Suit the Node Pair to the Case: A Multi-Scale Node Pair Grouping Strategy for Graph-MLP Distillation

Rui Dong[1], **Jiaxing Li**[1], **Weihuang Zheng**[1], **Youyong Kong**[1,2*]
[1] Jiangsu Provincial Joint International Research Laboratory of Medical Information Processing, School of Computer Science and Engineering, Southeast University
[2] Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications (Southeast University), Ministry of Education, China
{dongrui_0427, jiaxing_li, zhengweihuang, kongyouyong}@seu.edu.cn

Abstract

Graph Neural Network (GNN) is powerful in solving various graph-related tasks, while its message passing mechanism may lead to latency during inference time. Multi-Layer-Perceptron (MLP) can achieve fast inference speed but with limited performance. One solution to fill this gap is through Knowledge Distillation. However, current distillation methods follow a "node-to-node" paradigm, while considering the complex relationships between different node pairs, direct distillation fails to capture these multiple-granularity features in GNN. Furthermore, current methods which focuses on the alignment of logits between teacher and student ignores further learning within layers inside MLP. Therefore, in this paper, we introduce a multi-scale knowledge distillation method (**MSN-GDM**) aiming to capture multiple knowledge from GNN to MLP. We firstly propose a multi-scale node-pair grouping strategy to assign node pairs to different-scale groups according to node pair similarity metrics. The similarity metrics considers both node features and topological structures of the given node pair. Then based on the preprocessed node pair set groups, we design a multi-scale distillation method that can capture comprehensive knowledge in the corresponding groups. The hierarchical weighted sum of each layer is applied as the final output. Extensive experiments on eight real-world datasets demonstrate the effectiveness of our proposed method.

1 Introduction

Graph, as an abstract data structure, is currently widely applied in various real-world applications, social network analysis, recommendation system, neuroscience, etc [Qiu *et al.*, 2018; Wu *et al.*, 2022; Ye *et al.*, 2024]. Recently, the appearance of Graph Neural Networks (GNNs) has provided efficient solutions to graph-related tasks. Based on message passing mechanism where node can capture features from its neighborhood, GNNs are advantageous in capturing graph representations, and semi-supervised learning has witnessed a success in node-level classification task [Kipf and Welling, 2016]. However, when considering practical deployment, there exist certain gaps. One gap lies in its neighborhood-fetching mechanism, where the explicit utilization of structure information leads to latency during reference time. In contrast, Multi-Layer-Perceptrons (MLPs) achieve higher inference speed, but with poorer performance. One way to bridge its gap is through Knowledge Distillation, a model compression method aiming to transfer the knowledge from one complex teacher model to lightweight student model [Hinton *et al.*, 2015; Joshi *et al.*, 2022].

Graph Knowledge Distillation is useful in bridging GNN and MLP [Liu *et al.*, 2023; Tian *et al.*, 2023; Gou *et al.*, 2021]. Previous studies have explored different distillation frameworks from GNNs to MLPs. GLNN [Zhang *et al.*, 2021] followed the traditional distillation paradigm between teacher and student models, leading to better performance and fast inference speed. [Huo *et al.*, 2023] distills graph information from two aspects: feature and structure, which applies data augmentation during training. BGNN [Guo *et al.*, 2023] improves the performance of MLPs by boosting multiple teacher GNNs. KRD [Wu *et al.*, 2023b] is a framework which quantifies the knowledge and can make reliable distillation. NOSMOG [Tian *et al.*, 2022] can learn better structure information and is robust to feature noise. VQGraph [Yang *et al.*, 2024] proposed a VQ-VAE-based framework that can facilitate better structure-aware knowledge distillation. SimMLP [Wang *et al.*, 2024] revisited distillation from mutual information and introduced a "pretrain-finetune" framework without teacher's supervision.

However, these researches have neglected following aspects: Most distillation methods just follow a direct "node-to-node" fashion, which directly aligns logits between teacher nodes and corresponding student nodes. While for graph data, information within GNN may be more complicated, especially considering the complex topological structure of graph, where the relationship between one node and its neighbors can be various, and simple distillation is not capable of capturing these comprehensive knowledge. According to [Wu *et al.*, 2023a], GNN may carry two types of information from spectral perspective, and GNN's message pass-

*Corresponding author

ing mechanism promotes the similarity of representation between nodes and its neighbors. Thus, traditional "node-to-node" distillation may fail to transfer finer information at high frequency which is drown by low frequency. However, its FF-G2M method is a global solution for all nodes, ignoring difference in local structure for different node pairs in one graph. Specially, for one given node, its multiple relationships with different neighborhoods reflect features at different scales (shown in Fig. 1), and different distillation methods are required to capture corresponding different information. Moreover, compared to message passing mechanism for layers within GNN, the fully connected layers inside MLP just ignore the learning of graph structure. Current distillation methods just focus on alignment between different logits, ignoring the forward process within inside MLP layers.

To address the weakness mentioned above, we design a novel GNN-MLP distillation method. The motivation is that *the relationships between one node and its neighborhoods can be multiple and different relationships among node pairs require different methods to extract corresponding information*. The model we proposed, namely **M**ulti-**S**cale **N**ode-pair grouping **G**raph-**D**istillation-**M**LP (**MSN-GDM**), groups the node pairs at different scales and uses corresponding optim functions to capture multi-scale information. The main contribution of this paper is summarized as follows:

- We propose a multi-scale node pair grouping strategy for preprocess. Nodes are assigned to different groups according to the similarity metric between node pairs. The metrics calculation considers both node features and their structures.
- We design corresponding multi-scale distillation-optimization loss based on multi-scale node pair set groups in the previous step. The loss consists of intra-layer loss and inter-layer loss.
- Extensive experiments on eight real-world datasets demonstrate the effectiveness of our distillation method.

2 Related Work

2.1 Graph Neural Network

Recent studies have shown the superior performance of Graph neural networks (GNNs). Based on the Spectral Graph Theory, the original GNN captured the features of the graph from a spectral perspective [Bojchevski and Günnemann, 2017]. [Kipf and Welling, 2016] proposed a semi-supervised framework (**GCN**) which follows the paradigm of spatial domain by aggregating the neighborhood nodes and updating target nodes. Through this message passing mechanism, spatial GNN outperforms spectral GNN in its fast training time while making good use of graph structure. Recently, increasing GNN-based methods are dedicated to proposing novel architectures for better graph representation learning in spatial domain. **GAT** adds attention module for better aggregation [Veličković *et al.*, 2017], **GraphSAGE** designs an inductive framework that can generate embeddings for unseen nodes [Hamilton *et al.*, 2017].

