Q1. *The statement of "quality and quantity of neighbors" seems somewhat subjective and not very easy to measure or verify.*

Reply:

(1) we can define quality and quantity as follows:

Neighbor quality: If two nodes receive the same or similar information from neighbors, we argue that the neighbor quality of the two nodes is low from the perspective of oversmoothing. This can happen to two neighboring nodes in shallow layers as shown in Figure a1 and a2 (nodes A and B). If two nodes have very low-quality neighbors, they will become similar very quickly (oversmoothing happens). According to the small-world phenomenon [17], any two nodes will become neighbors directly or indirectly by 6 jumps. Therefore, with layers increasing, the neighbor quality will become low between any two nodes as shown in Figures a1 and a2 (nodes A and C)

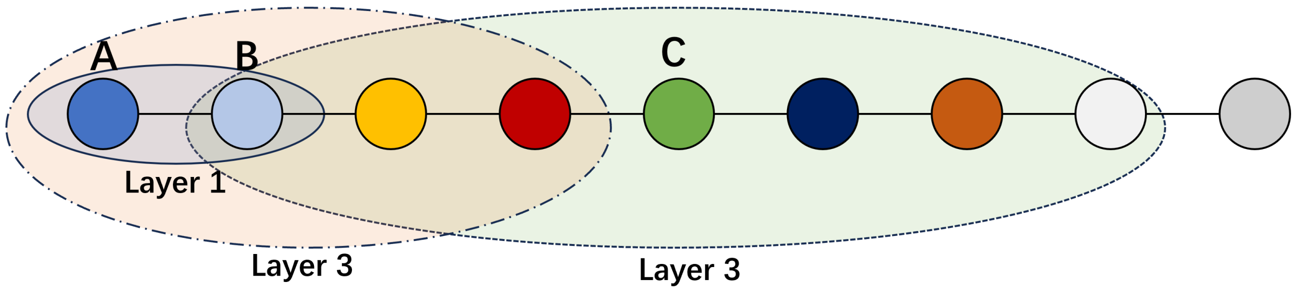


Figure a1. A simplest graph to demonstrate node similarity.

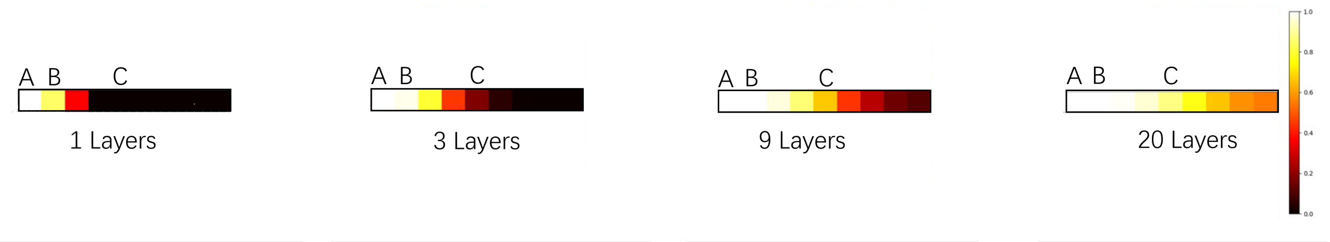


Figure a2. Node similarity changing between A and others. A and B become similar in the 1st layer, because they are neighbor to each other. With layers increasing, A and C becomes similar in layer >3 where they share similar neighbors.

