in\mathbb{V}}$ be the estimation of the input observation $\boldsymbol{x}(i)_{i\in\mathbb{V}}$. A low-pass filter:*

$$\bar{\boldsymbol{X}} = \tilde{\mathcal{A}}_{\mathrm{rw}}\boldsymbol{X}, \tag{1}$$

is the first-order approximation of *the optimal solution of the following optimization:*

$$\min_{\bar{\boldsymbol{X}}} \sum_{i\in\mathbb{V}} \|\bar{\boldsymbol{x}}(i) - \boldsymbol{x}(i)\|^2_{\tilde{\boldsymbol{D}}} + \sum_{i,j\in\mathbb{V}} a_{ij}\|\bar{\boldsymbol{x}}(i) - \bar{\boldsymbol{x}}(j)\|_2^2. \tag{2}$$

Proof. Let l denote the objective function. We have

$$l = \mathrm{tr}[(\bar{\boldsymbol{X}} - \boldsymbol{X})^T\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X})] + \mathrm{tr}(\bar{\boldsymbol{X}}^T\boldsymbol{L}\bar{\boldsymbol{X}}).$$

Then,

$$\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 2\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X}) + 2\boldsymbol{L}\bar{\boldsymbol{X}}.$$

If we let $\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 0$:

$$(\tilde{\boldsymbol{D}} + \boldsymbol{L})\bar{\boldsymbol{X}} = \tilde{\boldsymbol{D}}\boldsymbol{X}$$
$$(\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}})\bar{\boldsymbol{X}} = \boldsymbol{X}.$$

As the norm of eigenvalues of $\tilde{\mathcal{A}}_{\mathrm{rw}} = \boldsymbol{I} - \tilde{\mathcal{L}}_{\mathrm{rw}}$ is bounded by 1, $\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}}$ has eigenvalues in range [1, 3], which proves that $\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}}$ is a positive definite matrix. Therefore,

$$\bar{\boldsymbol{X}} = (\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}})^{-1}\boldsymbol{X}. \tag{3}$$

Unfortunately, solving the closed-form solution of Equation 3 is computationally expensive. Nevertheless, we can derive a simpler form, $\bar{\boldsymbol{X}} \approx (\boldsymbol{I} - \tilde{\mathcal{L}}_{\mathrm{rw}})\boldsymbol{X} = \tilde{\mathcal{A}}_{\mathrm{rw}}\boldsymbol{X}$, via first-order Taylor approximation which establishes the Definition. □

Definition 3 Residual Connection. *A graph convolution filter with residual connection:*

$$\bar{\boldsymbol{X}} = \tilde{\mathcal{A}}_{\mathrm{rw}}\boldsymbol{X} + \epsilon\boldsymbol{X}, \tag{4}$$

where $\epsilon > 0$ controls the strength of residual connection, ***is the first-order approximation of*** *the optimal solution of the following optimization:*

$$\min_{\bar{\boldsymbol{X}}} \sum_{i\in\mathbb{V}} (\|\bar{\boldsymbol{x}}(i) - \boldsymbol{x}(i)\|^2_{\tilde{\boldsymbol{D}}} - \epsilon\|\bar{\boldsymbol{x}}(i)\|^2_{\tilde{\boldsymbol{D}}}) + \sum_{i,j\in\mathbb{V}} a_{ij}\|\bar{\boldsymbol{x}}(i) - \bar{\boldsymbol{x}}(j)\|_2^2. \tag{5}$$

Proof. Let l denote the objective function. We have

$$l = \mathrm{tr}[(\bar{\boldsymbol{X}} - \boldsymbol{X})^T\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X})] - \epsilon\mathrm{tr}(\bar{\boldsymbol{X}}^T\tilde{\boldsymbol{D}}\bar{\boldsymbol{X}}) + \mathrm{tr}(\bar{\boldsymbol{X}}^T\boldsymbol{L}\bar{\boldsymbol{X}}).$$

Then,

$$\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 2\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X}) + 2(\boldsymbol{L} - \epsilon\tilde{\boldsymbol{D}})\bar{\boldsymbol{X}}.$$

