Learning to Distill Graph Neural Networks

Cheng Yang
Yuxin Guo
Beijing University of Posts and Telecommunications
Beijing, China

Yao Xu
Researcher
Beijing, China

Chuan Shi*
Jiawei Liu
Chunchen Wang
Beijing University of Posts and Telecommunications
Beijing, China

Xin Li
Ning Guo
Researcher
Beijing, China

Hongzhi Yin
The University of Queensland
Brisbane, Australia

ABSTRACT

Graph Neural Networks (GNNs) can effectively capture both the topology and attribute information of a graph, and have been extensively studied in many domains. Recently, there is an emerging trend that equips GNNs with knowledge distillation for better efficiency or effectiveness. However, to the best of our knowledge, existing knowledge distillation methods applied on GNNs all employed predefined distillation processes, which are controlled by several hyper-parameters without any supervision from the performance of distilled models. Such isolation between distillation and evaluation would lead to suboptimal results. In this work, we aim to propose a general knowledge distillation framework that can be applied on any pretrained GNN models to further improve their performance. To address the isolation problem, we propose to parameterize and learn distillation processes suitable for distilling GNNs. Specifically, instead of introducing a unified temperature hyper-parameter as most previous work did, we will learn node-specific distillation temperatures towards better performance of distilled models. We first parameterize each node's temperature by a function of its neighborhood's encodings and predictions, and then design a novel iterative learning process for model distilling and temperature learning. We also introduce a scalable variant of our method to accelerate model training. Experimental results on five benchmark datasets show that our proposed framework can be applied on five popular GNN models and consistently improve their prediction accuracies with 3.12% relative enhancement on average. Besides, the scalable variant enables 8 times faster training speed at the cost of 1% prediction accuracy.

*Corresponding author, shichuan@bupt.edu.cn

WSDM '23, February 27–March 3, 2023, Singapore, Singapore.

ACM ISBN 978-1-4503-9407-9/23/02…$15.00
https://doi.org/10.1145/3539597.3570480

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Networks** → **Network algorithms**.

KEYWORDS

Graph Neural Networks, Knowledge Distillation

ACM Reference Format:
Cheng Yang, Yuxin Guo, Yao Xu, Chuan Shi, Jiawei Liu, Chunchen Wang, Xin Li, Ning Guo, and Hongzhi Yin. 2023. Learning to Distill Graph Neural Networks. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27–March 3, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570480

1 INTRODUCTION

Graph Neural Networks (GNNs) have become the state-of-the-art technique for semi-supervised learning on graphs, and attracted much attention over the last five years [2, 36]. Hundreds of GNN models have been proposed and successfully used in various areas, *e.g.,* computer vision [20, 26], natural language processing [1, 17] and data mining [7, 16].

In recent years, there is an emerging trend that equips GNNs with knowledge distillation [11] to accelerate model inference and utilize unlabeled data [34], thereby improving its efficiency or effectiveness. Specifically, in typical knowledge distillation, a lightweight model (*i.e.,* student) learns knowledge by being trained to mimic the soft predictions of a high-capacity model (*i.e.,* teacher). From the perspective of efficiency, knowledge distillation can be utilized to compress a deep GCN [13] model into a shallow one for faster inference [30]. From the perspective of effectiveness, knowledge distillation can extract the knowledge of a GNN model (teacher) and inject it into a non-GNN model (student), in order to use more prior knowledge and unlabeled data for more precise predictions [29].

Besides the choices of teacher and student, the distillation process, which determines how the soft predictions of teacher and student models are matched in the loss function, is also vital to the prediction performance of a distilled student on downstream tasks [11]. For example, a global temperature hyper-parameter [11], which softens the predictions of both teacher and student models, is widely adopted in knowledge distillation to facilitate the knowledge transfer. However, to the best of our knowledge, existing knowledge

distillation methods applied on GNNs all employed predefined distillation processes, *i.e.,* with only hyper-parameters but without any learnable parameters. In other words, the distillation processes are designed heuristically or empirically without any supervision from the performance of distilled students, which isolates distillation from evaluation and thus would lead to suboptimal results.

