Table 3: Node classification results in terms of micro-f1 (20 labelled nodes per class for training).

dataset	Cora	Citeseer	Pubmed	Computer	Photo	CS	Physics
MLP	58.20±2.10	59.10±2.30	70.0±2.10	44.90±5.80	69.60±3.80	88.30±0.70	88.90±1.10
LogReg	57.10±2.30	61.00±2.20	64.10±3.10	64.10±5.70	73.00±6.50	86.40±0.90	86.70±1.50
LP	68.00±0.20	45.30±0.20	63.00±0.50	70.80±0.00	67.80±0.00	74.30±0.00	90.20±0.50
Chebyshev	81.20±0.50	69.80±0.50	74.40±0.30	62.60±0.00	74.30±0.00	91.50±0.00	92.10±0.30
MoNet	81.30±1.30	71.20±2.00	78.60±2.30	83.50±2.20	91.20±1.30	90.80±0.60	92.50±0.90
GCN	81.50±1.30	70.30±0.28	77.80±2.90	76.30±2.40	87.30±1.20	91.10±0.50	92.60±0.70
GAT	81.80±1.30	70.80±0.26	78.50±0.27	78.00±1.90	85.70±1.70	90.50±0.22	91.30±0.60
SGC	81.00±0.00	71.90±0.10	78.90±0.00	74.40±0.01	86.40±0.00	91.00±0.00	90.20±0.40
GCNII	**85.50±0.50**	73.40±0.60	**80.20±0.40**	57.11±13.92	63.03±4.43	88.30±1.25	OOM
APPNP	83.30±0.00	71.80±0.00	80.10±0.00	71.69±4.67	83.62±3.73	91.41±0.44	93.38±0.67
JKNet	81.10±0.00	69.80±0.00	78.10±0.00	64.08±2.10	78.10±7.07	87.07±1.34	92.69±0.73
C-GAT	80.60±0.45	70.99±0.37	79.60±0.11	OOM	OOM	OOM	OOM
GPRGNN	80.55±1.05	68.57±1.22	77.02±2.59	81.71±2.84	91.58±0.87	92.42±0.47	**93.51±0.59**
DMP	80.41±1.48	71.08±1.21	76.29±2.44	71.90±1.84	82.37±1.86	90.44±0.40	90.42±0.55
OPEN	81.68±0.44	**78.37±0.21**	79.35±0.25	**86.01±0.26**	**91.64±0.51**	**94.98±0.11**	92.17±0.56

A Proof of Theorem 1

In this section, the proof to Theorem 1 is given. To this end, some additional notations are defined in advance. Firstly, matrix $\mathbf{I}_v \in \mathbb{R}^{N\times N}$ is defined as the diagonal indicator matrix to indicate the neighbourhood of node v. It i^{th} diagonal element is 1, i.e., $(\mathbf{I}_v)_{ii} = 1$, if and only if node i is the neighbourhood of node v, i.e., $a_{vi} = 1$. Thus, the relationship between the v^{th} row of adjacency matrix $\mathbf{A}$, i.e., $\mathbf{a}_v$, and indicator matrix $\mathbf{I}_v \in \mathbb{R}^{N\times N}$ is

$$(\mathbf{1}\cdot\mathbf{a}_v)\odot\mathbf{I} = \mathbf{I}_v, \tag{12}$$

where $\odot$ denotes the element-wise product, $\mathbf{1}$ denotes vector of ones, and $\mathbf{I}$ stands for identity matrix. Thus, $\hat{\mathbf{H}}_v = \mathbf{I}_v\mathbf{H}$ only remains the rows corresponding to v's neighbourhoods and set other rows as zeros. In other word, $\hat{\mathbf{H}}_v \in \mathbf{R}^{N\times F}$ is the padding version of $\mathbf{H}_v \in \mathbf{R}^{|\mathcal{N}_v|\times F}$ with rows of zeros. Thus, the EVD in Eq. (6) can be extended to

$$\hat{\mathbf{S}}\hat{\mathbf{u}}_j = \lambda_j\hat{\mathbf{u}}_j, \quad j = 1, 2, ..., N \tag{13}$$

where $\hat{\mathbf{S}} = \hat{\mathbf{H}}_v\hat{\mathbf{H}}_v^T \in \mathbb{R}^{N\times N}$ and $\hat{\mathbf{u}}_j \in \mathbb{R}^N$ are the padding versions of $\mathbf{S} \in \mathbb{R}^{|\mathcal{N}_v|\times|\mathcal{N}_v|}$ and $\mathbf{u}_j \in \mathbb{R}^{|\mathcal{N}_v|}$, respectively. Thus, the obtained embedding for node v from our proposed OPEN can be written as

$$\mathbf{h}_v^{OPEN} = \mathbf{u}_1^T\mathbf{H}_v = \hat{\mathbf{u}}_1^T\hat{\mathbf{H}}_v = \hat{\mathbf{u}}_1^T\mathbf{I}_v\mathbf{H} = \left(\hat{\mathbf{u}}_1^T\odot\mathbf{a}_v\right)\mathbf{H}. \tag{14}$$