If we let $\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 0$:

$$\begin{aligned}[(1-\epsilon)\tilde{\boldsymbol{D}} + \boldsymbol{L}]\bar{\boldsymbol{X}} &= \tilde{\boldsymbol{D}}\boldsymbol{X} \\ \bar{\boldsymbol{X}} &= [(1-\epsilon)\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}}]^{-1}\boldsymbol{X} \\ \bar{\boldsymbol{X}} &= [\boldsymbol{I} + (\tilde{\mathcal{L}}_{\mathrm{rw}} - \epsilon\boldsymbol{I})]^{-1}\boldsymbol{X}.\end{aligned}$$

Therefore, the first-order approximation of the optimal solution is

$$\begin{aligned}\bar{\boldsymbol{X}} &\approx [\boldsymbol{I} - (\tilde{\mathcal{L}}_{\mathrm{rw}} - \epsilon\boldsymbol{I})]\boldsymbol{X} \\ &= \tilde{\mathcal{A}}_{\mathrm{rw}}\boldsymbol{X} + \epsilon\boldsymbol{X}.\end{aligned}$$

□

Definition 3' Concatenation. *A graph convolution filter concatenating with the input signal:*

$$\bar{\boldsymbol{X}} = \tilde{\mathcal{A}}_{\mathrm{rw}}\boldsymbol{X} + \epsilon\boldsymbol{X}\Theta\Theta^T, \tag{6}$$

is the first-order approximation of *the optimal solution of the following optimization:*

$$\min_{\bar{\boldsymbol{X}}} \sum_{i\in\mathbb{V}} (\|\bar{\boldsymbol{x}}(i) - \boldsymbol{x}(i)\|^2_{\tilde{\boldsymbol{D}}} - \epsilon\|\bar{\boldsymbol{x}}(i)\Theta\|^2_{\tilde{\boldsymbol{D}}}) + \sum_{i,j\in\mathbb{V}} a_{ij}\|\bar{\boldsymbol{x}}(i) - \bar{\boldsymbol{x}}(j)\|_2^2, \tag{7}$$

where $\epsilon > 0$ controls the strength of concatenation and Θ is the learning coefficients for the concatenated signal.

Proof. Let l denote the objective function. We have

$$l = \mathrm{tr}[(\bar{\boldsymbol{X}} - \boldsymbol{X})^T\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X})] - \epsilon\mathrm{tr}((\bar{\boldsymbol{X}}\Theta)^T\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}}\Theta)) + \mathrm{tr}(\bar{\boldsymbol{X}}^T\boldsymbol{L}\bar{\boldsymbol{X}}).$$

Then,

$$\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 2\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X}) + 2\boldsymbol{L}\bar{\boldsymbol{X}} - 2\epsilon\tilde{\boldsymbol{D}}\bar{\boldsymbol{X}}\Theta\Theta^T.$$

If we let $\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 0$:

$$\begin{aligned}(\tilde{\boldsymbol{D}} + \boldsymbol{L})\bar{\boldsymbol{X}} - \epsilon\tilde{\boldsymbol{D}}\bar{\boldsymbol{X}}\Theta\Theta^T &= \tilde{\boldsymbol{D}}\boldsymbol{X} \\ (\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}})\bar{\boldsymbol{X}} - \epsilon\bar{\boldsymbol{X}}\Theta\Theta^T &= \boldsymbol{X}.\end{aligned}$$

With the help of the Kronecker product operator $\otimes$ and first-order Taylor expansion, we have

$$\begin{aligned}\mathrm{vec}(\bar{\boldsymbol{X}}) &= [(\boldsymbol{I}\otimes(\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}})) - \epsilon((\Theta\Theta^T)\otimes\boldsymbol{I})]^{-1}\mathrm{vec}(\boldsymbol{X}) \\ &\approx [2\boldsymbol{I} - (\boldsymbol{I}\otimes(\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}})) + \epsilon((\Theta\Theta^T)\otimes\boldsymbol{I})]\mathrm{vec}(\boldsymbol{X}) \\ &= \mathrm{vec}(2\boldsymbol{X} - (\boldsymbol{I} + \tilde{\mathcal{L}}_{\mathrm{rw}})\boldsymbol{X} + \epsilon\bar{\boldsymbol{X}}\Theta\Theta^T) \\ &= \mathrm{vec}(\tilde{A}_{rw}\boldsymbol{X} + \epsilon\boldsymbol{X}\Theta\Theta^T).\end{aligned}$$

