R4:Official Review of Submission1720 by Reviewer sSK4

W1. *It is claimed that "Compared with GCN, SGC preserves more of the original node features, preventing them from becoming indistinguishable, and consequently alleviating over-smoothing." I am not sure why this is true. The normalized matrix L (S in the cited paper) will have one eigenvalue equal to one and all the other ones will be smaller. With a high value of l, the contribution of the smaller eigenvalues will disappear leaving$S^kK = ee^TX = e(e^TX)$, where e is the eigenvector corresponding to the eigenvalue 1. Therefore, all the nodes see the same row matrix (e^TX) pre-multiplied by a column matrix e that only depends on the graph and not the features. This is oversmoothing.*
Reply: It is true that when using the normalised adjacency matrix (also known as the Laplace matrix) for graph convolution, one of the eigenvalues will be equal to 1, and the others will all be less than or equal to 1. This will result in the small eigenvalues disappearing over the course of multiple convolution operations, leading to overly smoothed node features, and ultimately making nodes difficult to distinguish from each other. However, in practice, we will stop layer staking before the eigenvalues converge to 1. Consequently, we study the anti-oversmoothing-technique to collect more distant information to improve discriminative power before oversmoothing.

W2. *Not sure, but it seems to me that the masking matrix should be multiplied with element-wise product rather than matrix multiplication.*

Reply: The operations we have in the code are also element-wise, we just transform them into matrix operation form when we write them, they are the same.

W3. *In Section 5, claim (a), it is mentioned that as l approaches infinity, $d_i = d_j = N$. I believe this is incorrect. In Equation (13), d_i is the degree in L, not in $L^{(l)}$.*

Reply: yes, in Eq. 13, d_i is the degree in L, not in $L^{(l)}$, but at this point $N_i = V$ so we correct Eq. 14, which becomes as follows:
$\lim\limits_{l\to\infty} H_i^{(l)} = f(\frac{\sqrt{d_i+1}}{2m+n} \times \sum_{j\in V}(\sqrt{d_j + 1}X_j)))$.

W4. Derivation of Equations (16) and (17) has issues: It seems like f is assumed to be identity, otherwise, we should see compositions of f. Even then, the expansion in (16) is incorrect as the order of matrix multiplications has been changed to simplify, which cannot be done.
Reply: Thanks for the comments. Corrections have been made to equations (15), (16) and (17).
$H^{(l)} = f(L^lX)M^{(l)} + (1 - M^{(l)})f(L^{l-1}X)$ **(15)**

$$\lim_{l\to\infty} H^{(l)} = \lim_{l\to\infty} f(LX)M^{(1)} + (1 - M^{(1)})X + f(L^2X)M^{(2)} +$$
$$= \lim_{l\to\infty} \sum_{k=1}^{l-1}(M^{(k)} + (1 - M^{(k+1)}))f(L^kX) + f(L^lX)M^{(l)} + (1 - M^{(1)})X$$

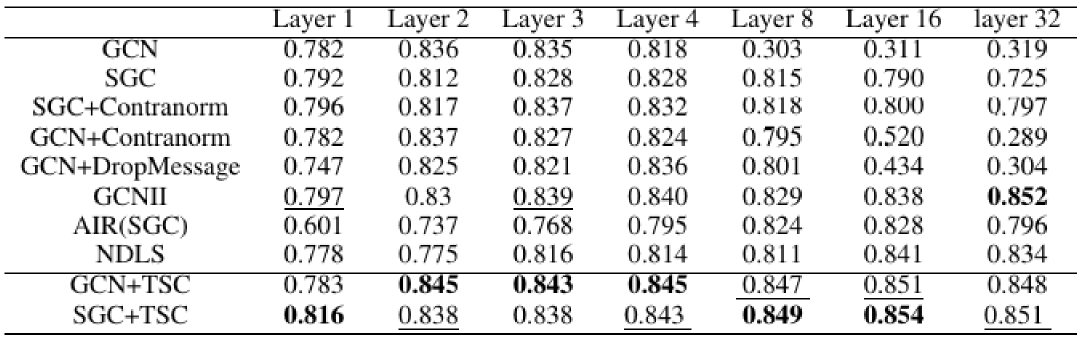
$$(1-M^{(2)})f(LX)+\ldots+f(L^lX)M^{(l)}+(1-M^{(l)})f(L^{l-1}X) \quad \textbf{(16)}$$