According to PCA, the mapping $\mathbf{u}_1$ and $\hat{\mathbf{u}}_1$ can be expressed as the combination of the columns of $\mathbf{H}_v$ and $\hat{\mathbf{H}}_v$, i.e.,

$$\mathbf{u}_1 = \sum_{f=1}^{F}\alpha_f\left(\mathbf{H}_v\right)_{\cdot,f} \quad \hat{\mathbf{u}}_1 = \sum_{f=1}^{F}\alpha_f\left(\hat{\mathbf{H}}_v\right)_{\cdot,f}. \tag{15}$$

It indicates that the propagation weights $\mathbf{u}_1$ and $\hat{\mathbf{u}}_1$ are not fully determined by the graph topology, but are mainly impacted by node attribute. Note that different from GAT, where propagation weights are determined by attributes of connected nodes and labels, the propagation weights in OPEN are not impacted by labels any more. Similar to Eq. (14), the embedding obtained from asymmetric adjacency matrix, such as GraphSAGE [5], can be written as

$$\mathbf{h}_v^{topology} = \frac{1}{d_v}\mathbf{1}^T\mathbf{H}_v = \frac{1}{d_v}\mathbf{1}^T\hat{\mathbf{H}}_v = \frac{1}{d_v}\mathbf{1}^T\mathbf{I}_v\mathbf{H} = \left(\frac{1}{d_v}\mathbf{1}^T\odot\mathbf{a}_v\right)\mathbf{H}. \tag{16}$$

Comparing Eqs. (14) and (16), the main difference between OPEN and propagation based on asymmetric adjacency matrix is that the topology-based propagation weight $\frac{1}{d_v}\mathbf{1}$ in propagation based on asymmetric adjacency matrix is replaced by attribute-based propagation weight $\mathbf{u}_1$ (Eq. 15) in the proposed OPEN. Therefore, the representation of node v, i.e., $\mathbf{h}_v^{OPEN}$, is relevant to the principal components of node v's ego-network's attribute, i.e., $\hat{\mathbf{u}}_1^T$.

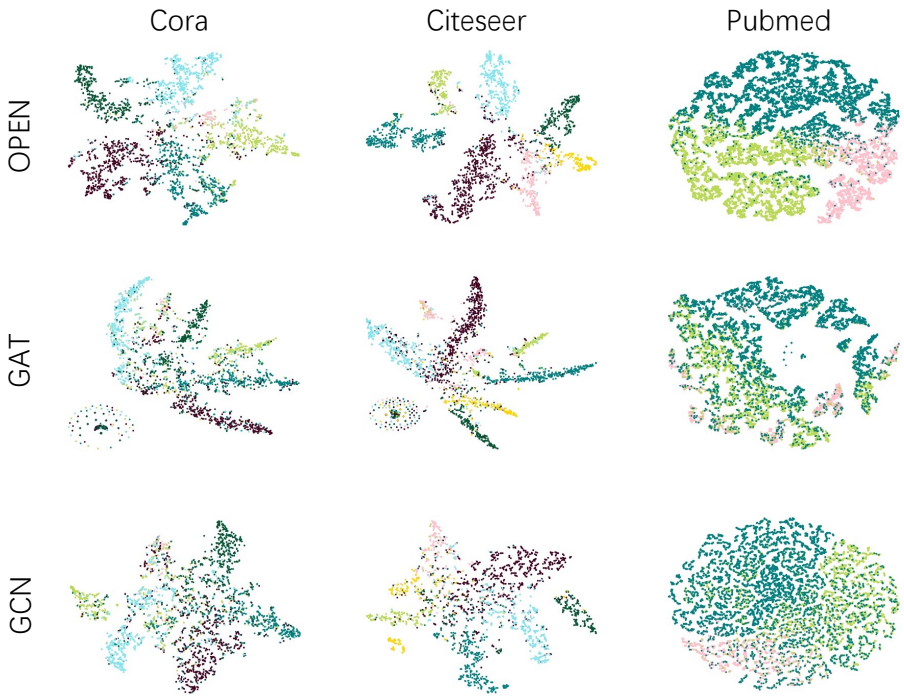

Figure 5: The visualizations of the embeddings obtained from GCN, GAT and OPEN.

B Additional Experimental Results

This section provides additional experimental results to verify some statements and the superiority of the proposed OPEN.