□

Definition 4 Attention-based GCNs. *An attention-based graph convolution filter:*

$$\bar{\boldsymbol{X}} = \boldsymbol{P}\boldsymbol{X}, \tag{8}$$

is the first-order approximation of *the optimal solution of the following optimization:*

$$\min_{\bar{\boldsymbol{X}}} \sum_{i\in\mathbb{V}} \|\bar{\boldsymbol{x}}(i) - \boldsymbol{x}(i)\|^2_{\tilde{\boldsymbol{D}}} + \sum_{i,j\in\mathbb{V}} p_{ij}\|\bar{\boldsymbol{x}}(i) - \bar{\boldsymbol{x}}(j)\|_2^2, \qquad s.t. \sum_{j\in\mathbb{V}} p_{ij} = \tilde{\boldsymbol{D}}_{ii}, \forall i \in \mathbb{V}. \tag{9}$$

Proof. Let l denote the objective function. We have

$$l = \mathrm{tr}[(\bar{\boldsymbol{X}} - \boldsymbol{X})^T \tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X})] + \mathrm{tr}(\bar{\boldsymbol{X}}^T \boldsymbol{L}\bar{\boldsymbol{X}}).$$

Then,

$$\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 2\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X}) + 2(\tilde{\boldsymbol{D}} - \tilde{\boldsymbol{D}}\boldsymbol{P})\bar{\boldsymbol{X}}.$$

If we let $\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 0$:

$$\begin{aligned}(2\tilde{\boldsymbol{D}} - \tilde{\boldsymbol{D}}\boldsymbol{P})\bar{\boldsymbol{X}} &= \tilde{\boldsymbol{D}}\boldsymbol{X} \\ (2\boldsymbol{I} - \boldsymbol{P})\bar{\boldsymbol{X}} &= \boldsymbol{X}.\end{aligned}$$

Similarly, we can prove that $(2\boldsymbol{I} - \boldsymbol{P})$ is a positive definite matrix, with eigenvalues in range [1, 3]. Therefore,

$$\begin{aligned}\bar{\boldsymbol{X}} &= (2\boldsymbol{I} - \boldsymbol{P})^{-1}\boldsymbol{X} \\ &\approx \boldsymbol{P}\boldsymbol{X}.\end{aligned}$$

□

Definition 5 & 6 Topology-based GCNs Due to the fact that most of the topology-based models adopt non-convolutional operations like concatenation, we derive a more general objective function by combining with the non-convolutional operations:

$$\min_{\bar{\boldsymbol{X}}} \ \alpha_0 \sum_{i\in\mathbb{V}} \|\bar{\boldsymbol{x}}(i) - \boldsymbol{x}(i)\|^2_{\tilde{\boldsymbol{D}}} + \sum_{k=1}^{t} \alpha_k \sum_{i,j\in\mathbb{V}} a_{ij}^{(k)} \|\bar{\boldsymbol{x}}(i)\Theta^{(k)} - \bar{\boldsymbol{x}}(j)\Theta^{(k)}\|_2^2, \tag{10}$$

where $\sum_{k=0}^{t} \alpha_k = 1$, $\alpha_0 > 0$ and $\alpha_k \geq 0, k = 1, 2, \ldots, t$. If we let d be the feature dimension of $\boldsymbol{X}$, $\Theta^{(k)} \in \mathbb{R}^{d\times d}$ correspond to the learning weights for the k_{th} hop neighborhood. Let l denote the objective function, we have:

$$\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = \alpha_0\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X}) + \sum_{k=1}^{t} \alpha_k(\tilde{\boldsymbol{D}} - \tilde{\boldsymbol{D}}\tilde{\mathcal{A}}_{rw}^k)\bar{\boldsymbol{X}}\Theta^{(k)}(\Theta^{(k)})^T.$$

