DIVERSIFIED MULTISCALE GRAPH LEARNING WITH GRAPH SELF-CORRECTION

Yuzhao Chen
Tsinghua University
{chen-yz19}@mails.tsinghua.edu.cn

Yatao Bian
Tencent AI Lab
{yatao.bian}@gmail.com

Jiying Zhang
Tsinghua University
{zhangjiy20}@mails.tsinghua.edu.cn

Xi Xiao
Tsinghua University
{xiaox}@sz.tsinghua.edu.cn

Tingyang Xu
Tencent AI Lab
{Tingyangxu}@tencent.com

Yu Rong
Tencent AI Lab
{yu.rong}@hotmail.com

ABSTRACT

Though the multiscale graph learning techniques have enabled advanced feature extraction frameworks, we find that the classic ensemble strategy shows inferior performance while encountering the high homogeneity of the learnt representation, which is caused by the nature of existing graph pooling methods. To cope with this issue, we propose a diversified multiscale graph learning model equipped with two core ingredients: a graph self-correction mechanism to generate informative embedded graphs, and a diversity boosting regularizer to achieve a comprehensive characterization of the input graph. The proposed mechanism compensates the pooled graph with the lost information during the graph pooling process by feeding back the estimated residual graph, which serves as a plug-in component for popular graph pooling methods. Meanwhile, pooling methods enhanced with the self-correcting procedure encourage the discrepancy of node embeddings, and thus it contributes to the success of ensemble learning strategy. The proposed regularizer instead enhances the ensemble diversity at the graph-level embeddings by leveraging the interaction among individual classifiers. Extensive experiments on popular graph classification benchmarks show that the approaches lead to significant improvements over state-of-the-art graph pooling methods, and the ensemble multiscale graph learning models achieve superior enhancement.

1 INTRODUCTION

While many GNNs learn graph representation at a fixed scale, the multiscale graph learning has also attracted a surge of interests, for its capability of capturing both fine and coarse graph structures and features. Several advanced feature extraction frameworks have been proposed for multiscale graph learning. Some attempts adopt the encoder-decoder pipeline to embed the input-graph and perform feature updating in the latent coarsest graph (Chen et al., 2017; Liang et al., 2018; Deng et al., 2019). Other works utilize the pyramid architecture to extract multiscale graph features, and perform feature aggregation via skip-connection (Gao & Ji, 2019; Lee et al., 2019), or cross-layer summation fusion (Li et al., 2020).

Orthogonal to various multiscale feature learning techniques, one straightforward and promising strategy for adequately leveraging the extracted graph embeddings is to construct ensembles of multiple individual classifiers in a stacking style, where each classifier receives a graph embedding at a fixed granularity, and predicted logits of all classifiers are then averaged to produce the final prediction. As a generic method, such ensemble models are expected to obtain more reliable predictions. However, a crucial dependency of high diversity among classifiers, may not hold in the scenario.

Table 1: The failing of the naive ensemble strategy (with '+') on enhancing multiscale GNNs.

Method	PROTEINS	NCI1	NCI109
SAGPool	$73.26_{\pm 0.78}$ (-)	$69.88_{\pm 0.82}$ (-)	$70.07_{\pm 0.69}$ (-)
SAGPool+	$72.56_{\pm 0.76}$ (↓)	$69.61_{\pm 0.62}$ (↓)	$69.41_{\pm 0.66}$ (↓)
ASAP	$74.14_{\pm 0.33}$ (-)	$74.27_{\pm 0.63}$ (-)	$72.90_{\pm 0.59}$ (-)
ASAP+	$74.21_{\pm 0.69}$ (↑)	$73.62_{\pm 0.56}$ (↓)	$72.52_{\pm 0.45}$ (↓)

As indicated in recent researches (Mesquita et al., 2020; Bianchi et al., 2020), the indispensable graph pooling operation for establishing multiscale graph learning may have led to the high homogeneity of node embeddings. Mesquita et al. (Mesquita et al., 2020) observe that graph pooling exacerbate the homogeneity of node embeddings. Due to the homogeneity of the node embeddings, the generated graph embeddings at various granularity tend to become homogeneous as well. Under this circumstance, the graph embeddings as inputs lead to limited diversity among learned classifiers. As a result, the ensemble strategy will fail in boosting the performance of GNNs equipped with these pooling modules, which have been verified by the experimental results in Table 1.