2.2 Knowledge Distillation

Knowledge Distillation (KD) is a method to compress models at a large scale, through which the knowledge is transferred from a large-scale teacher model to a lightweight student model, thereby imporoving the performance of student model while keeping its low complexity of time and space. Current graph knowledge distillation methods can be divided into three types [Gou *et al.*, 2021; Tian *et al.*, 2023]: **GNN-GNN** distillation that intends to decrease the layers and dimensions of teacher GNNs [Yang *et al.*, 2020; Yan *et al.*, 2020]; **GNN-MLP** distillation that aims to transfer knowledge from GNNs to MLPs [Wu *et al.*, 2022; Zhang *et al.*, 2021; Wu *et al.*, 2023b; Yang *et al.*, 2024; Wang *et al.*, 2024] and **Self-Distillation** which refines the transfer of knowledge inside one single model [Chen *et al.*, 2021; Zhang *et al.*, 2023; Deng and Zhang, 2021; Wu *et al.*, 2024].

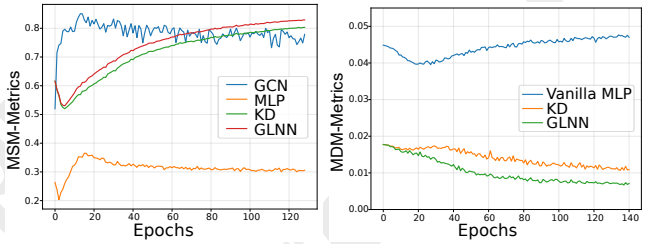


Figure 1: Different knowledge metrics on Cora dataset: MSM (the larger, the better), MDM (the lower, the better).

3 Preliminaries

Graph: We denote a Graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which stands for its two components: nodes and edges. Each node v_i has a d-dimension feature x_i and the corresponding feature matrix of $\mathcal{G}$ is $X \in \mathbb{R}^{N \times d}$. For node-level task, each node has a class label y_i, and for graph-level task, the label $y_{\mathcal{G}}$ is for the whole graph.

Graph Neural Network: For most graph neural networks (GNNs), they all follow message passing mechanism which aggregates its neighborhood nodes before updating the value of its node v_i [Kipf and Welling, 2016; Veličković *et al.*, 2017; Hamilton *et al.*, 2017].

$$h_i^{l+1} = \textbf{UPDATE}(h_i^l,\ \textbf{AGG}(\{h_j^l | j \in \mathcal{N}_i\})) \qquad (1)$$

Where h_i^l is the representation for node v_i in l-th layer, **AGG**(.) stands for the aggregation of its neighborhood nodes $\mathcal{N}_i$, and **UPDATE**(.) means the update operation of h_i.

Knowledge Distillation: Knowledge Distillation aims to transfer the knowledge from a cumbersome teacher model to a lightweight student model [Hinton *et al.*, 2015]. Knowledge Distillation for graphs mainly takes the form of imposing KL-divergence between the soft labels of teacher and student models. The distillation loss function can be defined as follows:

$$\mathcal{L}_{KD} = \tau^2 \mathcal{D}_{KL}(\text{softmax}(z^T/\tau), \text{softmax}(h^S/\tau)) \qquad (2)$$

Where z_T and h_S means the soft labels of teacher and student respectively, τ is the temperature which controls the process of distillation.

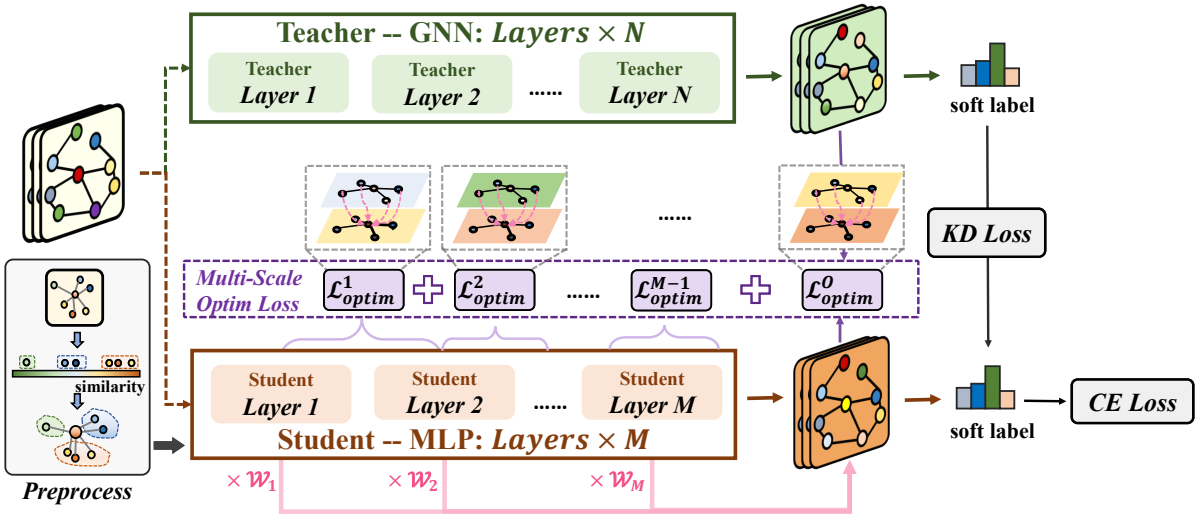


Figure 2: The overall framework of our proposed MSN-GDM. We define $L = 3$ for illustration, which stands for the number of student model layers. We preprocess the dataset by dividing different nodes to corresponding multi-scale node pair groups based on our similarity metric. Based on the preprocessed node pair groups, a multi-scale optim loss is designed to distill multi-scale knowledge.

4 Methods

This section focuses on details of the proposed **MSN-GDM**. Before formal introduction, we first make an analysis of measurement of different knowledge in GNNs. The framework of **MSN-GDM** is shown in Fig. 2, which is composed of two steps: In **step 1**, we preprocess the graph by grouping different neighbor node pairs of each node v_i based on our node similarity metric. Then in **step 2**, a multi-scale optimization function is designed according to mean value of corresponding node pair set groups. The weighted sum of each layer will be the final output for MLP to capture comprehensive representations. More details will be presented in the following subsections.