By letting $\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 0$, we have:

$$\alpha_0\bar{\boldsymbol{X}} + \sum_{k=1}^{t}(\boldsymbol{I}_n - \tilde{\mathcal{A}}_{\mathrm{rw}}^k)\bar{\boldsymbol{X}}\Theta^{(k)}(\Theta^{(k)})^T = \alpha_0\boldsymbol{X}.$$

Therefore, with the help of the Kronecker product operator $\otimes$ and first-order Taylor expansion, we have

$$[\alpha_0\boldsymbol{I}_n + \sum_{k=1}^{t}(\alpha_k\Theta^{(k)}(\Theta^{(k)})^T) \otimes (\boldsymbol{I}_n - \tilde{\mathcal{A}}_{\mathrm{rw}}^k)]\mathrm{vec}(\bar{\boldsymbol{X}}) = \alpha_0\mathrm{vec}(\boldsymbol{X}). \tag{11}$$

We can observe that $\sum_{k=1}^{t}(\alpha_k\Theta^{(k)}(\Theta^{(k)})^T)$ and $(\boldsymbol{I}_n - \tilde{\mathcal{A}}_{\mathrm{rw}}^k)$ have non-negative eigenvalues. Due to the property of the Kronecker product that the eigenvalues of the Kronecker product $(\boldsymbol{A} \otimes \boldsymbol{B})$ equal to the product of eigenvalues of $\boldsymbol{A}$ and $\boldsymbol{B}$, the filter $(\alpha_0\boldsymbol{I}_n + \sum_{k=1}^{t}(\alpha_k\Theta^{(k)}(\Theta^{(k)})^T)$ is proved to be a positive definite matrix. Therefore,

$$\begin{aligned}\mathrm{vec}(\bar{\boldsymbol{X}}) &= \alpha_0[\alpha_0\boldsymbol{I}_n + \sum_{k=1}^{t}(\alpha_k\Theta^{(k)}(\Theta^{(k)})^T) \otimes (\boldsymbol{I}_n - \tilde{\mathcal{A}}_{\mathrm{rw}}^k)]^{-1}\mathrm{vec}(\boldsymbol{X}) \\ &\approx \alpha_0[(2-\alpha_0)\boldsymbol{I}_n - \sum_{k=1}^{t}(\alpha_k\Theta^{(k)}(\Theta^{(k)})^T) \otimes (\boldsymbol{I}_n - \tilde{\mathcal{A}}_{\mathrm{rw}}^k)]\mathrm{vec}(\boldsymbol{X}) \\ &= \alpha_0\mathrm{vec}[(2-\alpha_0)\boldsymbol{X} - \sum_{k=1}^{t}\alpha_k(\boldsymbol{I}_n - \tilde{\mathcal{A}}_{\mathrm{rw}}^k)\boldsymbol{X}\Theta^{(k)}(\Theta^{(k)})^T].\end{aligned}$$

If we let

$$\boldsymbol{W}^{(0)} = \frac{2-\alpha_0}{\alpha_0}\boldsymbol{I}_n - \sum_{k=1}^{t}\frac{\alpha_k}{\alpha_0}\Theta^{(k)}(\Theta^{(k)})^T; \tag{12}$$

$$\boldsymbol{W}^{(k)} = \Theta^{(k)}(\Theta^{(k)})^T), \;\; k = 1, 2, \ldots, t; \tag{13}$$

we can denote the convolution filter as:

$$\bar{\boldsymbol{X}} = \sum_{k=0}^{t}\alpha_k\tilde{\mathcal{A}}_{\mathrm{rw}}^k\boldsymbol{X}\boldsymbol{W}^{(k)}. \tag{14}$$

As we have stated in the Section 2.2.2, although the learning weights has a constrained expressive capability, it can be compensated by the following feature learning module. We omit the proofs of Definition 5 and 6, as they can be viewed as particular instances of (10).