To address the issues discussed above, we design a diversified multiscale graph learning model which improve the graph pooling quality from two aspects: 1) Less information loss: we propose a graph self-correction mechanism to generate informative pooled graphs, which also hampers the homogeneity of node embeddings and thus promotes the ensemble diversity at the *node-level embeddings*, and 2) More diversity: we introduce a diversity boosting regularizer to directly model and optimize the ensemble diversity at the *graph-level embeddings*. The ensembled multiscale graph learning framework and the schemata of our approaches is depicted in Figure 1.

2 Graph Self-Correction Mechanism

A core idea of Graph Self-Correction (GSC) mechanism is to reduce such information loss through compensating information generated by the feedback procedures. It contains three phases: graph pooling, compensated information calculation and information feedback.

2.1 Phase 1: Graph pooling

In the settings of GSC, the initial graph pooling layer concentrates on searching for the optimal structure of the coarsened graphs rather than directly learning both the topological and informative embeddings. As a preliminary step of the approach, it can be implemented by any kind of existing pooling methods. As a explanation case, SAGPool (Lee et al., 2019) adopts a 1-layer GCN (Kipf & Welling, 2016) as the node scorer, and uses the Top-K selection strategy to select K nodes with index of $\text{Idx}^{(t)}$ at the t-th pooling layer, constructing the pool coarsened graph. Assumed the input graph has the vertex set $\mathcal{V}$ with N nodes. Its node features and adjacency matrix is $\mathbf{X}^{(t)}$ and $\mathbf{A}^{(t)}$. The ones in the pooled graph is denoted as $\mathbf{X}^{(t+1)}$ and $\mathbf{A}^{(t+1)}$.

2.2 Phase 2: Compensated information calculation

As noticed in the pooling process given by SAGPool, the subsequent pooled node features discard all information preserved in those unselected nodes, which might hurt the exploitation of rich semantic features in original graphs. Inspired by the feedback networks (Carreira et al., 2016; Haris et al., 2018; Li et al., 2016), GSC introduces a residual estimation procedure to calculate compensated information that empowers the self-correction of the embedded coarsened graph in the feedback stage. Specifically, we propose two parallel schemes to calculate such residual signals: 1) complement graph fusion, and 2) coarsened graph back-projection.

1) Complement graph fusion: The graph composited by the unselected nodes after the graph pooling layer and their adjacency relations is defined as the complement of the pooled graph here. Then the indices of nodes in the complement graph formulate: $\text{CompIdx} = \{i(v) | v \in \mathcal{V} \text{ and } i(v) \notin \text{Idx}^{(t)}\}$, where $i(\cdot)$ gives a unique index to a node. The complement denotes the information that has been lost during the first phase of GSC, and it can be adopted as the residual signal to be fused with the