4.1 Measuring Different Knowledge in GNN

According to [Wu *et al.*, 2022], a trained GNN contains both low and high frequency features, while simple distillation method only captures low-frequency information, ignoring high-frequency information, which can be viewed as the difference between two nodes spatially. Two ways are proposed to measure different knowledge in GNN. The distance between connected nodes can be utilized as low-frequency measurement and the pairwise distance differences between teacher GNN and student MLP shows the capability for MLP to capture fine-grained information.

In general, the knowledge can be divided into two parts. Here we define these two types of knowledge in Eq. 3.

$$\begin{cases} \textbf{MSM}(\mathcal{G}) = \dfrac{1}{|\mathcal{E}|} \displaystyle\sum_{(i,j)\in\mathcal{E}} \dfrac{x_i^T \cdot x_j}{|x_i| \cdot |x_j|} \\ \textbf{MDM}(\mathcal{G}) = \dfrac{1}{|\mathcal{E}|} \displaystyle\sum_{(i,j)\in\mathcal{E}} \left\| |z_j^S - z_i^S| - |z_j^T - z_i^T| \right\|^2 \end{cases} \tag{3}$$

We use cosine similarity to measure low-frequency knowledge in GNN which represents similarities between node pairs (Mean-Sim-Metric, **MSM**), and use **MSE**(.) between teacher and student logits to measure those fine-grained information , which shows the student model's ability to keep its distance between two nodes (Mean-Differ-Metric, **MDM**). The following plots are shown in Fig. 1.

According to the Mean-Sim-Metric figure in Fig. 1, the MSM value of vanilla MLP is the lowest, showing its weakness in capturing graph information, while the value of GCN and its distilling student MLP increases from the beginning and converges to a certain value during the training process [Chen *et al.*, 2020a; Keriven, 2022; Chen *et al.*, 2021]. The reason lies in GNN's message passing mechanism, which finally minimizes the distance between one node and its neighbors. Current "node-to-node" fashion just ignores the difference in local features of graph, where the neighbors of one node may have different features. In other word, a multi-scale distillation is necessary which is tailored for capturing features at different scales.

4.2 Multi-Scale Node Pair Grouping Strategy

According to our motivation, different node pairs contain different feature patterns, and we hope to tell if the node pair shares similar patterns or different ones before designing corresponding operations. Thus, we propose a multi-scale node pair grouping strategy to assign node pairs into different groups. The grouping strategy consists of two parts, namely **node pair similarity metric calculation** and **multi-scale node pair grouping**, and the whole process is shown in Fig. 3.

Node pair similarity metric calculation. Considering that the two attributes of the graph: feature and structure, together make the representation, here we define feature similarity (φ_f) and structure similarity (φ_s) accordingly. For feature similarity, we utilize cosine similarity between given node pair (v_i, v_j). For structure similarity, intuitively, the representation of a node can also be influenced by its neighbors [Xing *et al.*, 2024]. Nodes with more "similar" neighbors

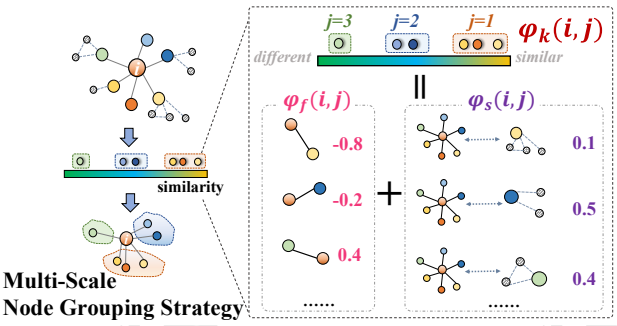


Figure 3: Overall framework of Multi-Scale Node Pair Grouping strategy. We define $J = 3$ and values of node similarities just for illustration.

tend to maintain their own features while nodes with more "different" neighbors are more likely to be influenced. Thus, we consider the neighbor nodes for given v_i (denote as $\mathcal{N}_i$). We normalize the mean value of feature similarity for v_i:

$$\mathcal{H}_i = \frac{1}{\deg_i} \sum_{j \in \mathcal{N}_i \cup i} \frac{x_i^T \cdot x_j}{|x_i| \cdot |x_j|} \tag{4}$$

Here, the term $\deg_i$ refers to the degree of node v_i. Based on above definitions, φ_f and φ_s for node pair (v_i, v_j) are as follows:

$$\begin{cases} \varphi_f(i,j) & = -\frac{x_i^T \cdot x_j}{|x_i| \cdot |x_j|} \\ \varphi_s(i,j) & = \|\mathcal{H}_i - \mathcal{H}_j\| \end{cases} \tag{5}$$

Where the $||.||$ operator is the L1 Norm of the variable, and the smaller value means larger similarity between v_i and v_j.

The value of φ_f and φ_s together decide the type of knowledge between (v_i, v_j). We define knowledge similarity φ_k as the combination of the two values. Moreover, encouraged by [Wu *et al.*, 2023a], the abstract value of φ_k should reflect the extent of knowledge (v_i, v_j) carries. So the final value of φ_k is as follows:

$$\begin{aligned} \varphi_k(i,j) &= \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}^T \cdot \begin{bmatrix} \varphi_{f_{ij}} \\ \varphi_{s_{ij}} \end{bmatrix} + b \\ &= \begin{bmatrix} \alpha \\ 1-\alpha \end{bmatrix}^T \cdot \begin{bmatrix} \varphi_f(i,j) \\ \varphi_s(i,j) \end{bmatrix} + [1-\alpha] \end{aligned} \tag{6}$$

Here α is a hyper-parameter ranging $[0, 1]$, by which the range of φ_k is normalized to $[-1, 1]$, where **smaller value means larger similarity between two nodes**.

Multi-scale node pair grouping. The value of φ_k varies from -1 to 1, and for node v_i and its neighbors $\{j | j \in \mathcal{N}_i\}$, we assign v_j to different groups according to $\varphi_k(i,j)$.