Definition 7 Regularized Feature Variance. *Let $\otimes$ be the Kronecker product operator, $\mathrm{vec}(\boldsymbol{X}) \in \mathbb{R}^{nd}$ be the vectorized signal $\boldsymbol{X}$. Let $\boldsymbol{D}_X$ be a diagonal matrix whose value is defined by $\boldsymbol{D}_X(i,i) = \|x_{\cdot i}\|_2$. A graph convolution filter with regularized feature variance:*

$$\mathrm{vec}(\bar{\boldsymbol{X}}) = (\boldsymbol{I}_n \otimes [(\alpha_1+\alpha_2)I - \alpha_2\tilde{\mathcal{A}}_{\mathrm{rw}}] - \alpha_3[\boldsymbol{D}_x^{-1}(\boldsymbol{I} - \frac{1}{d}\boldsymbol{I}\boldsymbol{I}^T)\boldsymbol{D}_x^{-1}] \otimes \tilde{\boldsymbol{D}}^{-1})^{-1}\mathrm{vec}(\boldsymbol{X}) \tag{15}$$

is equivalent to *the optimal solution of the following optimization:*

$$\min_{\bar{\boldsymbol{X}}} \alpha_1\sum_{i\in\mathbb{V}}\|\bar{\boldsymbol{x}}(i) - \boldsymbol{x}(i)\|_{\tilde{\boldsymbol{D}}}^2 + \alpha_2\sum_{i,j\in\mathbb{V}}a_{ij}\|\bar{\boldsymbol{x}}(i) - \bar{\boldsymbol{x}}(j)\|_2^2 - \alpha_3\frac{1}{d}\sum_{i,j\in d}\|\bar{\boldsymbol{x}}_{\cdot i}/\|\boldsymbol{x}_{\cdot i}\| - \bar{\boldsymbol{x}}_{\cdot j}/\|\boldsymbol{x}_{\cdot j}\|\|_2^2, \tag{16}$$

where $\alpha_1 > 0$, $\alpha_2, \alpha_3 \geq 0$. For computation efficiency, we approximate $\boldsymbol{D}_{\bar{X}}$ with $\boldsymbol{D}_X$ as we assume that a single convolution filter provides little effect to the norm of features.

Proof. Let l denote the objective function. We have

$$l = \alpha_1\mathrm{tr}[(\bar{\boldsymbol{X}} - \boldsymbol{X})^T\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X})] + \alpha_2(\bar{\boldsymbol{X}}^T\boldsymbol{L}\bar{\boldsymbol{X}}) - \alpha_3\mathrm{tr}[\bar{\boldsymbol{X}}\boldsymbol{D}_x^{-1}(\boldsymbol{I} - \frac{1}{d}\boldsymbol{1}\boldsymbol{1}^T)\boldsymbol{D}_x^{-1}\bar{\boldsymbol{X}}^T].$$

Then,

$$\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 2\alpha_1\tilde{\boldsymbol{D}}(\bar{\boldsymbol{X}} - \boldsymbol{X}) + 2\alpha_2 L\bar{\boldsymbol{X}} - 2\alpha_3\bar{\boldsymbol{X}}\boldsymbol{D}_x^{-1}(\boldsymbol{I} - \frac{1}{d}\boldsymbol{1}\boldsymbol{1}^T)\boldsymbol{D}_x^{-1}.$$

If we let $\frac{\partial l}{\partial \bar{\boldsymbol{X}}} = 0$:

$$[(\alpha_1+\alpha_2)\boldsymbol{I} - \alpha_2\tilde{\boldsymbol{D}}^{-1}\tilde{\mathcal{A}}_{\mathrm{rw}}]\bar{\boldsymbol{X}} - \alpha_3\tilde{\boldsymbol{D}}^{-1}\bar{\boldsymbol{X}}\boldsymbol{D}_x^{-1}(\boldsymbol{I} - \frac{1}{d}\boldsymbol{1}\boldsymbol{1}^T)\boldsymbol{D}_x^{-1} = \alpha_1\boldsymbol{X}.$$