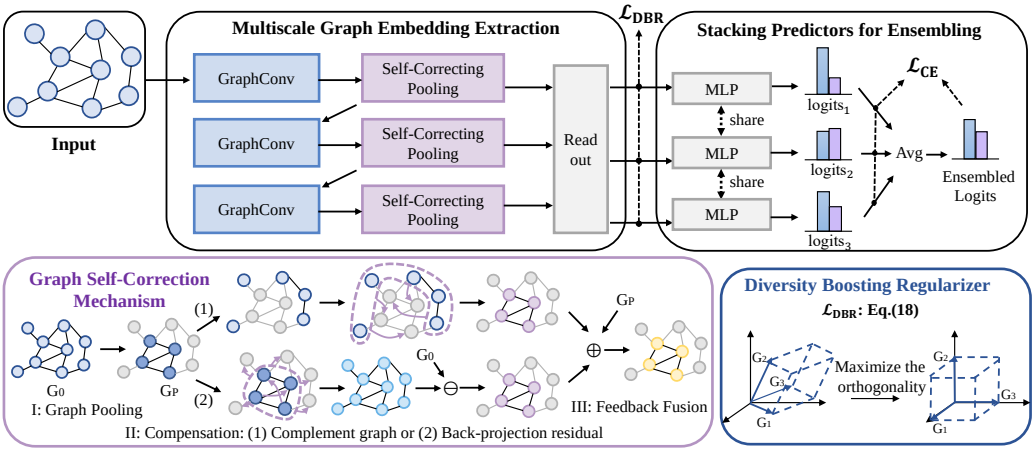


Figure 1: The ensembled multiscale graph learning framework (top), equipped with the proposed graph self-correction mechanism (bottom left) and diversity boosting regularizer (bottom right). The bottom-left module illustrates the two modes of graph self-correction, and the bottom-right module provides a geometric interpretation of the diversity regularizer.

pooled graph. The approach is to propagate node features from the complement graph to the pooled graph leveraging an UnPool layer: $\mathbf{E}^{(t)}_{\text{Comp}} = \mathrm{UnPool}(\mathbf{X}^{(t)}_{\text{Comp}}, \mathbf{A}^{(t)})$, where the UnPool denotes an unpooling process. It receives node features of the complement graph as the input feature and the original graph structure as adjacency relation, and interpolates compensated information to those selected nodes of the pooled graph. Similar to (Gao & Ji, 2019), it can be devised by initializing the input feature matrix $\mathbf{X}^{(t)}_{\text{Comp}} \in \mathbb{R}^{N\times d}$ as:

$$\mathbf{X}^{(t)}_{\text{Comp}}[i(v), :] = \begin{cases} \mathbf{X}^{(t)}_{i(v),:} & i(v) \in \text{CompIdx} \\ \mathbf{0} & \text{otherwise,} \end{cases} \tag{1}$$

here we denote the index selection operator in the brackets of the left-hand side for clarity, and then UnPool propagate it by a message passing process, which is implemented by a 1-layer GCN.

2) Coarsened graph back-projection: Another source of the compensated information is inspired by the classic back-projection algorithm (Irani & Peleg, 1991) used for iteratively refining the restored high-resolution images. The intuition is that a pooled graph is ideal if it contains adequate information for reconstructing the original graph. Rather than introducing additional knowledge to model the pattern of lost information, coarsened graph back-projection manages to restore the original graph using the pooled node features themselves, and then calculates the reconstruction error to serve as the compensated signals. The restoration process is given by: $\mathbf{X}^{(t)}_{\text{Recon}} = \mathrm{UnPool}(\mathbf{X}^{(t)}_{\text{Coarse}}, \mathbf{A}^{(t)})$, where the input feature matrix of $\mathbf{X}^{(t)}_{\text{Coarse}} \in \mathbb{R}^{N\times d}$ is initialized by the pooled node features $\mathbf{X}^{(t+1)} \in \mathbb{R}^{|\text{Idx}|\times d}$ with padding zero vectors:

$$\mathbf{X}^{(t)}_{\text{Coarse}}[i(v), :] = \begin{cases} \mathbf{X}^{(t+1)}_{i(v),:} & i(v) \in \text{Idx} \\ \mathbf{0} & \text{otherwise,} \end{cases} \tag{2}$$

After that, the residual graph is calculated by: $\mathbf{E}^{(t)}_{\text{Recon}} = \mathbf{X}^{(l_t,P_t)} - \mathbf{X}^{(t)}_{\text{Recon}}$.

2.3 Phase 3: Information Feedback

GSC finally refines the pooled graph $\mathbf{X}^{(t+1)}$ with the residual embedded graph of $\mathbf{E}^{(t)} \in \mathbb{R}^{N\times d}$ as: $\tilde{\mathbf{X}}^{(t+1)} = \mathbf{X}^{(t+1)} + \mathbf{E}^{(t)}_{\text{Idx},:}$, where $\mathbf{E}^{(t)}_{\text{Idx},:} \in \mathbb{R}^{|\text{Idx}|\times d}$ is tailored from the residual signal, either $\mathbf{E}^{(t)}_{\text{Comp}} \in \mathbb{R}^{N\times d}$ or $\mathbf{E}^{(t)}_{\text{Recon}} \in \mathbb{R}^{N\times d}$, using the indices $\text{Idx}^{(t)}$.