For scale factor J, we firstly divide the interval of [-1,1] equally into J parts, where the range of j-th interval is between $[-1 + \frac{2(j-1)}{J}, -1 + \frac{2j}{J}] \rightarrow [\ell_j, r_j]$. Then we assign node to the correct interval according to its φ_k. As is shown in Eq. 7, for each node v_i, we maintain a J-$scale$ node set (denote as $\mathcal{MS}_i$), where the j-th element contains nodes whose value is among $[\ell_j, r_j)$. Finally, we merge the node set of each node as the multi-scale node pair set group for the given graph $\mathcal{G}$:

$$\begin{cases} \mathcal{MS}_i(j) = \{k | k \in \mathcal{N}_i,\ \varphi_k(i,k) \in [\ell_j, r_j)\} \\ \mathcal{MS}_{\mathcal{G}}(j) = \bigcup\limits_i \mathcal{MS}_i(j), i \in \mathcal{V} \end{cases} \tag{7}$$

After merging, two connected nodes may appear in the same group. Essentially, the grouping strategy can be viewed as operations on edges. For implementation, we do both calculation and grouping based on edges ($\mathcal{E}$) of the graph. For graphs at larger scale where the expense to directly preprocess whole graph can be extremely great, we randomly sample edges from original $\mathcal{E}$ for scalability. Then we employ above preprocess operation on the edge-scaling graph.

$$\begin{cases} \mathcal{E}' \leftarrow \textbf{sample}(\mathcal{E}, \rho) \\ \mathcal{G}' \leftarrow \mathcal{G}(\mathcal{V}, \mathcal{E}', X) \end{cases} \tag{8}$$

Where ρ is a hyper-parameter controlling number of edges to sample on large-scale graphs. In this paper we set it as 0.3.

4.3 Multi-Scale Distillation-Optimization Loss

According to Eq. 1, the GNN forward process can be regarded as type of "aggregate then update " operation. Encouraged by this, Our optim loss is composed of two parts: **Intra-Layer Loss** within one single layer and **Inter-Layer Loss** between two layers, whose process in shown in Fig. 4.

Intra-Layer Optim Loss. The idea of intra-layer optim loss ($\mathcal{L}_{intra}$) is from graph self distillation method, where a model distills knowledge from itself without participation of teacher model [Hu *et al.*, 2021; Luo *et al.*, 2021]. Instead, model distills knowledge from structure information by certain constraints [Wu *et al.*, 2024].

Further, inspired by message passing mechanism, for MLP model with L layers, the intra layer optim function can be formulated as:

$$\mathcal{L}_{intra} = \frac{1}{L \cdot |\mathcal{E}|} \sum_{l=1}^{L} \sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{N}_i \cup \{i\}} \left\| z_i^l - z_j^l \right\|^2 \tag{9}$$

Where z_i^l means the representation of node v_i at l-th layer. By adding this item, the node may be closer to nodes around, **implicitly utilizing graph structure information in the same layer**.

Inter-Layer Optim Loss. FF-G2M designed two types of loss function to capture low-frequency and high-frequency features respectively. Our inter-layer optim loss derives from its loss function, yet we step further by applying it into multi-scale node groups. Briefly, **we select which type of knowledge to be passed to deeper layer**. Corresponding to our analysis before, we consider two situations: aggregate knowledge distillation ($\mathcal{L}_{AKD}$) for similar node pairs and different knowledge distillation ($\mathcal{L}_{DKD}$) for different node pairs.

For graph $\mathcal{G}$ and the corresponding node pair set group $\mathcal{MS}_{\mathcal{G}}$, the main idea is to **treat different groups differently** according to the overall feature of each group. The mean value of its φ_k (namely $\tilde{\mu_j}$) is calculated for each group ($\mathcal{MS}_{\mathcal{G}}(j)$) firstly. The inter-layer optim loss ($\mathcal{L}_{inter}$) is defined as follows. (see in Eq. 10)

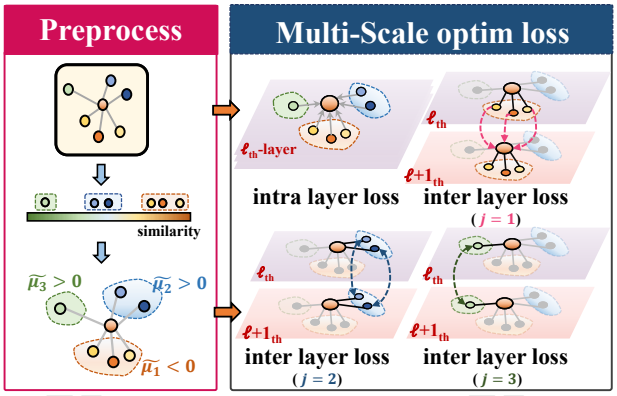


Figure 4: Overall process of Multi-Scale optim loss. (Here $J = 3$ for illustration.)

$$\mathcal{L}^l_{inter_j} = \begin{cases} \mathcal{L}^l_{AKD_j} = \sum_{\substack{i,k \in \mathcal{MS}_{\mathcal{G}}(j), \\ (i,k)\in\mathcal{E}}} \textbf{MSE}(z^l_k, z^{l+1}_i) \cdot |\tilde{\mu_j}|, \tilde{\mu_j} \leq 0 \\ \mathcal{L}^l_{DKD_j} = \sum_{\substack{i,k \in \mathcal{MS}_{\mathcal{G}}(j), \\ (i,k)\in\mathcal{E}}} \textbf{MSE}(\mathcal{K}(z^l_k, z^l_i), \mathcal{K}(z^{l+1}_k, z^{l+1}_i)) \\ \qquad\qquad \cdot |\tilde{\mu_j}|, \tilde{\mu_j} > 0 \end{cases} \tag{10}$$

Here $\mathcal{K}(.)$ works as a kernel function to measure the distance between two representations, and here we realize it with L1 Norm.

For $\mathcal{L}^l_{inter_j}$, which denotes the inter-layer optim loss for *l-th* layer at *j-scale* ($j \in \{1, 2, ..., J\}, l \in \{1, 2, ..., L-1\}$), it considers two cases: If $\tilde{\mu_j}$ is negative, we consider $\mathcal{L}_{AKD}$ which makes the distance closer between node pairs, otherwise $\mathcal{L}_{DKD}$ to keep the difference. The abstract value of $\tilde{\mu_j}$ reflects the extent of knowledge to be distilled.