With the help of the Kronecker product operator $\otimes$, we have

$$(\boldsymbol{I}_n \otimes [(\alpha_1+\alpha_2)\boldsymbol{I} - \alpha_2\tilde{\mathcal{A}}_{\mathrm{rw}}] - \alpha_3[\boldsymbol{D}_x^{-1}(\boldsymbol{I} - \frac{1}{d}\boldsymbol{1}\boldsymbol{1}^T)\boldsymbol{D}_x^{-1}] \otimes \tilde{\boldsymbol{D}}^{-1})\mathrm{vec}(\bar{\boldsymbol{X}}) = \mathrm{vec}(\boldsymbol{X}). \tag{17}$$

By setting α_3 with a small positive value, the filter in Equation 17 is still a positive definite matrix. Therefore we complete the proof. □

Similarly, we can derive a simpler form via Taylor approximation. If we let:

$$\boldsymbol{A} = (\alpha_1+\alpha_2)\boldsymbol{I} - \alpha_2\tilde{\mathcal{A}}_{\mathrm{rw}}, \qquad \boldsymbol{B} = \boldsymbol{I}_n, \tag{18}$$

$$\boldsymbol{C} = -\alpha_3\tilde{\boldsymbol{D}}^{-1}, \qquad \boldsymbol{D} = \boldsymbol{D}_x^{-1}(1 - \frac{1}{d}\boldsymbol{1}\boldsymbol{1}^T)\boldsymbol{D}_x^{-1}. \tag{19}$$

Then, the first-order approximation of Equation 15 is summarized as:

$$\begin{aligned}\mathrm{vec}(\bar{\boldsymbol{X}}) &= (\boldsymbol{B}^T \otimes \boldsymbol{A} + \boldsymbol{D}^T \otimes \boldsymbol{C})^{-1}\mathrm{vec}(\boldsymbol{X}) \\ &\approx (2\boldsymbol{I} - \boldsymbol{B}^T \otimes \boldsymbol{A} - \boldsymbol{D}^T \otimes \boldsymbol{C})\mathrm{vec}(\boldsymbol{X}) \\ &= \mathrm{vec}(2\boldsymbol{X} - \boldsymbol{A}\boldsymbol{X}\boldsymbol{B} - \boldsymbol{C}\boldsymbol{X}\boldsymbol{D}).\end{aligned}$$

Additionally, we can also derive a t-order approximated formulation:

$$\mathrm{vec}(\bar{\boldsymbol{X}}^{(t)}) = (\boldsymbol{I} + \sum_{i=1}^{t}[\boldsymbol{I} - (\boldsymbol{B}^T \otimes \boldsymbol{A} + \boldsymbol{D}^T \otimes \boldsymbol{C})]^i)\mathrm{vec}(\boldsymbol{X}).$$

However, it is computationally expensive to calculate the Kronecker product. Therefore, we consider utilizing a iterative algorithm. For any $0 \leq k < t$

$$\begin{aligned}
\mathrm{vec}(\bar{\boldsymbol{X}}^{(k+1)}) &= (\boldsymbol{I} + \sum_{i=1}^{k+1}[\boldsymbol{I} - (\boldsymbol{B}^T \otimes \boldsymbol{A} + \boldsymbol{D}^T \otimes \boldsymbol{C})]^i)\mathrm{vec}(\boldsymbol{X}) \\
&= [\boldsymbol{I} - (\boldsymbol{B}^T \otimes \boldsymbol{A} + \boldsymbol{D}^T \otimes \boldsymbol{C})](\boldsymbol{I} + \sum_{i=1}^{k}[\boldsymbol{I} - (\boldsymbol{B}^T \otimes \boldsymbol{A} + \boldsymbol{D}^T \otimes \boldsymbol{C})]^i)\mathrm{vec}(\boldsymbol{X}) + \mathrm{vec}(\boldsymbol{X}) \\
&= [\boldsymbol{I} - (\boldsymbol{B}^T \otimes \boldsymbol{A} + \boldsymbol{D}^T \otimes \boldsymbol{C})]\mathrm{vec}(\bar{\boldsymbol{X}}^{(k)}) + \mathrm{vec}(\boldsymbol{X}) \\
&= \mathrm{vec}(\boldsymbol{X} + \bar{\boldsymbol{X}}^{(k)} - \boldsymbol{A}\bar{\boldsymbol{X}}^{(k)}\boldsymbol{B} - \boldsymbol{C}\bar{\boldsymbol{X}}^{(k)}\boldsymbol{D}).
\end{aligned} \tag{20}$$

B. Reformulation Examples

The reformulation examples of GCN derivatives are presented in Table 1.