B.1 Node Classification on Another Split

Firstly, to demonstrate the superior performance of the proposed OPEN. Node classification results based on another split, where 20 labelled nodes per class are employed as the training as in GPRGNN [27]. The results are given in Tab. 3. We obtain the similar conclusion as in the main body. Especially, OPEN achieves remarkable improvements on large networks, such as Amazon-Computer and Coauthor-CS. Since the overfitting issue may be more serious on large networks with little nodes labelled, this demonstratesx the capability of OPEN on preventing over-smoothing issue.

B.2 Embedding Visualization

To provide intuitive interpretation, the t-SNE visualizations of node embeddings obtained from GCN, GAT and OPEN are given in Fig. 5. The regions of embeddings of nodes from different classes are with different colors. The shapes of these regions reflect the characteristics of the corresponding methods. The GCN's regions of the embeddings for different classes are overlapped. Thus, the GCN model tends to be under-fitting. The GAT's and OPEN's regions of the embeddings for different classes are distinct. The regions of the embeddings from GAT are very sharp. It indicates that labelled data plays a very essential rule on the embedding, which tends to be overfitting. In contrary, the regions of the embeddings from OPEN are regular. It indicates that the graph topology, which induces smoothing effect, play more important role than the labels., and thus can prevent overfitting. Besides, the results from OPEN are much better than those from GAT on pubmed dataset. GAT's embeddings of nodes with pink color are overlapped from nodes of other colors, while OPEN's embeddings of nodes with pink color are distinct from nodes of other colors. This illustrates the effectiveness of the proposed OPEN compared to GCN and GAT.

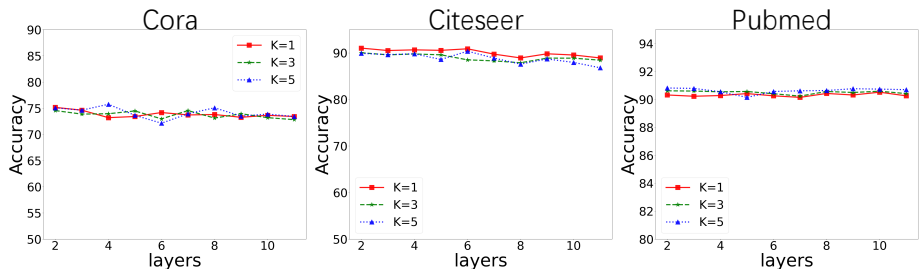

Figure 6: Node classification performances with different numbers of both channels and layers

Table 4: Running Time Comparison (in seconds).

dataset	Cora	Citeseer	Pubmed	Computer	Photo	CS	Physics
GCN	9.89	6.23	5.32	16.8	6.59	19.2	21.58
GAT	10.45	49.31	12.85	95.23	42.11	106.06	201.79
OPEN-Weight	2.61	8.93	3.05	18.46	9.22	18.39	52.94
OPEN-Propagation	10.11	36.09	12.8	65.36	35.62	88.81	149.37
OPEN-layer	271.11	929.09	317.8	1911.36	957.62	1927.81	5443.37
OPEN	12.72	45.02	15.85	83.82	44.84	107.2	202.31

B.3 Ablation Study on Prevent Over-smoothing Issue

Section 4.6 demonstrates the capability of OPEN on preventing over-smoothing issue. DMP [20] proves that the diverse multi-channel propagations provide potentials to prevent over-smoothing issue To investigate which component is more important, Ego-Network modeling or Orthogonal Propagation, ablation study is performed in this section. To this end, the node classification performances with different numbers of channels K and different numbers of layers are given in Fig. 6. It can be observed that OPEN with different K can prevent oversmoothing issue. Thus, ego-network modeling component, which is equivalent to OPEN with $K = 1$, can prevent over-smoothing issue. Note that it is hard to only employ orthogonal propagation without ego-network modeling. Thus, we do not experimentally evaluate the effectiveness of this component. DMP [20] shows that the diversity of the channels promote the ability on preventing over-smoothing issue. Essentially, the orthogonality in multi-channels in OPEN realize the requirement on diversity. Therefore, both two components can prevent over-smoothing issue.

B.4 Running Time Comparisons

To demonstrate the efficiency of the proposed OPEN, the running time comparisons are given in Table. 4. OPEN-Weight and OPEN-Propagation are the weight calculation and propagation parts of the OPEN. Compared to OPEN, where weights are fixed, OPEN-layer denote the variant which learns weights in each layer. The results show that OPEN has the similar running time as GAT. The running time of GAT and OPEN is similar. Note that the running time of GAT and OPEN is longer than that of GCN, due to their multiple-channel propagations and combinations. Besides, while the weight calculation is efficient compared to propagation, the afford of layer-wise weight calculation, i.e., OPEN-layer, is too high. These meet our complexity analysis in Section 3.1.