In this work, we aim to propose a general knowledge distillation framework that can be applied on any pretrained GNN models to further improve their performance. Note that we focus on the distillation process rather than the choice of student models, and thus simply let a student model have the same neural architecture with its teacher as suggested by BAN [6]. To overcome the isolation problem between distillation and evaluation, instead of introducing the global temperature as a hyper-parameter, we innovatively propose to learn node-specific temperatures supervised by the performance of distilled GNN students, as shown in Figure 1. Specifically, we parameterize each node's temperature by a function of its neighborhood's encodings and predictions. Due to the isolation problem in traditional knowledge distillation frameworks [11], the partial derivative of a distilled student's performance with respect to node temperatures does not exist, which makes it non-trivial to learn the parameters in temperature parameterization. Therefore, we design a novel iterative learning process, which alternatively performs preparation, distillation and learning steps, for parameter training. In the preparation step, we will compute the temperature for each node with current parameters, and set up a knowledge distillation loss based on node-specific temperatures; In the distillation step, the parameters of the student model will be updated by the distillation loss; In the learning step, the parameters in temperature modeling will be updated towards higher classification accuracy of distilled GNN students. Moreover, we also introduce a scalable variant of our method by heuristically updating the node-specific temperatures.

We conduct experiments on five public benchmark datasets and apply our framework on five typical GNN models for evaluation. Compared with pretrained teacher model or the student distilled by a global temperature hyper-parameter, experimental results show that on average our distilled GNN student has 3.12% and 2.40% relative improvements in prediction accuracy, respectively. Compared with our full model, the scalable variant enjoys 8 times faster training speed at the cost of 1% prediction accuracy. We also compare our algorithm with state-of-the-art knowledge distillation methods on GNNs, showing consistent improvement for all the five GNN models. Ablation studies and the analysis on learned temperatures further demonstrate the effectiveness of our framework.

Our contributions are summarized as follows:

- To our knowledge, this is the first work to propose a learnable distillation process supervised by the performance of distilled students, for knowledge distillation frameworks applied on GNNs.
- We propose a novel algorithm based on an iterative process of preparation, distillation and learning steps, to train node-specific temperatures for better distillation performance. A scalable variant is also proposed as a trade-off between speed and accuracy.
- Experimental results on five benchmark datasets show that our proposed framework can be successfully applied on five popular GNN models to further improve their prediction accuracy, which demonstrates the generality and effectiveness of our method.

2 RELATED WORK

2.1 Graph Neural Networks

Graph Neural Networks (GNNs) can effectively characterize the structural semantics via the message passing mechanism [8]. In this subsection, we will introduce some typical GNN models, which are also chosen as the teacher/student models in our experiments.

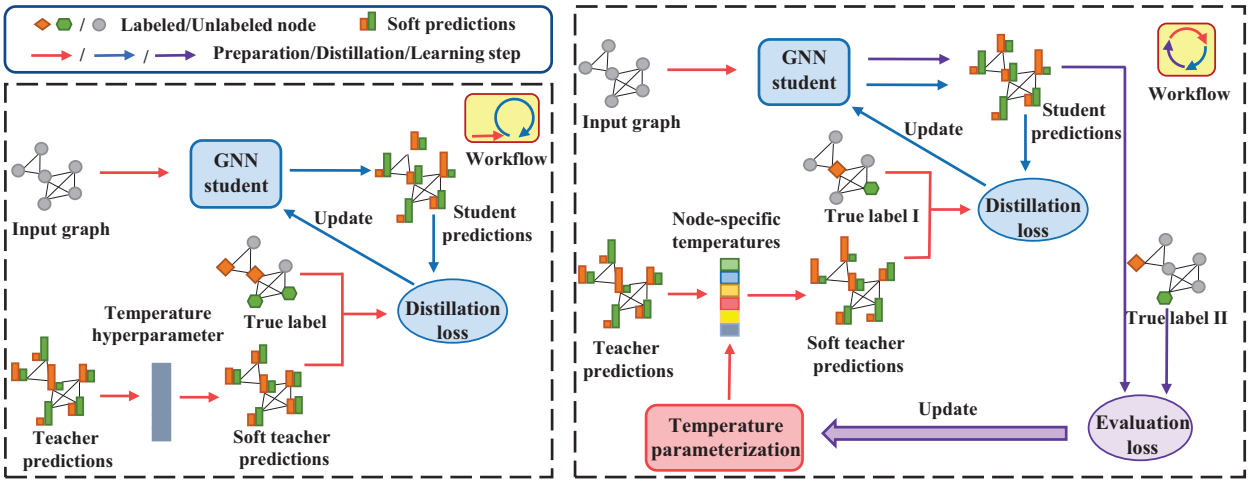
GCN [13] is one of the most famous GNN models, which adapted convolutional neural network to non-Euclidean domain and can be seen as a low-pass filter from the spectral view [15]. GAT [23] introduced attention mechanism to capture the different importance of neighbors. GraphSAGE [9] sampled a fixed number of neighbors to improve efficiency, and designed diverse aggregation methods to improve flexibility. Besides, there are some GNN models that disentangled neighbor aggregation and feature transformation. APPNP [14] performed feature transformation operations before neighbor aggregations, thus increasing the depth of neighbor aggregation while avoiding overfitting. SGC [24] removed the non-linear mapping function between each layer, thus aggregating multi-layer neighbor information before feature transformation, which can improve the computational efficiency.