Table 1: Reformulation of convolution-based graph neural networks. D and d_i in the attention-based modules are normalization coefficients.

Models	Non-Conv Module	Attention-based Module	Topology-based Module
GIN (Xu et al., 2018)	Residual Connection	-	-
GraphSAGE (Hamilton et al., 2017)	Concatenation	-	-
RGCN (Schlichtkrull et al., 2018)	Concatenation $\boldsymbol{W} = \sum_{r\in\mathbb{R}} \frac{1}{c_r}\boldsymbol{W}_r$	-	-
SplineCNN (Fey et al., 2018)	-	$p_{ij} = h_\theta(e_{ij})$	-
AGNN (Thekumparampil et al., 2018)	-	$p_{ij} = d_i \frac{\exp(\beta cos(\boldsymbol{x}_i, \boldsymbol{x}_j))}{\sum_{k\in\mathbb{N}(i)\cup i}\exp(\beta cos(\boldsymbol{x}_i, \boldsymbol{x}_k))}$	-
MoNet (Monti et al., 2017)	Concatenation	$p_{ij}^{(k)} = d_i \exp(-\frac{1}{2}(e_{ij} - \mu_k)^T \Sigma_k^{(-1)}(e_{ij} - \mu_k))$	-
GAT Veličković et al. (2018)	Concatenation	$p_{ij}^{(k)} = d_i \frac{\exp(\sigma(a_{(k)}^T[\theta\boldsymbol{x}_i \| \theta\boldsymbol{x}_j]))}{\sum_{k\in\mathbb{N}(i)\cup i}\exp(\sigma(a_{(k)}^T[\theta\boldsymbol{x}_i \| \theta\boldsymbol{x}_k]))}$	-
Cluster GCN (Chiang et al., 2019)	Concatenation	$\boldsymbol{P} = \boldsymbol{D}(\tilde{\mathcal{A}}_{\mathrm{rw}} + \lambda \mathrm{diag}(\tilde{\mathcal{A}}_{\mathrm{rw}}))$	
SGC (Wu et al., 2019)	-	$\boldsymbol{P} = \boldsymbol{D}\tilde{\mathcal{A}}_{\mathrm{sym}}^k$	-
Hyper-Atten (Bai et al., 2019)	-	$\boldsymbol{P} = \boldsymbol{H}\boldsymbol{W}\boldsymbol{B}^{-1}\boldsymbol{H}^T$	-
APPNP (Klicpera et al., 2018)	-	-	$\alpha_0 = \gamma, \alpha_1 = 1 - \gamma$
GDC (Klicpera et al., 2019)	-	-	$\alpha_i = \theta_i$
TAGCN (Du et al., 2017)	-	-	$\alpha_0 = \cdots = \alpha_k = 1/(k+1)$
MixHop (Kapoor et al., 2019)	Concatenation	-	$\alpha_0 = \alpha_1 = \alpha_2 = 1/3$

C. Data Statistics and Experimental Setups

We conduct experiments on four real-world graph datasets, whose statistics are listed in Table 2. For transductive learning, we evaluate our method on the Cora, Citeseer, Pubmed datasets, following

Table 2: Dataset Statistics

Dataset	Cora	Citeseer	Pubmed	PPI
Nodes	2,708	3,327	19,717	56,944(24 graphs)
Edges	5,429	4,732	44,338	818,716
Features	1,433	3,703	500	50
Classes	7	6	3	121(multilabel)
Training Nodes	140	120	60	44,906(20 graphs)
Validation Nodes	500	500	500	6,514(2 graphs)
Test Nodes	1,000	1,000	1,000	5,524(2 graphs)

the experimental setup in Sen et al. (2008). There are 20 nodes per class with labels to be used for training and all the nodes' features are available. 500 nodes are used for validation and the generalization performance is tested on 1000 nodes with unseen labels. PPI (Zitnik & Leskovec, 2017) is adopted for inductive learning, which is a protein-protein interaction dataset containing 20 graphs for training, 2 for validation and 2 for testing while testing graphs remain unobserved during training.