For the last layer of MLP which also serves as the output layer (z^o), we calculate the optim function between it and the last layer of the teacher model (h^o). The formula is the same as Eq. 10 except for $z^l \to z^o$ and $z^{l+1} \to h^o$ (denote as $\mathcal{L}^o_{inter}$). Each layer's inter-layer loss together makes the final $\mathcal{L}_{inter}$ (shown in Eq. 11).

$$\mathcal{L}_{inter} = \frac{1}{L \cdot J}\left(\sum_{l=1}^{L-1}\sum_{j=1}^{J}\mathcal{L}^l_{inter_j} + \sum_{j=1}^{J}\mathcal{L}^o_{inter_j}\right) \tag{11}$$

4.4 Hierarchical Weighted Sum as MLP Output

The receptive field of GNN may expand as the layer of the model increases and different layers will capture features to different extent [Li *et al.*, 2020; Zhang *et al.*, 2022; Chen *et al.*, 2020b]. Thus, we utilize the weighted sum of each layer as the output of the final layer: $z_o = \sum_{l=1}^{L} w_i z_i$, where $\{w_i\}_i$ are learnable parameters and $\sum_i w_i = 1$.

The final format of the optim loss is the combination of the two items: $\mathcal{L}_{optim} = \mathcal{L}_{intra} + \mathcal{L}_{inter}$. Here we utilize $\mathcal{L}_{optim}$ as an appendix item for the knowledge distillation loss, and the final loss function is the combination of the classification loss, distillation loss and our optim loss, as is shown in Eq. 12, where γ is another hyper-parameter.

$$\mathcal{L} = \mathcal{L}_{CE} + \gamma \cdot (\mathcal{L}_{KD} + \mathcal{L}_{optim}) \tag{12}$$

5 Experiments

5.1 Experiment Setup

Datasets. We conduct our experiments on node-classification task and select eight benchmark datasets: (1) three citation networks: **Cora**, **Pubmed**, **Citeseer** [Bojchevski and Günnemann, 2017]; (2) two co-purchase networks: **Amazon Computers** and **Amazon Photo**; (3) two co-authorship networks: **Coauthor CS** and **Coauthor Physics** [Shchur *et al.*, 2018] (4) one large graph from ogbn dataset: **Ogbn-Arxiv** [Hu *et al.*, 2020].

Baselines. For teacher models, we select three classic spatial GNNs and one spectral GNN. For spatial GNNs, we select **GCN** [Kipf and Welling, 2016], **GAT** [Veličković *et al.*, 2017] and **GraphSAGE** [Hamilton *et al.*, 2017]. We set these models $L = 3, d = 128$ for layer and hidden dimension respectively. For spectral GNN, we select **BernNet** [He *et al.*, 2021]. We also select two classic different distillation methods: **GLNN** [Zhang *et al.*, 2021] and **FF-G2M** [Wu *et al.*, 2023a]. Code implementations of all baseline methods are taken from their respective original papers and the best results of the mentioned distillation methods are recorded.

Experimental Settings. Our task follows the semi-supervised node classification task [Kipf and Welling, 2016] and the splitting ratio of the datasets is $20\%/20\%/60\%$ for train/val/test for all datasets except Arxiv, which follows standard ogbn dataset splitting. As the preprocess of the graph requires information of whole graph, so here all the experiments follow ***semi-supervised transductive*** setting, and the datasets are split randomly and run for 10 times. We calculate their mean accuracy as well as the standard deviation and record the best result. The maximum epochs of each training process is 2000 and we utilize an early stop mechanism with a patience of 300. For ogbn dataset, we rewrite the loss function in Eq. 12 as Eq. 13.

$$\mathcal{L} = (1-\gamma) \cdot \mathcal{L}_{CE} + \gamma \cdot (\mathcal{L}_{KD} + \mathcal{L}_{optim}) \tag{13}$$

For student model, we utilize a 3-layer MLP with a dimension of 64 for all the datasets except Arxiv. For Arxiv the layer is 3 and the dimension is 512. We select the best model by performing a random search on three hyper-parameters in our method: $\alpha \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}, J \in \{2, 3, 4, 5, 6, 7\}, \gamma \in \{0.4, 0.5, 0.6, 0.7\}$. We optimize the model through Adam optimizer. The model is implemented through PYG and all the experiments are conducted on NVIDIA 3090s. Code can be available at https://github.com/KamonRiderDR/MSN-GDM.

5.2 Comparison Results on Different Teachers

The results on different teacher models and different distillation methods are shown in Tab. 1. In general, the proposed MSN-GDM significantly improves the performance of MLP, making it outperform the given teacher models, among which the maximum improvement in absolute accuracy is 9.71% in