$$\underset{l\to\infty}{lim}\sum_{k=1}^{l-1}\widehat{M}^{(k)}f(L^kX)+\widehat{M}^{(l)}f(L^lX)+\overrightarrow{M}X$$
$$=\underset{l\to\infty}{lim}\sum_{k=1}^{l}\widehat{M}^{(k)}H^{(k)}+\overrightarrow{M}X \quad \textbf{(17)}$$

W5. In Section 5, claim (b), what is shown is an explanation of the loss function, and does not necessarily prove that it has more "discriminative power".

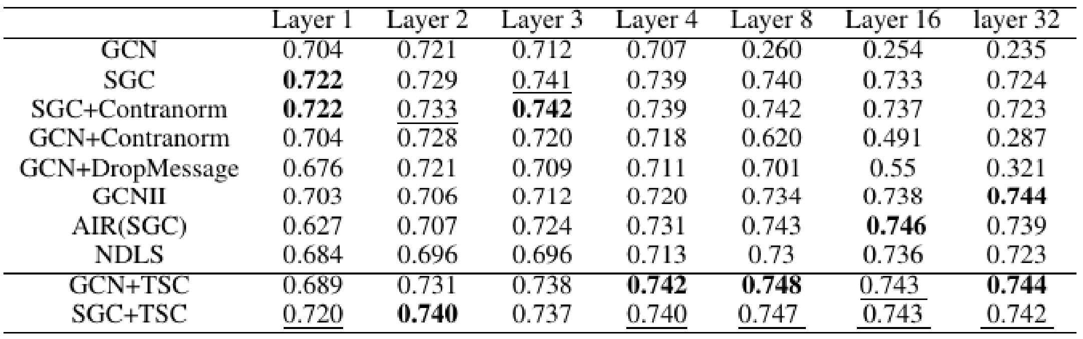
Reply: Thanks for this suggestion. The "discriminative power" mainly benefits from equation (19) which will make the nodes to be heterogeneous. This claim has been demonstrated by contrastive learning methods. Actually, we didn't provide experiments to validate it, we will adjust this claim in revisions.

W6. Section 6.3 focuses on over-smoothing, and so the number of layers are selected to be 4, 8, 16, 32. I think results for layers 1, 2 and 3 should be shown to understand if it is necessary to go beyond 3. If the accuracy drops beyond 3, there is no guarantee that the drop is due to oversmoothing alone.
Reply: We have done experiments on layers 1, 2 and 3 of each model on the datasets and the results are shown as follows:



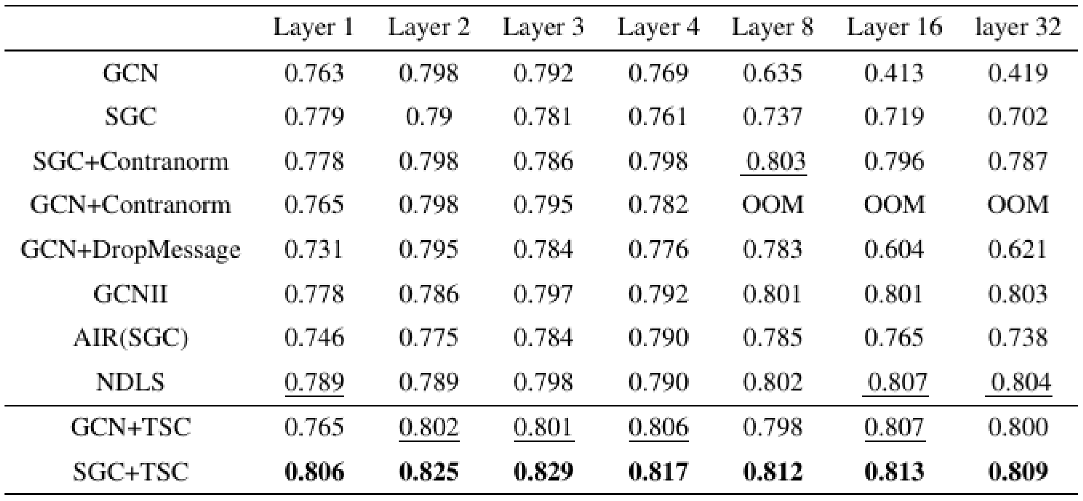
	Layer 1	Layer 2	Layer 3	Layer 4	Layer 8	Layer 16	layer 32
GCN	0.782	0.836	0.835	0.818	0.303	0.311	0.319
SGC	0.792	0.812	0.828	0.828	0.815	0.790	0.725
SGC+Contranorm	0.796	0.817	0.837	0.832	0.818	0.800	0.797
GCN+Contranorm	0.782	0.837	0.827	0.824	0.795	0.520	0.289
GCN+DropMessage	0.747	0.825	0.821	0.836	0.801	0.434	0.304
GCNII	0.797	0.83	0.839	0.840	0.829	0.838	**0.852**
AIR(SGC)	0.601	0.737	0.768	0.795	0.824	0.828	0.796
NDLS	0.778	0.775	0.816	0.814	0.811	0.841	0.834
GCN+TSC	0.783	**0.845**	**0.843**	**0.845**	0.847	0.851	0.848
SGC+TSC	**0.816**	0.838	0.838	0.843	**0.849**	**0.854**	0.851

Table 1: ACC Comparison in Different Depth on Cora



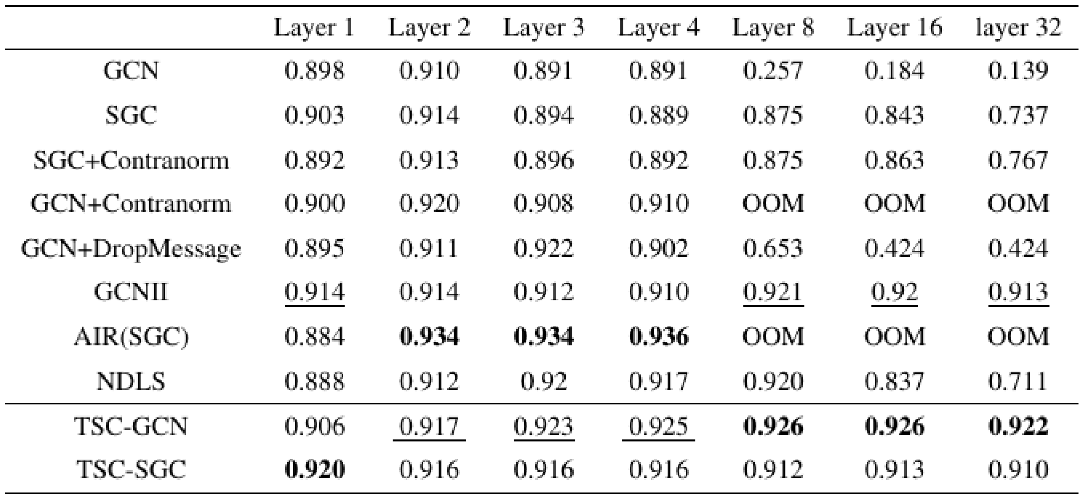
	Layer 1	Layer 2	Layer 3	Layer 4	Layer 8	Layer 16	layer 32
GCN	0.704	0.721	0.712	0.707	0.260	0.254	0.235
SGC	**0.722**	0.729	0.741	0.739	0.740	0.733	0.724
SGC+Contranorm	**0.722**	0.733	**0.742**	0.739	0.742	0.737	0.723
GCN+Contranorm	0.704	0.728	0.720	0.718	0.620	0.491	0.287
GCN+DropMessage	0.676	0.721	0.709	0.711	0.701	0.55	0.321
GCNII	0.703	0.706	0.712	0.720	0.734	0.738	**0.744**
AIR(SGC)	0.627	0.707	0.724	0.731	0.743	**0.746**	0.739
NDLS	0.684	0.696	0.696	0.713	0.73	0.736	0.723
GCN+TSC	0.689	0.731	0.738	**0.742**	**0.748**	0.743	**0.744**
SGC+TSC	0.720	**0.740**	0.737	0.740	0.747	0.743	0.742