Table 2: Performance comparison of GSC mechanism with other pooling methods, and its effect on promoting the ensemble strategy. Methods denoted with '+' are the ensembled models. Best result on each comparison is bolded. Best result of the non-ensembled models is underlined.

Dataset	D&D	PROTEINS	NCI1	NCI109	FRANKENSTEIN	OGB-MOLHIV
#Graphs	1178	1113	4110	4127	4337	41127
Avg #Nodes	284.3	39.1	29.9	29.7	16.9	25.5
Set2Set	71.60 ± 0.87	72.16 ± 0.43	66.97 ± 0.74	61.04 ± 2.69	61.46 ± 0.47	–
GlobalAttention	71.38 ± 0.78	71.87 ± 0.60	69.00 ± 0.49	67.87 ± 0.40	61.31 ± 0.41	–
SortPool	71.87 ± 0.96	73.91 ± 0.72	68.74 ± 1.07	68.59 ± 0.67	63.44 ± 0.65	–
DiffPool	66.95 ± 2.41	68.20 ± 2.02	62.23 ± 1.90	61.98 ± 1.98	60.60 ± 1.62	–
TopK	75.01 ± 0.86	71.10 ± 0.90	67.02 ± 2.25	66.12 ± 1.60	61.46 ± 0.84	–
SAGPool	75.88 ± 0.72	73.26 ± 0.78	69.88 ± 0.82	70.07 ± 0.69	60.68 ± 0.49	73.16 ± 2.3
w/ GSC-B	$\underline{76.03} \pm 0.68$ (-)	74.27 ± 0.80 (-)	71.91 ± 0.93 (-)	71.69 ± 0.70 (-)	61.85 ± 0.79 (-)	73.53 ± 2.6 (-)
w/ GSC-B+	76.49 ± 0.96 (↑)	74.47 ± 0.72 (↑)	$\mathbf{73.10} \pm 0.69$ (↑)	72.13 ± 0.69 (↑)	62.54 ± 0.47 (↑)	72.73 ± 2.1 (↑)
w/ GSC-C	76.02 ± 0.67 (-)	$\underline{74.36} \pm 0.66$ (-)	$\underline{72.38} \pm 0.69$ (-)	$\underline{72.03} \pm 0.70$ (-)	$\underline{63.19} \pm 0.62$ (-)	$\underline{73.87} \pm 1.6$ (-)
w/ GSC-C+	$\mathbf{76.57} \pm 1.04$ (↑)	$\mathbf{74.89} \pm 0.70$ (↑)	72.77 ± 0.73 (↑)	$\mathbf{72.76} \pm 0.57$ (↑)	$\mathbf{63.65} \pm 0.54$ (↑)	$\mathbf{74.01} \pm 1.6$ (↑)
ASAP	76.77 ± 0.58	74.14 ± 0.33	74.27 ± 0.63	72.90 ± 0.59	64.58 ± 0.23	73.41 ± 2.2
w/ GSC-B	76.80 ± 0.54 (-)	74.39 ± 0.71 (-)	74.52 ± 0.69 (-)	$\underline{73.64} \pm 0.49$ (-)	64.34 ± 0.45 (-)	$\underline{74.99} \pm 1.7$ (-)
w/ GSC-B+	$\mathbf{77.46} \pm 0.60$ (↑)	74.86 ± 0.51 (↑)	$\mathbf{74.93} \pm 0.67$ (↑)	$\mathbf{74.32} \pm 0.47$ (↑)	65.01 ± 0.65 (↑)	$\mathbf{75.31} \pm 1.5$ (↑)
w/ GSC-C	$\underline{77.30} \pm 0.72$ (-)	$\underline{74.77} \pm 0.49$ (-)	$\underline{74.83} \pm 0.41$ (-)	73.37 ± 0.58 (-)	$\underline{65.90} \pm 0.32$ (-)	74.08 ± 1.8 (-)
w/ GSC-C+	77.35 ± 0.53 (↑)	$\mathbf{75.15} \pm 0.60$ (↑)	74.84 ± 0.60 (↑)	73.57 ± 0.45 (↑)	$\mathbf{66.17} \pm 0.68$ (↑)	74.76 ± 1.4 (↑)