Teacher	Methods	Cora	Pubmed	Citeseer	Computers	Photo	CS	Physics
GCN	GCN	81.04 ± 0.05	84.28 ± 0.02	69.51 ± 0.06	87.95 ± 0.43	91.76 ± 0.24	88.70 ± 0.01	93.24 ± 0.39
	MLP	51.83 ± 2.10	82.86 ± 0.71	52.58 ± 2.67	67.61 ± 1.54	78.58 ± 2.32	70.29 ± 2.55	85.53 ± 1.43
	GLNN	83.89 ± 0.21	85.29 ± 0.03	72.70 ± 0.10	88.20 ± 0.02	92.50 ± 0.13	91.72 ± 0.62	94.53 ± 0.01
	FF-G2M	85.06 ± 0.08	86.79 ± 0.13	**74.83 ± 0.07**	87.74 ± 0.07	93.41 ± 0.10	92.15 ± 0.07	94.91 ± 0.11
	MSN-GDM	**86.08 ± 0.11**	**87.82 ± 0.12**	74.75 ± 0.15	**88.94 ± 0.31**	**94.19 ± 0.13**	**93.64 ± 0.11**	**95.79 ± 0.08**
GAT	GAT	81.19 ± 0.02	85.23 ± 0.02	70.13 ± 0.02	88.04 ± 0.09	92.58 ± 0.14	90.21 ± 0.39	93.51 ± 0.12
	MLP	51.83 ± 2.10	82.86 ± 0.71	52.58 ± 2.67	67.61 ± 1.54	78.58 ± 2.32	70.29 ± 2.55	85.53 ± 1.43
	GLNN	84.47 ± 0.11	87.05 ± 0.09	73.22 ± 0.75	87.74 ± 0.05	93.22 ± 0.11	93.03 ± 0.08	94.05 ± 0.01
	FF-G2M	85.22 ± 0.09	87.27 ± 0.03	74.45 ± 0.31	87.96 ± 0.18	93.33 ± 0.04	92.48 ± 0.54	94.48 ± 0.06
	MSN-GDM	**86.13 ± 0.05**	**87.87 ± 0.11**	**76.45 ± 0.28**	**89.04 ± 0.22**	**94.62 ± 0.11**	**93.83 ± 0.10**	**95.14 ± 0.12**
GraphSAGE	GraphSAGE	80.04 ± 0.03	85.62 ± 0.01	68.35 ± 0.02	86.60 ± 0.21	93.23 ± 0.13	92.13 ± 0.01	**93.39 ± 0.01**
	MLP	51.83 ± 2.10	82.86 ± 0.71	52.58 ± 2.67	67.61 ± 1.54	78.58 ± 2.32	70.29 ± 2.55	85.53 ± 1.43
	GLNN	83.34 ± 0.21	86.58 ± 0.36	76.13 ± 0.05	**87.35 ± 0.47**	94.93 ± 0.44	93.70 ± 0.09	90.73 ± 0.07
	FF-G2M	82.87 ± 0.24	86.27 ± 0.02	76.15 ± 0.03	85.28 ± 0.71	94.38 ± 0.24	92.92 ± 0.17	91.13 ± 0.05
	MSN-GDM	**85.95 ± 0.17**	**87.77 ± 0.21**	**78.06 ± 0.02**	87.11 ± 0.35	**95.02 ± 0.14**	**94.50 ± 0.10**	92.27 ± 0.02
BernNet	BernNet	83.84 ± 0.02	87.54 ± 0.01	71.54 ± 0.01	**89.43 ± 0.02**	94.80 ± 0.01	93.88 ± 0.01	95.89 ± 0.01
	MLP	51.83 ± 2.10	82.86 ± 0.71	52.58 ± 2.67	67.61 ± 1.54	78.58 ± 2.32	70.29 ± 2.55	85.53 ± 1.43
	GLNN	86.33 ± 0.33	88.49 ± 0.04	78.11 ± 0.16	87.44 ± 0.25	**95.99 ± 0.05**	94.12 ± 0.06	96.15 ± 0.03
	FF-G2M	86.41 ± 0.03	87.49 ± 0.05	77.59 ± 0.02	88.06 ± 0.25	94.55 ± 0.07	93.65 ± 0.08	96.19 ± 0.05
	MSN-GDM	**87.22 ± 0.10**	**89.05 ± 0.22**	**78.31 ± 0.01**	89.02 ± 0.28	95.63 ± 0.17	**95.95 ± 0.14**	**96.50 ± 0.09**

Table 1: Node classification results on seven real-world datasets following ***transductive*** setting. The mean accuracy and standard deviation on 10 runs are recorded. The **best results** are marked in bold, and second best results are marked underline.

Citeseer with GrapgSAGE as teacher model. Moreover, compared to other Graph-MLP distillation methods, MSN-GDM achieves best or second in all the datasets. Specially, For Cora, Pubmed and CS, MSN-GDM outperforms all the other three models, and the maximum advantages over the second method (GLNN) reaches 2.61% for Cora where GraphSAGE works as the teacher model.

5.3 Comparison Results with Other Distillation Methods

We also conduct comparison with other SOTA distillation methods. We choose GraphSAGE (SAGE) as teacher model. Besides from the two distillation methods above, we also select several different distillation methods: **KRD** [Wu *et al.*, 2023b], **NOSMOG** [Tian *et al.*, 2022], **VQGraph** [Yang *et al.*, 2024] and **SimMLP** [Wang *et al.*, 2024]. The comparision experiments follow their own implementations and the results are listed in Tab. 2. The results show the effectiveness of MSN-GDM, no matter in small graphs (e.g. Cora, Pubmed, Citeseer), medium graphs (e.g. Photo, CS) and large graphs (e.g. Arxiv).

5.4 Ablation Studies

Effects of sub-modules. We first conduct our ablation studies on the two sub-modules of MSN-GDM: hierarchical weight sum (*weight*) and optim loss (*optim*). As is shown in left Fig. 5, the accuracy improves with the addition of each sub-module. Moreover, for most datasets the addition of *optim* sub-module witnesses a larger improvement in accuracy compared to *weight* sub-module. The histogram shows the effectiveness of each sub-module in MSN-GDM.

Effects of different optim loss. Further, as our optim loss consists of two parts namely $\mathcal{L}_{inter}$ and $\mathcal{L}_{intra}$, here we also explore the effects of the two types of loss within $\mathcal{L}_{optim}$, which is shown in right Fig. 5. Discarding either type of loss may lead to a decline in the result, which demonstrates the effectiveness of both $\mathcal{L}_{intra}$ and $\mathcal{L}_{inter}$ inside $\mathcal{L}_{optim}$.

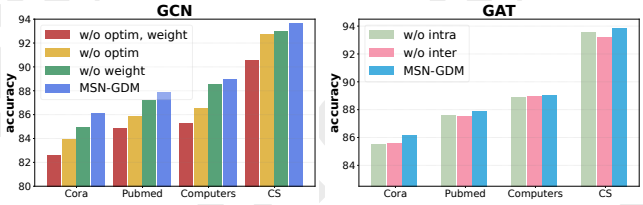


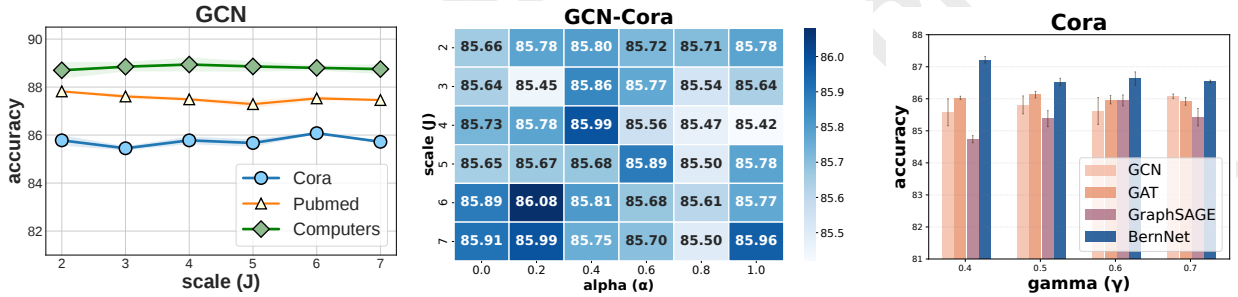
Figure 5: Ablation studies on MSN-GDM. The left is on *weight* and *optim*. The right is on $\mathcal{L}_{intra}$ and $\mathcal{L}_{inter}$ for four datasets.