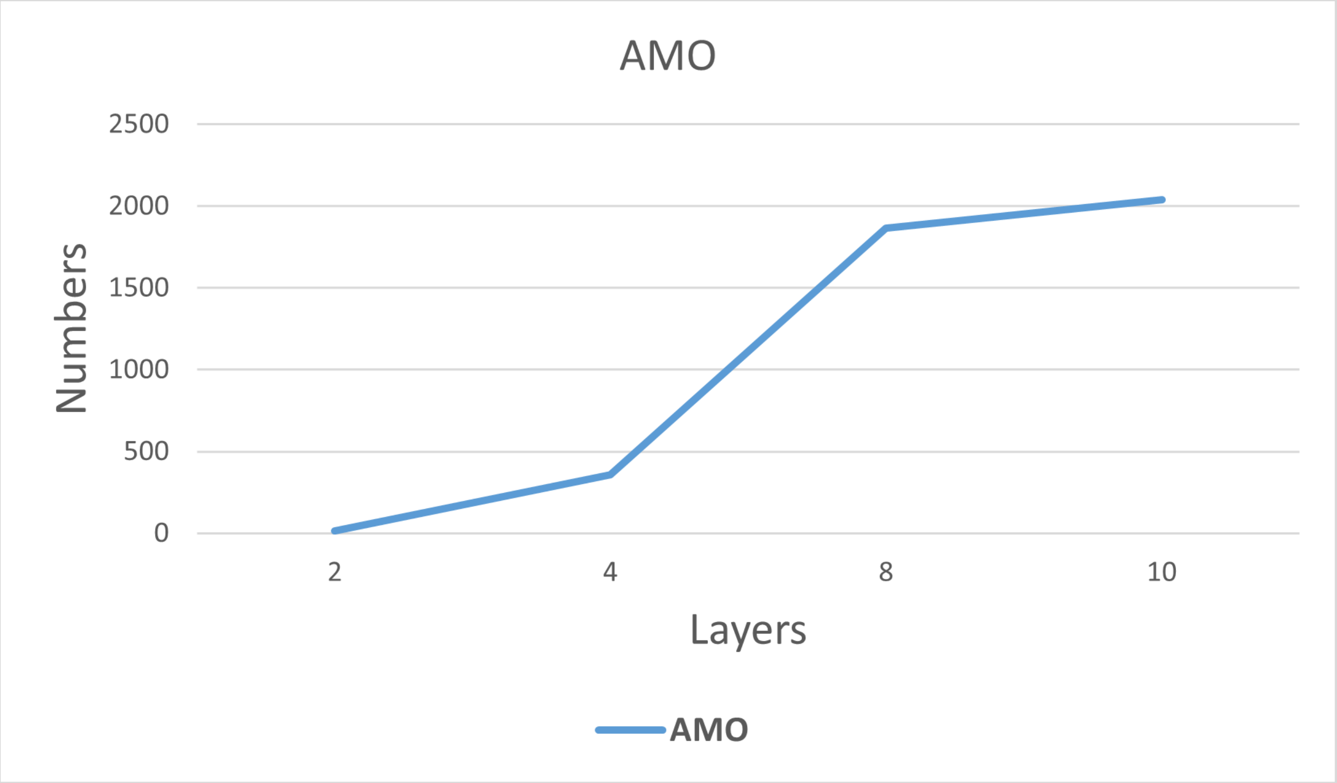
We can use an indicator **Average Mutual Overlapping** (AMO) to describe the neighborhood quality. AMO is defined as follows:

$$S_{i,j} = \begin{cases} 1, A^{l}_{i,j} > 0 \wedge i \neq j \\ 0, A^{l}_{i,j} = 0 \end{cases}, Num = \mathrm{Mean}(\mathrm{SS}^{T}))$$

l is the number of layers and A is the adjacency matrix.

As its name suggests, this indicator averages the number of common neighbors of two nodes over all possible pairs of nodes. It is a sufficient indicator since two nodes have similar aggregation information when they have a large number of common neighbors.

On cora, we stack SGC by 2, 4, 8, and 10 layers and measure the AMO. The measurement of the AMO is presented in the Fig. a3.



.

Figure a3. The neighbor quality changeing with respoect to layer numbers

Neighbor quantity: The information received from neighbors grows exponentially, causing nodes to lose their own individuality and accuracy decreases. As shown in Fig. a1, node C has 6 neighbors in layer 3. If the graph become non-linear, the number of neighbors will be exponential to layer number. The Neighbor quantity can be measured by **Average Number of Different Classes of Nodes in the neighbourhood** :

$$ANDCN = \frac{1}{N}\sum_{i=0 j=0}^{N} 1_{[L_i != L_j \wedge S_{i,j}=1]},$$

where N is the number of nodes, L_i the node i's label,$S_{i,j}$ is defined in AMO.