3 Diversity Boosting Regularizers

The collection of graph embeddings with T scales formulates $\mathbf{G} = (\mathbf{x}_G^{(1)}, \cdots, \mathbf{x}_G^{(T)})^{\mathrm{T}} \in \mathbb{R}^{T \times d}$, where each element $\mathbf{x}_G^{(t)} \in \mathbb{R}^{1 \times d}$ is readout from $\mathbf{X}^{(t)}$. Motivated by the theory of determinant point processes (DPPs) (Kulesza & Taskar, 2012), we define the diversity of multiscale graph embeddings as $DoG = \det(\mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the similarity matrix of the multiscale graph embeddings. For example, it could be the gram matrix, where each entry represents the inner-product similarity score of a pair of graph embeddings $\mathbf{\Sigma} = \tilde{\mathbf{G}}\tilde{\mathbf{G}}^{\mathrm{T}}$, where $\tilde{\mathbf{G}}$ is row-wise normalized from $\mathbf{G}$, for guaranteeing the property of positive semi-definiteness on $\mathbf{\Sigma}$. According to the matrix theory (Zhang, 2011), the value of $\det(\mathbf{G}\mathbf{G}^{\mathrm{T}})$ equals to the squared volume of subspace spanned by graph embeddings $\{\mathbf{x}_G^{(t)} | t \in \{1, ..., T\}\}$, and hence DoG reaches the maximum value if and only if the graph embeddings are mutually orthogonal. Noticed that the normalization of graph embeddings would reduce the variance of them, and lead the optimization problem to become trivial for regularizing the networks. We introduce the gaussian kernel to parameterize the similarity matrix, which formulates $\mathbf{\Sigma}_{i,j} = \exp(-\gamma \cdot d^2(\mathbf{G}_i, \mathbf{G}_j))$, where $d(\cdot, \cdot)$ calculates the Euclidean distance and γ is a hyper-parameter to control the flatness of the similarity matrix. Under this definition, we propose the Diversity Boosting Regularizer (DBR) to further diversify the multiscale graph embeddings,

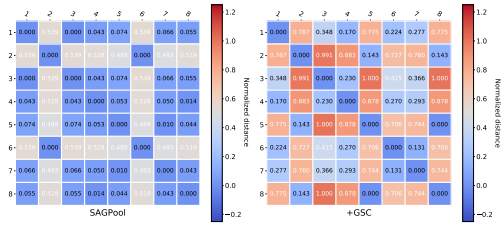
$$L_{\text{DBR}}(\mathbf{G}) = -\log(DoG) + \log(\det(\mathbf{\Sigma} + \mathbf{I})), \tag{3}$$

where the first term is the logarithm of embeddings diversity, the second term as normalization controls the magnitude of the similarity matrix. The calculations of DBR is efficient since the pooling number T grows much slower with the scale of problem. Lastly, we combine the regularizer $\alpha \cdot L_{\text{DBR}}$ (α as loss weight) with the summed cross-entropy loss $\Sigma_T L_{\text{CE}}$ over all predicted logits as the training objective function.

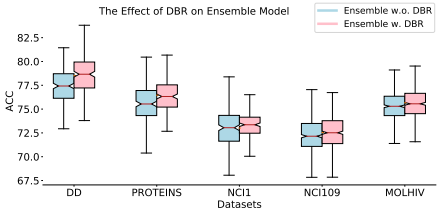
4 Experiments

We compare the approach with previous graph learning methods, including hierarchical pooling based models of DiffPool (Ying et al., 2018), TopK (Gao & Ji, 2019), SAGPool (Lee et al., 2019), ASAP (Ranjan et al., 2020), and global pooling based models of Set2Set (Vinyals et al., 2015), GlobalAttention (Li et al., 2015) and SortPool (Zhang et al., 2018). Among them, we select the two state-of-the-art pooling methods, SAGPool and ASAP as the evaluation baseline backbones to conduct in-depth comparisons. We use the same hyper-parameters search strategy for all the baselines and our method.