5.5 Sensitivity Analysis

Firstly, we conduct sensitivity analysis experiment on J (shown in Fig. 6 (a)), where we can see that for different datasets, the value J for best result remains different. For dataset Cora and GCN as teacher, the best value of is 6, while for datasets Computers it is 4. Moreover, for different datasets, the results fluctuate differently when J changes. For further analysis, we plot the heatmap of hyper-parameter J and α, as the final grouping result in preproess part is decided jointly by both two hyper-parameters. For different J, the corresponding optimal α is also different, and heatmap can reflect the relation of the two factors. The results in Fig.

Datasets	SAGE	MLP	GLNN	FF-G2M	KRD	NOSMOG	VQGraph	SimMLP	MSN-GDM	*Impr.*
Cora	80.04 ± 0.03	51.83 ± 2.10	83.34 ± 0.21	82.87 ± 0.24	83.54 ± 0.21	84.10 ± 0.53	84.17 ± 0.38	84.47 ± 0.14	**85.95 ± 0.17**	32.12↑
Pubmed	85.62 ± 0.01	82.86 ± 0.71	86.58 ± 0.36	86.27 ± 0.02	86.55 ± 0.11	86.82 ± 0.39	87.28 ± 0.42	87.19 ± 0.11	**87.77 ± 0.21**	4.91↑
Citeseer	68.35 ± 0.02	52.58 ± 2.67	76.13 ± 0.05	76.15 ± 0.03	77.02 ± 0.07	77.85 ± 0.13	78.03 ± 0.06	77.94 ± 0.10	**78.06 ± 0.02**	25.48↑
Photo	93.23 ± 0.13	78.58 ± 2.32	94.93 ± 0.44	94.38 ± 0.24	94.51 ± 0.37	94.51 ± 0.58	94.80 ± 0.19	94.05 ± 0.33	**95.02 ± 0.14**	16.44↑
CS	92.13 ± 0.01	70.29 ± 2.55	93.70 ± 0.09	92.92 ± 0.17	94.02 ± 0.15	93.92 ± 0.04	94.12 ± 0.11	94.31 ± 0.09	**94.50 ± 0.10**	24.21↑
Arxiv	70.13 ± 0.14	54.73 ± 1.12	67.76 ± 0.51	69.53 ± 0.29	69.11 ± 0.30	71.22 ± 0.15	72.14 ± 0.21	71.95 ± 0.12	**72.48 ± 0.26**	17.75↑

Table 2: Comparison results on different distillation methods on six datasets following ***transductive*** setting. The mean accuracy and standard deviation on 10 runs are recorded. The **best results** are marked in bold, second best marked underline and third best colored in gray. The "*Impr.*" column records the improvement of MSN-GDM on student models.



(a) (b) (c)

Figure 6: Experiment results on different hyper-parameters, with (a) for J, (b) for (α, J) and (c) for γ.

6 (b) show that the best result lies in the middle part of α and J ($\alpha = 0.2, J = 6$). We also conduct the sensitivity analysis experiment on the loss function coefficient γ (shown in Fig. 6 (c)). For Cora, MSN-GDM with different teacher models has best results with different γ, and the results have a smaller variation as γ changes.

Finally we conduct experiments on inference speed and parameters during inference time and results in Tab. 3 show MSN-GDM's effectiveness compared to teacher models.

Models	Cora		Computers	
	param (k)	infer(ms)	param (k)	infer(ms)
GCN	213.13	9.22	130.26	11.66
BernNet	180.16	16.17	97.29	18.46
MSN-GDM	**98.59**↓	**7.03**↓	**57.15**↓	**9.64**↓

Table 3: Parameters/inference time for teacher/student models.

5.6 Empirical Analysis

Finally, we conduct empirical analysis experiment on our method. Corresponding to the first subsection in Methods section, we aim to verify if the student model learned multi-scale knowledge from teacher model. If the mean value $\tilde{\mu_j}$ is negative, then we plot MSM curve (left of Fig. 7) of its nodes pair group, otherwise MDM curve (right of Fig. 7).

From Fig. 7, we can see that **for nodes assigned to "similar" groups** (here the group is when $(j = 2)$), the similarity value increases and is larger than other methods, indicating that for similar node-set group, MSN-GDM enhances the model's ability to capture "similar" patterns; **for nodes assigned to "different" groups** (here $j = 3$), the value decreases gradually and remains the smallest among the four distillation methods, showing the model's advantage in capturing "different" patterns among corresponding group. The figures of 2 cases demonstrate that our MSN-GDM is better at capturing knowledge at different scales.

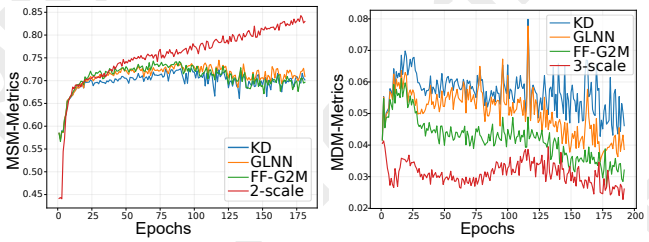


Figure 7: Metrics curves on Pubmed ($J = 4$). For MSM ($j = 2$), the larger, the better. For MDM ($j = 3$), the smaller, the better.

6 Conclusion

In this paper, we propose a GNN-MLP distillation method that captures muti-scale knowledge among different node pairs. We firstly design a multi-scale node pair grouping strategy based on our node pair similarity metric, then a multi-scale optim function is proposed to capture knowledge at different scales. Extensive experiment results demonstrate the effectiveness of our proposed MSN-GDM. For future work, we may explore more adaptive and more efficient grouping methods for better distillation. We may also consider to apply it into inductive setting and more general graph tasks.

Acknowledgments

The work is supported in part by the National Natural Science Foundation of China under Grant No. 62471133, the Central University Basic Research Fund of China under Grant No. 2242024K40020, and the Big Data Computing Center of Southeast University.

References

[Bojchevski and Günnemann, 2017] Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*, 2017.

[Chen *et al.*, 2020a] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3438–3445, 2020.