We plot the variation of $ANDCN$ in layers 2, 4, 8 and 10 on the cora dataset in Figure a4.:

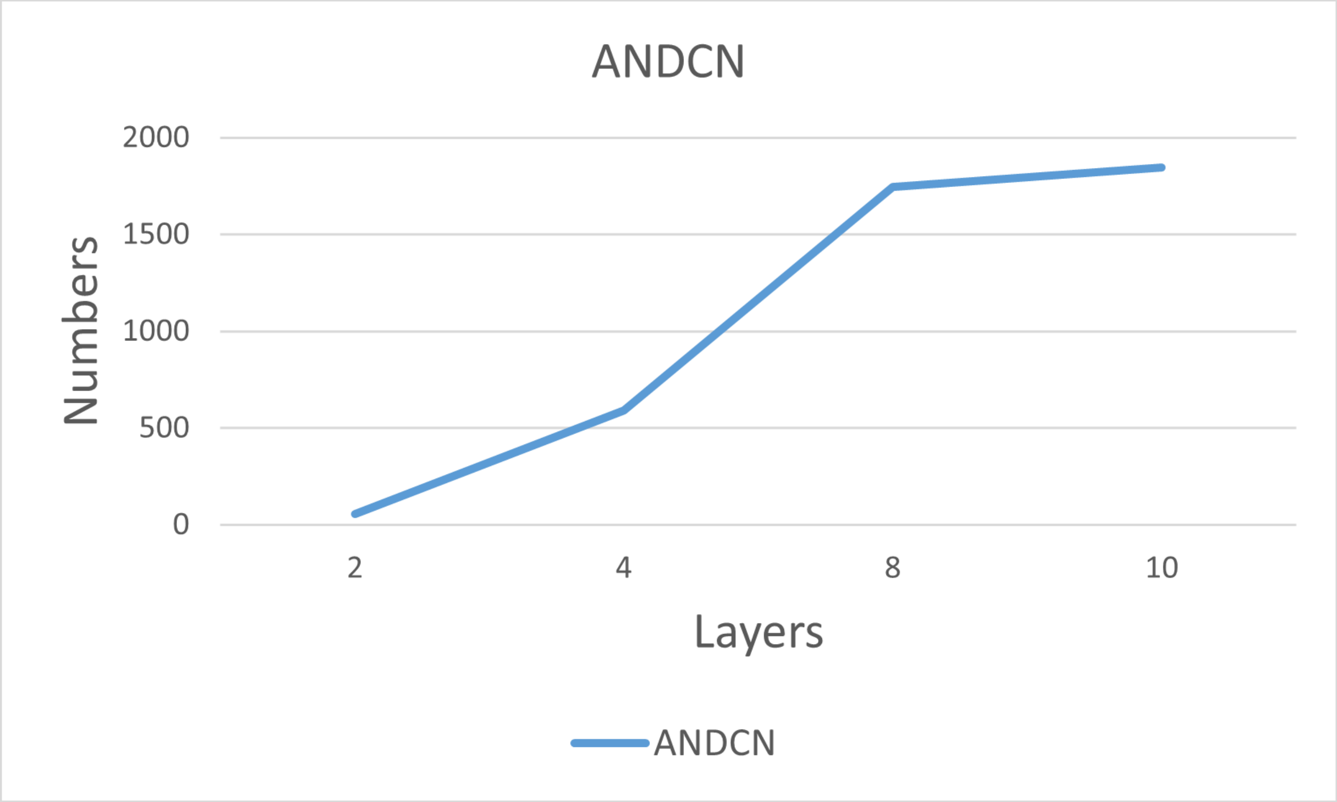


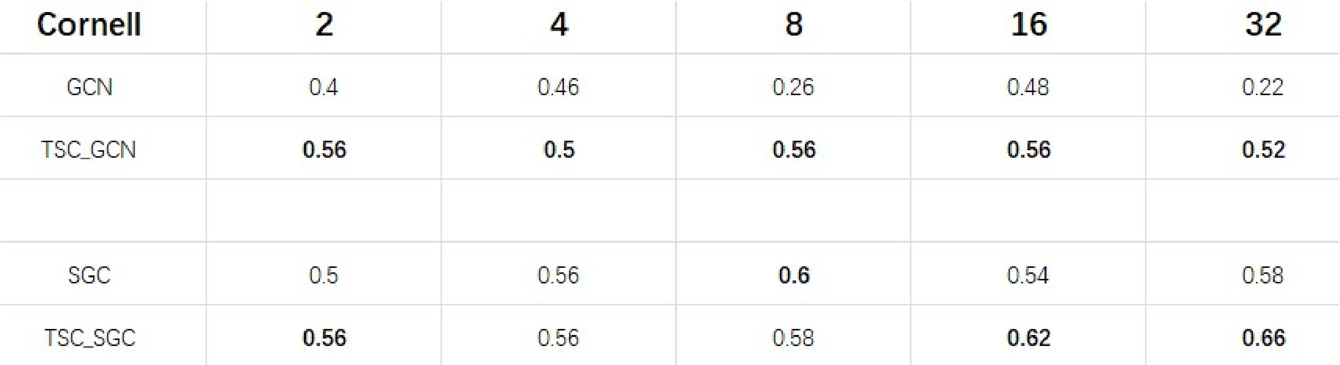
Figure a4. The neighbor quality changeing with respoect to layer numbers

(2) Our motivation can be generally summarized: (a) using random masking to drop a large number of aggregated information and maintain the node's individual information generate from shallow layers. (b) adopting contrastive constraint to enhance node's difference to each other in deep layers.

Q2. *Current datasets are small datasets; stacking layers is typically necessary for large-scale datasets, and some large-scale datasets can be added.*