We denote GSC that calculates the feedback signal from complement graph as 'GSC-C', and the one from graph back-projection as 'GSC-B'. The overall results are summarized in Table 2. One can observe that the GSC mechanism achieves consistent and considerable performance improvement



(a) Distance heatmap of pooled node features



(b) Evaluation of Diversity Boosting Regularizer

Figure 2: a) Node features become highly homogenized in SAGPool (left), while graph self-correction (GSC) well preserves node discrepancy (right) from the initial features. b) Performance of ensemble models training with (pink) and without (lightblue) the regularizer.

for all the cases. It improves around 1.5% accuracy and 1.0% accuracy over six benchmarks on the baseline of SAGPool and ASAP, respectively. Generally, the complement graph fusion ('GSC-C') performs better than the back projection ('GSC-B') on the SAGPool method, because the complement graph provides the pooled graph with richer information of the semantic representations of unselected nodes. Instead, in the cases of NCI109 and MOLHIV on ASAP, GSC based on back projection ('GSC-B') achieves superior performance, which might indicate that the reconstruction residuals have well refined the updating pooled node features.

We particularly compare the ensemble performance between with and without graph self-correction mechanism on the pooling process. The results are separately given in Table 2 and Table 1. Table 1 illustrates the failure of ensemble strategy on boosting multiscale GNNs built on baseline pooling modules. The homogeneity of node features of standard pooling methods is shown in Figure 2 (a). In contrast, GSC could relieve such homogeneity and further contribute to the success of the ensemble strategy. For the multiscale GNNs equipped with GSC procedure ('GSC-C' or 'GSC-B'), the ensemble learning ('GSC-C+' or 'GSC-B+') successfully achieves over 0.5% average accuracy improvement on the even stronger baselines.

We conduct a comparison study on the ensemble multiscale GNNs equipped with GSC, under the difference between training with and without the proposed DBR. The results are displayed in Figure 2 (b), which verify that DBR can jointly improve the ensemble performance with the GSC mechanism. One can see that DBR achieves smaller standard deviation under the 10-fold cross validation, which means the diversified graph embeddings indeed provide a more comprehensive characterization of the input graph, and thus improve the training stability.

REFERENCES

Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *International Conference on Machine Learning*, pp. 874–883. PMLR, 2020.

Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4733–4742, 2016.

Haochen Chen, Bryan Perozzi, Yifan Hu, and Steven Skiena. Harp: Hierarchical representation learning for networks. *arXiv preprint arXiv:1706.07845*, 2017.

Chenhui Deng, Zhiqiang Zhao, Yongyu Wang, Zhiru Zhang, and Zhuo Feng. Graphzoom: A multi-level spectral approach for accurate and scalable graph embedding. In *International Conference on Learning Representations*, 2019.

Hongyang Gao and Shuiwang Ji. Graph u-nets. *arXiv preprint arXiv:1905.05178*, 2019.

Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1664–1673, 2018.

Jingjia Huang, Zhangheng Li, Nannan Li, Shan Liu, and Ge Li. Attpool: Towards hierarchical feature representation in graph convolutional networks via attention mechanism. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6480–6489, 2019.

Michal Irani and Shmuel Peleg. Improving resolution by image registration. *CVGIP: Graphical models and image processing*, 53(3):231–239, 1991.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *arXiv preprint arXiv:1207.6083*, 2012.

Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. *arXiv preprint arXiv:1904.08082*, 2019.

Ke Li, Bharath Hariharan, and Jitendra Malik. Iterative instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3659–3667, 2016.

Maosen Li, Siheng Chen, Ya Zhang, and Ivor W Tsang. Graph cross networks with vertex infomax pooling. *arXiv preprint arXiv:2010.01804*, 2020.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

Jiongqian Liang, Saket Gurukar, and Srinivasan Parthasarathy. Mile: A multi-level framework for scalable graph embedding. *arXiv preprint arXiv:1802.09612*, 2018.