2.2 Knowledge Distillation

Knowledge distillation [11] was originally proposed for model compression, where a light-weight student model is trained to mimic the soft predictions of a pretrained teacher model. The student can benefit from the knowledge of the teacher, so it is more computationally efficient without sacrificing much prediction performance [18].

Besides the motivation of compressing models, recent studies [6, 27] found that a student can even outperform its teacher if they are parameterized identically. In other words, a model can achieve better prediction performance, if it learns from a pretrained teacher model with the same architecture rather than ground truth labels. For possible explanations of this phenomenon, learning towards soft predictions of a teacher can be interpreted as early stopping [4] or label smoothing [35], and thus has better generalization ability.

Recently, there have been several works applying knowledge distillation to improve the efficiency or effectiveness of GNNs. For efficiency, [30] proposed a local structure preserving module to distill the topological structure from a deep GCN to a shallow one. [28] designed a peer-aware module to help shallow student models to explore the rich structural information during distillation. [12] designed a gradient-based topological semantics alignment loss and a slimmable graph convolution layer to support distillation from diversified teachers. [32] proposed to bring GNNs and multi-layer perceptrons together via knowledge distillation to solve graph dependency issue. For effectiveness, [34] considered both node reliability and edge reliability to make better use of high quality data. [29] designed a non-GNN student model to utilize more structure-based and feature-based prior knowledge. [3] proposed a multi-level self-distillation framework to retain high discrepancy of consecutive layers. [33] designed a reception-aware decoupled GNN model and ensembled several students to construct a powerful teacher. [10] proposed a trainable discriminator that tells student and teacher apart based on adversarial training. [5] employed reinforcement learning to design a new knowledge distillation framework without the need of a well-optimized teacher GNN.



(a) The typical knowledge distillation framework applied on GNNs. (b) Our proposed knowledge distillation framework.

Figure 1: An illustration of (a) the typical framework [11] for distilling GNNs [28–30]; and (b) our proposed distillation framework. Instead of introducing a unified temperature hyper-parameter, we will learn node-specific temperatures supervised by the performance of distilled GNN student based on a novel iterative workflow. Note that the true labels used in the typical framework is divided to two subsets in our framework, and thus we did not employ additional training data.

To the best of our knowledge, existing knowledge distillation methods applied on GNNs all employed predefined distillation processes, which are designed heuristically and controlled by several hyper-parameters. In contrast, we aim to parameterize and learn distillation processes under the supervision of distilled students' performance. In addition, these work all employed a unified temperature hyper-parameter, while we propose to learn node-specific temperatures as trainable parameters. To our knowledge, even in a broader scope of knowledge distillation applications (*e.g.*, computer vision), little work has explored such learnable instance-specific temperatures guided by the performance of distilled students. In this work, we focus on the distillation of GNNs and leave the exploration of combining other neural architectures as future work.

3 METHODOLOGY

In this section, we will first formalize GNNs and the corresponding evaluation task. Then we will introduce the most popular knowledge distillation framework [11] with a temperature hyper-parameter as our basis. Afterward, we will present our algorithm for learning node-specific temperatures towards better performance of distilled GNN students. Finally, we will introduce a scalable variant of our method by heuristically updating node-specific temperatures.