C Algorithm Description of OPEN

To make the OPEN easy to follow, algorithms description are given. Table. 5 provides the ego-network modeling algorithm, whose output is the propagation weights. Table. 6 is the algorithm of OPEN-layer, which performs the ego-network modeling in each layer. Table. 7 is the final OPEN algorithm, where the ego-network modeling is only performed once.

Table 5: Algorithm-1: Ego-network modeling via PCA

Algorithm-1: Ego-network modeling via PCA

Input: Ego-network of node v: $\mathbf{H}_v = \{\mathbf{h}_{.,1}, \mathbf{h}_{.,2}, \ldots, \mathbf{h}_{.,F}\} \in \mathbf{R}^{|N_v| \times F}$
Output: J eigenvectors of the ego-network of node v: $\{\mathbf{u}_{1,v}, \mathbf{u}_{2,v}, \ldots, \mathbf{u}_{J,v}\}$

Step1: Calculate covariance matrix of $\mathbf{H}_v$:$\mathbf{S}_v = 1/F \sum_{j=1}^{F} (\mathbf{h}_{.,j} - \overline{\mathbf{h}})(\mathbf{h}_{.,j} - \overline{\mathbf{h}})^T$;
Step2: Calculate eigenvectors of covariance matrix $\mathbf{S}_v$ via **Eq**. 6;
Step3: Generate top J eigenvectors:$\{\mathbf{u}_{1,v}, \mathbf{u}_{2,v}, \ldots, \mathbf{u}_{J,v}\}$ via sorting λ_j in descending order;
return $\{\mathbf{u}_{1,v}, \mathbf{u}_{2,v}, \ldots, \mathbf{u}_{J,v}\}$.

Table 6: OPEN-layer

Algorithm-2: OPEN-layer

Input: Feature matrix $\mathbf{X} \in \mathbf{R}^{N \times F}$, Adjacency matrix $\mathbf{A} \in \mathbf{R}^{N \times N}$
Output: Node representations: $\mathbf{H}$

for l=1 to L **do**
  for v=1 to N **do**
    % Ego-network modeling %
    Generate ego-network of v from $\{\mathbf{H}^{(l-1)}, \mathbf{A}\}$: $\mathbf{H}_v^{(l-1)}$;
    % Generate propagation weights for l-th layer %
    Generate propagation weights of v:$\{\mathbf{u}_{1,v}^{(l-1)}, \mathbf{u}_{2,v}^{(l-1)}, \ldots, \mathbf{u}_{J,v}^{(l-1)}\}$ via Algorithm-1 on $\mathbf{H}_v^{(l-1)}$;
    % Orthogonal propagations in multi-channels %
    Update representation of v of channel j: $\mathbf{h}_{j,v}^{(l)\,T} = \mathbf{u}_{j,v}^{(l-1)\,T} \mathbf{H}_v^{(l-1)}, j = 1, 2, \ldots, J$;
    Update representation of v from different channels via **Eq**. 10;
  end for
end for
return $\mathbf{H}^{(L)} = \{\mathbf{h}_1^{(L)}, \mathbf{h}_2^{(L)}, \ldots, \mathbf{h}_N^{(L)}\}$.

Table 7: OPEN

Algorithm-3: OPEN

Input: Feature matrix $\mathbf{X} \in \mathbf{R}^{N \times F}$, Adjacency matrix $\mathbf{A} \in \mathbf{R}^{N \times N}$
Output: Node representations: $\mathbf{H}$

for v=1 to N **do**
  % Ego-network modeling %
  Generate ego-network of v from $\{\mathbf{X}, \mathbf{A}\}$: $\mathbf{H}_v$;
  % Generate propagation weights for all layers %
  Generate propagation weights of v: $\{\mathbf{u}_{1,v}, \mathbf{u}_{2,v}, \ldots, \mathbf{u}_{J,v}\}$ via Algorithm-1 on $\mathbf{H}_v$;
end for
for l=1 to L **do**
  for v=1 to N **do**
    % Ego-network modeling %
    Generate ego-network of v from $\{\mathbf{H}^{(l-1)}, \mathbf{A}\}$: $\mathbf{H}_v^{(l-1)}$;
    % Orthogonal propagations in multi-channels %
    Update representation of v of channel j: $\mathbf{h}_{j,v}^{(l)\,T} = \mathbf{u}_{j,v}^{T} \mathbf{H}_v^{(l-1)}, j = 1, 2, \ldots, J$;
    Update representation of v from different channels via **Eq**. 10;
  end for
end for
return $\mathbf{H}^{(L)} = \{\mathbf{h}_1^{(L)}, \mathbf{h}_2^{(L)}, \ldots, \mathbf{h}_N^{(L)}\}$.