Reply: Yes, stacking layers also work for large-scale datasets. In our draft, the datasets chosen are standard for comparison to previous methods for handling oversmoothing. We have added AmazonPhoto and CoauthorCS, which are somewhat larger datasets compared to cora and citeseer in the experiments; the empirical results demonstrate the superiority of our model over others on both datasets.

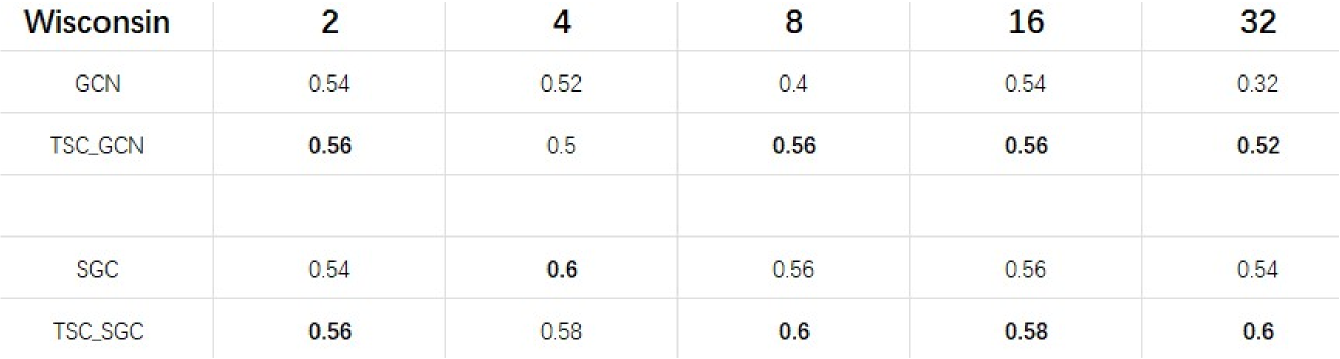
Q3. *Also, stacking layers is also necessary when the close neighbors do not share similar labels, that's why we need more layers to aggregate far neighbors' information. Some heterophylious datasets are better to be included.*
Reply: We added two heterogeneous datasets and the experimental results show the advantages of our approach. The the ACC on Wisconsin and Cornell are reported in Tables a1 and a2.