Table 2: ACC Comparison in Different Depth on Citeseer



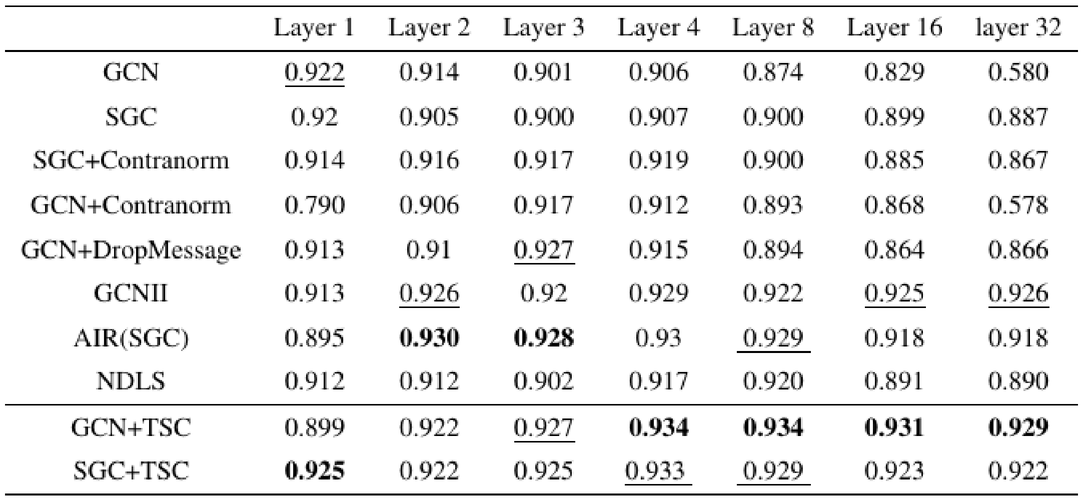
	Layer 1	Layer 2	Layer 3	Layer 4	Layer 8	Layer 16	layer 32
GCN	0.763	0.798	0.792	0.769	0.635	0.413	0.419
SGC	0.779	0.79	0.781	0.761	0.737	0.719	0.702
SGC+Contranorm	0.778	0.798	0.786	0.798	0.803	0.796	0.787
GCN+Contranorm	0.765	0.798	0.795	0.782	OOM	OOM	OOM
GCN+DropMessage	0.731	0.795	0.784	0.776	0.783	0.604	0.621
GCNII	0.778	0.786	0.797	0.792	0.801	0.801	0.803
AIR(SGC)	0.746	0.775	0.784	0.790	0.785	0.765	0.738
NDLS	0.789	0.789	0.798	0.790	0.802	0.807	0.804
GCN+TSC	0.765	0.802	0.801	0.806	0.798	0.807	0.800
SGC+TSC	**0.806**	**0.825**	**0.829**	**0.817**	**0.812**	**0.813**	**0.809**

Table 3: Performance comparison results on Pubmed



	Layer 1	Layer 2	Layer 3	Layer 4	Layer 8	Layer 16	layer 32
GCN	0.898	0.910	0.891	0.891	0.257	0.184	0.139
SGC	0.903	0.914	0.894	0.889	0.875	0.843	0.737
SGC+Contranorm	0.892	0.913	0.896	0.892	0.875	0.863	0.767
GCN+Contranorm	0.900	0.920	0.908	0.910	OOM	OOM	OOM
GCN+DropMessage	0.895	0.911	0.922	0.902	0.653	0.424	0.424
GCNII	0.914	0.914	0.912	0.910	0.921	0.92	0.913
AIR(SGC)	0.884	**0.934**	**0.934**	**0.936**	OOM	OOM	OOM
NDLS	0.888	0.912	0.92	0.917	0.920	0.837	0.711
TSC-GCN	0.906	0.917	0.923	0.925	**0.926**	**0.926**	**0.922**
TSC-SGC	**0.920**	0.916	0.916	0.916	0.912	0.913	0.910