Diego Mesquita, Amauri Souza, and Samuel Kaski. Rethinking pooling in graph neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.

Ekagra Ranjan, Soumya Sanyal, and Partha P Talukdar. Asap: Adaptive structure aware pooling for learning hierarchical graph representations. In *AAAI*, pp. 5470–5477, 2020.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Advances in neural information processing systems*, pp. 4800–4810, 2018.

Fuzhen Zhang. *Matrix theory: basic results and techniques*. Springer Science & Business Media, 2011.

Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

A THE HOMOGENIZATION OF NODE FEATURES DURING POOLING

Figure 3 shows the phenomenons that the nodes become severely homogenized after the second graph pooling process with SAGPool (Lee et al., 2019), while the GSC mechanism helps relieve this issue. The visualization analysis has provided a deeper insight into the beneficial effect of graph self-correction on the ensemble multiscale learning, that is, promoting the preserving of nodes discrepancy in the coarse graph via restoring the fine graph, which further helps to increase the diversity of input information of the ensemble model.

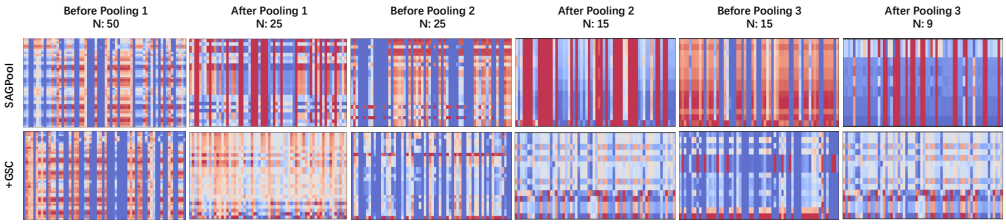


Figure 3: Visualization of node embeddings of a graph before and after pooling layers on PROTEINS. Visualization maps are stretched to be vertically aligned, and a node is represented by a row. The number of nodes for each embedding matrix is given above the figure. On the SAGPool baseline, the high homogeneity of node embeddings is observed (the same as findings in (Mesquita et al., 2020)), meaning the graph pooling process exacerbate the information loss. The GSC instead preserves more diversified representation patterns from the original features.

Table 3: Test accuracy comparison of the ensemble strategy under different evaluation protocols.

Dataset: NCI1	Select-on-Validation	Select-on-Test
SAGPool	69.88 (-)	70.85 (-)
SAGPool+	69.61 (↓)	72.00 (↑)

B Evaluation protocol for pooling-based methods

A fair evaluation protocol is crucial for evaluating the performance of ensemble-based models. Since the ensemble strategy aims at making generalized predictions, the ensemble model with good ability would show superior performance than the individual base learners while the distribution of validation set is inconsistent with test set. However, if one performs model selection on the testset (Xu et al., 2018; Huang et al., 2019; Bianchi et al., 2020; Li et al., 2020), it could result in a misleading conclusion about the performance of ensemble strategy. This could be verified by the experimental example provided in Table 3.

C Hyper-parameter search strategy

For all the experiments, we use Adam optimizer to train the model for 200 epochs in total, with learning rate decay of 0.5 after every 50 epochs. We follow previous works to perform hyper-parameter search for each experiment, and the search range is given by Table 4 (a). For the γ in the gaussian kernel and α the loss weight of DBR, the search range is provided in Table 4 (b). Specially, we fix the γ for the experiments on MOLHIV as 0.003 due to the statistical difference of the data ego-features.

Table 4: Hyper-parameter search range

(a) Hyper-parameter search range for multiscale graph learning model.

Hyper-parameter	Range
Learning rate	$\{0.01, 0.005, 0.001\}$
Hidden dimension	$\{16, 32, 64, 128\}$
Pooling ratio	$\{0.25, 0.5\}$

(b) Hyper-parameter search range for diversity boosting regularizer.

Hyper-parameter	Range
γ of gaussian kernel	$\{1, 5, 10\}$
α of loss weight	$\{0.1, 1, 10\}$