[Chen *et al.*, 2020b] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International conference on machine learning*, pages 1725–1735. PMLR, 2020.

[Chen *et al.*, 2021] Yuzhao Chen, Yatao Bian, Xi Xiao, Yu Rong, Tingyang Xu, and Junzhou Huang. On self-distilling graph neural network. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2278–2284. International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track.

[Deng and Zhang, 2021] Xiang Deng and Zhongfei Zhang. Graph-free knowledge distillation for graph neural networks. *arXiv preprint arXiv:2105.07519*, 2021.

[Gou *et al.*, 2021] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[Guo *et al.*, 2023] Zhichun Guo, Chunhui Zhang, Yujie Fan, Yijun Tian, Chuxu Zhang, and Nitesh V Chawla. Boosting graph neural networks via adaptive knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7793–7801, 2023.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[He *et al.*, 2021] Mingguo He, Zhewei Wei, Hongteng Xu, et al. Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. *Advances in Neural Information Processing Systems*, 34:14239–14251, 2021.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[Hu *et al.*, 2020] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[Hu *et al.*, 2021] Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wang, Erjin Zhou, and Yue Gao. Graph-mlp: Node classification without message passing in graph. *arXiv preprint arXiv:2106.04051*, 2021.

[Huo *et al.*, 2023] Cuiying Huo, Di Jin, Yawen Li, Dongxiao He, Yu-Bin Yang, and Lingfei Wu. T2-gnn: Graph neural networks for graphs with incomplete features and structure via teacher-student distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4339–4346, 2023.

[Joshi *et al.*, 2022] Chaitanya K Joshi, Fayao Liu, Xu Xun, Jie Lin, and Chuan Sheng Foo. On representation knowledge distillation for graph neural networks. *IEEE transactions on neural networks and learning systems*, 2022.

[Keriven, 2022] Nicolas Keriven. Not too little, not too much: a theoretical analysis of graph (over) smoothing. *Advances in Neural Information Processing Systems*, 35:2268–2281, 2022.

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

[Li *et al.*, 2020] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. Deepergcn: All you need to train deeper gcns. *arXiv preprint arXiv:2006.07739*, 2020.

[Liu *et al.*, 2023] Jing Liu, Tongya Zheng, Guanzheng Zhang, and Qinfen Hao. Graph-based knowledge distillation: A survey and experimental evaluation. *arXiv preprint arXiv:2302.14643*, 2023.

[Luo *et al.*, 2021] Yi Luo, Aiguo Chen, Ke Yan, and Ling Tian. Distilling self-knowledge from contrastive links to classify graph nodes without passing messages. *arXiv preprint arXiv:2106.08541*, 2021.

[Qiu *et al.*, 2018] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2110–2119, 2018.

[Shchur *et al.*, 2018] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

[Tian *et al.*, 2022] Yijun Tian, Chuxu Zhang, Zhichun Guo, Xiangliang Zhang, and Nitesh Chawla. Learning mlps on graphs: A unified view of effectiveness, robustness, and efficiency. In *The Eleventh International Conference on Learning Representations*, 2022.

[Tian *et al.*, 2023] Yijun Tian, Shichao Pei, Xiangliang Zhang, Chuxu Zhang, and Nitesh V Chawla. Knowledge distillation on graphs: A survey. *arXiv preprint arXiv:2302.00219*, 2023.

[Veličković *et al.*, 2017] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and

Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[Wang *et al.*, 2024] Zehong Wang, Zheyuan Zhang, Chuxu Zhang, and Yanfang Ye. Training mlps on graphs without supervision. *arXiv preprint arXiv:2412.03864*, 2024.

[Wu *et al.*, 2022] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[Wu *et al.*, 2023a] Lirong Wu, Haitao Lin, Yufei Huang, Tianyu Fan, and Stan Z Li. Extracting low-/high-frequency knowledge from graph neural networks and injecting it into mlps: An effective gnn-to-mlp distillation framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 10351–10360, 2023.

[Wu *et al.*, 2023b] Lirong Wu, Haitao Lin, Yufei Huang, and Stan Z Li. Quantifying the knowledge in gnns for reliable distillation into mlps. In *International Conference on Machine Learning*, pages 37571–37581. PMLR, 2023.

[Wu *et al.*, 2024] Lirong Wu, Haitao Lin, Zhangyang Gao, Guojiang Zhao, and Stan Z Li. A teacher-free graph knowledge distillation framework with dual self-distillation. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[Xing *et al.*, 2024] Yujie Xing, Xiao Wang, Yibo Li, Hai Huang, and Chuan Shi. Less is more: on the over-globalizing problem in graph transformers. *arXiv preprint arXiv:2405.01102*, 2024.

[Yan *et al.*, 2020] Bencheng Yan, Chaokun Wang, Gaoyang Guo, and Yunkai Lou. Tinygnn: Learning efficient graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1848–1856, 2020.

[Yang *et al.*, 2020] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7074–7083, 2020.

[Yang *et al.*, 2024] Ling Yang, Ye Tian, Minkai Xu, Zhongyi Liu, Shenda Hong, Wei Qu, Wentao Zhang, CUI Bin, Muhan Zhang, and Jure Leskovec. Vqgraph: Rethinking graph representation space for bridging gnns and mlps. In *The Twelfth International Conference on Learning Representations*, 2024.

[Ye *et al.*, 2024] Hongting Ye, Yalu Zheng, Yueying Li, Ke Zhang, Youyong Kong, and Yonggui Yuan. Rh-brainfs: regional heterogeneous multimodal brain networks fusion strategy. *Advances in Neural Information Processing Systems*, 36, 2024.

[Zhang *et al.*, 2021] Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. Graph-less neural networks: Teaching old mlps new tricks via distillation. *arXiv preprint arXiv:2110.08727*, 2021.

[Zhang *et al.*, 2022] Ke Zhang, Xinyan Pu, Jiaxing Li, Jiasong Wu, Huazhong Shu, and Youyong Kong. Hierarchical diffusion scattering graph neural network. In *IJCAI*, pages 3737–3743, 2022.

[Zhang *et al.*, 2023] Hanlin Zhang, Shuai Lin, Weiyang Liu, Pan Zhou, Jian Tang, Xiaodan Liang, and Eric P Xing. Iterative graph self-distillation. *IEEE Transactions on Knowledge and Data Engineering*, 2023.