3.1 Preliminaries

3.1.1 ***Semi-supervised Learning on Graphs.*** Node classification, which aims at classifying unlabeled nodes in a graph given labeled ones and the graph structure, is a typical semi-supervised learning task on graphs and widely adopted in the evaluation of many GNN models [25, 31].

Formally, given a connected graph $G = (V, E)$ where V is the vertex set and E is the edge set, node classification is to predict the label of each node v in the unlabeled node set $V_U \subset V$ based on graph structure G, labeled node set $V_L = V \setminus V_U$ and node features $\mathbf{X} \in \mathbb{R}^{|V| \times d}$. Here each row $\mathbf{x}_v \in \mathbb{R}^d$ of matrix $\mathbf{X}$ represents the d-dimensional feature of node v. Let Y be the set of node labels, then the ground truth label of each node v can be denoted as a $|Y|$-dimensional one-hot vector $\mathbf{y}_v$.

3.1.2 ***Graph Neural Networks.*** GNNs can encode each node v into a $|Y|$-dimensional logit vector $\mathbf{f}_v$ by iteratively aggregating the neighborhood information, *i.e.,* the message passing mechanism. In this work, our proposed algorithm is not designed for a specific GNN model and thus can be applied on any GNNs. Therefore, we simply formalize a GNN encoder in a black-box form as

$$\mathbf{f}_{v;\Theta} = \mathrm{GNN}(v|G, \mathbf{X}; \mathbf{\Theta}) \in \mathbb{R}^{|Y|}, \mathbf{p}_{v;\Theta} = \mathrm{softmax}(\mathbf{f}_{v;\Theta}), \quad (1)$$

where $\mathbf{\Theta}$ is the learnable parameters in the GNN and $\mathbf{p}_{v;\Theta}$ is the predicted label distribution normalized by the softmax operator.

Then GNNs will minimize the distance between ground truth label $\mathbf{y}_v$ and predicted label $\mathbf{p}_{v;\Theta}$ for each labeled node $v \in V_L$, and usually employ the cross entropy loss to train the parameters $\mathbf{\Theta}$:

$$\min_{\mathbf{\Theta}} \sum_{v \in V_L} \mathcal{L}_{CE}(\mathbf{y}_v, \mathbf{p}_{v;\Theta}), \quad (2)$$

$$\mathcal{L}_{CE}(\mathbf{y}_v, \mathbf{p}_{v;\Theta}) = -\sum_{i=1}^{|Y|} \mathbf{y}_v[i] \cdot \log \mathbf{p}_{v;\Theta}[i], \quad (3)$$

where $\mathbf{y}_v[i]$ and $\mathbf{p}_{v;\Theta}[i]$ are the i-th entry of vector $\mathbf{y}_v$ and $\mathbf{p}_{v;\Theta}$.

3.2 Knowledge Distillation on GNNs

In this work, we focus on the study of distillation process instead of the choice of student models. Thus we simply let the teacher and student models have the same neural architecture as suggested by BAN [6], and denote them as GNN_T and GNN_S with parameters $\mathbf{\Theta}_T$

and Θ_S, respectively. Given pretrained parameters Θ_T of the teacher model GNN_T (learned by Eq. (2)), we will train the parameters Θ_S of the student model GNN_S by matching the soft predictions between GNN_T and GNN_S. Formally, the knowledge distillation framework aims at optimizing

$$\min_{\Theta_S} \sum_{v \in V} \mathcal{L}_{CE}(\mathbf{p}_{v;\Theta_T}, \mathbf{p}_{v;\Theta_S}) + \lambda \sum_{v \in V_L} \mathcal{L}_{CE}(\mathbf{y}_v, \mathbf{p}_{v;\Theta_S}), \tag{4}$$

where the first term is the cross entropy with the teacher's predictions, the second term is the cross entropy with ground truth labels on V_L, and λ is the balance hyper-parameter.