To ensure a fair comparison with other methods, we implement our module without interfering the original network structure. In all three settings, we use two convolution layers with hidden dimension $h = 64$. We set $\alpha_1 = 0.2$, $\alpha_2 = 0.8$ and $\alpha_3 = 0.05$ for all four datasets. We apply L_2 regularization with $\lambda = 0.0005$ and use dropout on both layers. For training strategy, we initialize weights using the initialization described in Glorot & Bengio (2010) and follow the method proposed in GCN, adopting an early stop if validation loss does not decrease for certain consecutive epochs. The implementations of baseline models are based on the PyTorch-Geometric library (Fey & Lenssen, 2019) in all experiments.

D. Random Splits

As illustrated in Shchur et al. (2018), using the same train/validation/test splits of the same datasets precludes a fair comparison of different architectures. Therefore, we follow the setup in Shchur et al. (2018) and evaluate the performance of our model on three citation networks with random splits. Empirically, for each dataset, we use 20 labeled nodes per class as the training set, 30 nodes per class as the validation set, and the rest as the test set. For every model, we choose the hyperparameters that achieve the best average accuracy on Cora and CiteSeer datasets and applied to Pubmed dataset.

Table 3 shows the results on three citation networks under the random split setting. As we can observe, our model consistently achieves higher performances on all the datasets. On Citeseer, our model achieves higher accuracy than on the original split. On Cora and Pubmed, the test accuracies of our model are comparable to the original split, while most of the baselines suffer from a serious decline.

Table 3: Test accuracy (%) on transductive learning datasets with random slits. We report mean values and standard deviations of the test accuracies over 100 random train/validation/test splits.

Dataset	Citeseer	Cora	Pubmed
GCN(Kipf & Welling, 2017)	71.9±1.9	81.5±1.3	77.8±2.9
GAT(Veličković et al., 2018)	71.4±1.9	81.8±1.3	78.7±2.3
MoNet (Monti et al., 2017)	71.2±2.0	81.3±1.3	78.6±2.3
GraphSAGE (Hamilton et al., 2017)	71.6±1.9	79.2±7.7	77.4±2.2
GCN+reg (ours)	**72.9±1.4**	**83.6±1.2**	**79.9±1.6**

References

S. Bai, F. Zhang, and P. Torr. Hypergraph convolution and hypergraph attention. *ArXiv*, abs/1901.08150, 2019.

Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, S. Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. 2019.

Jian Du, Shanghang Zhang, Guanhang Wu, José MF Moura, and Soummya Kar. Topology adaptive graph convolutional networks. *CoRR*, abs/1710.10370, 2017.

M. Fey, J. E. Lenssen, F. Weichert, and H. Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 869–877, 2018.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *CoRR*, abs/1903.02428, 2019.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

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

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

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

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

Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning, 2019.

Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5115–5124, 2017.

M. Schlichtkrull, Thomas Kipf, P. Bloem, R. V. Berg, Ivan Titov, and M. Welling. Modeling relational data with graph convolutional networks. In *ESWC*, 2018.

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. *CoRR*, abs/1811.05868, 2018.

Kiran K Thekumparampil, Chong Wang, Sewoong Oh, and Li-Jia Li. Attention-based graph neural network for semi-supervised learning. *CoRR*, abs/1803.03735, 2018.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. Simplifying graph convolutional networks. *International Conference on Machine Learning*, 2019.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *CoRR*, abs/1810.00826, 2018.

Marinka Zitnik and Jure Leskovec. Predicting multicellular function through multi-layer tissue networks. *Bioinformatics*, 33(14):i190–i198, 2017.