Table 4: Performance comparison results on CoauthorCS



	Layer 1	Layer 2	Layer 3	Layer 4	Layer 8	Layer 16	layer 32
GCN	0.922	0.914	0.901	0.906	0.874	0.829	0.580
SGC	0.92	0.905	0.900	0.907	0.900	0.899	0.887
SGC+Contranorm	0.914	0.916	0.917	0.919	0.900	0.885	0.867
GCN+Contranorm	0.790	0.906	0.917	0.912	0.893	0.868	0.578
GCN+DropMessage	0.913	0.91	0.927	0.915	0.894	0.864	0.866
GCNII	0.913	0.926	0.92	0.929	0.922	0.925	0.926
AIR(SGC)	0.895	**0.930**	**0.928**	0.93	0.929	0.918	0.918
NDLS	0.912	0.912	0.902	0.917	0.920	0.891	0.890
GCN+TSC	0.899	0.922	0.927	**0.934**	**0.934**	**0.931**	**0.929**
SGC+TSC	**0.925**	0.922	0.925	0.933	0.929	0.923	0.922

Table 5: Performance comparison results on AmazonPhoto

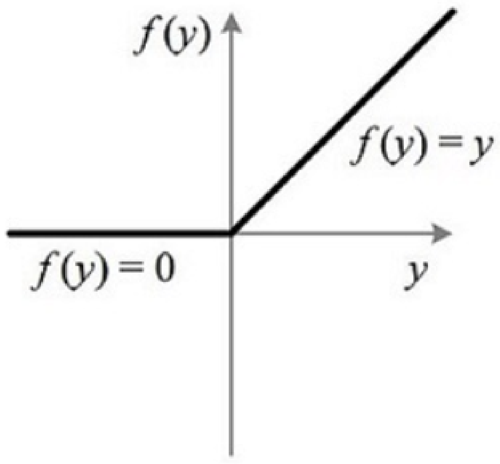
There are indeed many factors affecting the performance of the model, but when the number of layers increases, oversmoothing is its main influence, as we verified in papers

[a1],[a2], and secondly, we also show MAD and T-SNE plots illustrating that the node features are progressively oversmoothed as the number of layers increases.

[a1]Chen M, et al. Simple and deep graph convolutional networks.International conference on machine learning. PMLR, 2020.
[a2]Zhao L,et al. Pairnorm: Tackling oversmoothing in gnns.ICLR, 2019.

W7. over-smoothing technique can not always prevent GNN from performance degradation." Over-smoothing is a phenomenon, not a technique. Not sure what this statement is trying to convey.
Reply: In this context, "oversmoothing techniques" refer to techniques applied to graph neural networks to prevent oversmoothing.

W8. Equation 10 seems incorrect: I believe it should be $H^{(l)} = \sigma(L\sigma(L(\ldots\sigma(LXW^{(1)})W^{(2)}\ldots W^{(l)})$. Regardless, you do not need this if you are using SGCN simplification.
Reply: Properties of Rule functions:



$$ReLU(x) = \begin{cases} x & if x > 0 \\ 0 & if x \leq 0 \end{cases}$$

Also because the normalised Laplace matrix is semi-positive definite, one can write $H^{(l)} = \sigma(L\sigma(L(\ldots\sigma(LXW^{(1)})W^{(2)}\ldots W^{(l)})$ as $H^{(l)} = \sigma(L^{l}\sigma((\ldots\sigma(XW^{(1)})W^{(2)}\ldots W^{(l)})$