Table a1. ACC on Cornell



Cornell	2	4	8	16	32
GCN	0.4	0.46	0.26	0.48	0.22
TSC_GCN	**0.56**	**0.5**	**0.56**	**0.56**	**0.52**
SGC	0.5	0.56	**0.6**	0.54	0.58
TSC_SGC	**0.56**	0.56	0.58	**0.62**	**0.66**

Table a2. ACC on Wisconsin



Wisconsin	2	4	8	16	32
GCN	0.54	0.52	0.4	0.54	0.32
TSC_GCN	**0.56**	0.5	**0.56**	**0.56**	**0.52**
SGC	0.54	**0.6**	0.56	0.56	0.54
TSC_SGC	**0.56**	0.58	**0.6**	**0.58**	**0.6**

Q4 *The conclusion for Eq (11) mitigating overshooting is not very straightforward; more explanations will be appreciated.*

Reply:

The Eq (11) is $H^{(l)} \ = f(\ L^l X)$. It does not mitigating oversmoothing, and we explain why our model is able to mitigate oversmoothing in Eqs. (15), (16), and (17), with the corrected Eqs. (15), (16) and (17) as follows:
Eq. (15) amend to read as follows, and $f(\cdot)$ stands for linear transformation.
$H^{(l)} = f_l(L^l X)M^{(l)} + (1 - M^{(l)})f_{l-1}(L^{l-1}X)$
We have amended Equation 16 as follows:

$$\begin{aligned}\underset{l\to\infty}{lim} H^{(l)} &= \underset{l\to\infty}{lim} f_1(LX)M^{(1)} + (1 - M^{(1)})X + f_2(L^2X)M^{(2)} + (1 - M^{(2)})f_1(LX) + ...\\ &\quad + f_l(L^lX)M^{(l)} + (1 - M^{(l)})f_{l-1}(L^{l-1}X)\\ &= \underset{l\to\infty}{lim} \textstyle\sum_{k=1}^{l-1}(M^{(k)} + (1 - M^{(k+1)}))f_k(L^kX) + f_l(L^lX)M^{(l)} + (1 - M^{(1)})X\end{aligned}$$

Eq.(17) amend to read as follows:

$$\underset{l\to\infty}{lim} \textstyle\sum_{k=1}^{l-1} \widehat{M}^{(k)} f_k(L^kX) + \widehat{M}^{(l)} f_l(L^lX) + \overrightarrow{M}X$$

$$= \underset{l\to\infty}{lim} \textstyle\sum_{k=1}^{l} \widehat{M}^{(k)} H^{(k)} + \overrightarrow{M}X$$

Q5 *How is L_{GCN} analyzed under Eq (18) and Eq (19)?*

Reply:

After dropout in Equations (7) and (8), two views of the original graph -- topologically identical to the original graph but with different dropout features at each node-- are created.

$$L_{GCN} = l = -\sum_i^n log \frac{e^{\left(s(\widehat{h}_i^{(l)}, \widetilde{h}_i^{(l)})/\tau\right)}}{\sum_{j\neq i}^N e^{\left(s(\widehat{h}_i^{(l)}, \widehat{h}_j^{(l)})/\tau\right)} + e^{\left(s(\widehat{h}_i^{(l)}, \widetilde{h}_j^{(l)})/\tau\right)}}.$$

During the training, the alignment loss aligns the features (namely, $\widehat{h_i}$, $\widetilde{h_i}$) of the same node in the two views and the heterogeneity loss separates the features (namely, $\widehat{h_i}$, $\widetilde{h_j}$) of the different nodes in the two views:

$$l_{align}(i) = s(\widehat{h}_i^{(L)}, \widetilde{h}_i^{(L)})/\tau$$

$$l_{heter}(i) = log(\sum_{j\neq i}^n e^{\left(s(\widehat{h}_i^{(L)}, \widehat{h}_j^{(L)})/\tau\right)} + e^{\left(s(\widehat{h}_i^{(L)}, \widetilde{h}_j^{(L)})/\tau\right)})$$