Note that many knowledge distillation methods since [11] will soften both teacher and student's predictions $\mathbf{p}_{v;\Theta_T}, \mathbf{p}_{v;\Theta_S}$ in the first term of Eq. (4) before distillation, by introducing extra temperature hyper-parameters:

$$\begin{aligned} \mathbf{p}_{v;\Theta_T}(\tau_v^T) &= \text{softmax}(\mathbf{f}_{v;\Theta_T}/\tau_v^T), \\ \mathbf{p}_{v;\Theta_S}(\tau_v^S) &= \text{softmax}(\mathbf{f}_{v;\Theta_S}/\tau_v^S), \end{aligned} \tag{5}$$

where $\tau_v^T, \tau_v^S \in \mathbb{R}_+$ are temperature hyper-parameters. A temperature of 1 corresponds to the original softmax operation. Larger temperatures will produce softer predictions (towards uniform distribution), while smaller temperatures will produce harder predictions (towards one-hot distribution). In the most popular distillation framework [11], all the temperatures are set as the same hyper-parameter τ, *i.e.*, $\tau_v^T = \tau_v^S = \tau$ for every node v. By tuning the global temperature hyper-parameter, the distilled GNN student is evaluated and expected to have better performance than the teacher.

3.3 Learning Node-specific Temperatures

Instead of introducing the global temperature as a hyper-parameter, we innovatively propose to learn node-specific temperatures as trainable parameters for better distillation performance. We will first present how we introduce learnable parameters in temperature parameterization, and then design a novel algorithm for parameter training based on an iterative learning process.

3.3.1 ***Temperature Parameterization.*** Directly assigning each node a free parameter as node-specific temperature would lead to serious overfitting problem. Therefore, we assume that nodes with similar encodings and neighborhood predictions should have similar distillation temperatures. In practice, each node v's temperature can be parameterized by a function with respect to the following features: (1) the student's logit vector $\mathbf{f}_{v;\Theta_S}$ of v, which directly characterizes the current prediction status of the GNN student; (2) the L2-norm of $\mathbf{f}_{v;\Theta_S}$, where a larger norm usually indicates a harder predicted distribution due to the exponential operator in the softmax function; and (3) the prediction entropy of v's neighbors, which describes the label diversity in node v's neighborhood. Intuitively, all the above features will affect the confidence of model predictions and thus should be considered in temperature parameterization.

Formally, we set all the student temperatures τ_v^S to 1 for more calibrated predictions [35], and parameterize the teacher temperature τ_v^T of node v as

$$\tau^T_{v;\Theta_S,\Theta_T,\Omega} = \text{MLP}(\text{Concat}(\mathbf{f}_{v;\Theta_S}, ||\mathbf{f}_{v;\Theta_S}||_2, e_{v;\Theta_T}); \Omega), \tag{6}$$

where $\text{MLP}(\cdot;\Omega)$ denotes a multi-layer perception with parameters Ω, $\text{Concat}()$ is the concatenation operator, $||\cdot||_2$ is the L2-norm, and $e_{v;\Theta_T}$ is defined as the entropy of the average predictions of v's neighbors:

$$e_{v;\Theta_T} = \mathcal{L}_{CE}\Big(\frac{1}{|N_v|}\sum_{u \in N_v} \mathbf{p}_{u;\Theta_T}, \frac{1}{|N_v|}\sum_{u \in N_v} \mathbf{p}_{u;\Theta_T}\Big), \tag{7}$$

where N_v is the set of v's neighbors.

Here $\mathbf{f}_{v;\Theta_S}$ and $||\mathbf{f}_{v;\Theta_S}||_2$ depend on the student parameter Θ_S, while $e_{v;\Theta_T}$ is a node-specific constant since the teacher parameter Θ_T is pretrained and fixed. We use the teacher instead of the student to model the prediction entropy for better numerical stablility. We will investigate the effect of each concatenated component $\mathbf{f}_{v;\Theta_S}, ||\mathbf{f}_{v;\Theta_S}||_2, e_{v;\Theta_T}$, and discuss the learned temperatures in our experiments. In addition, to avoid the gradient explosion or vanishment issue, we also restrict the temperatures within range $[l, r]$ by a function based on sigmoid operation $(r-l)\sigma(\cdot) + l$. Note that alternative formulations of temperature modeling may also exist, but we find that the three terms used in Eq. (6) are sufficient for enhancing the performance and all contribute to the improvement.

3.3.2 ***Iterative Learning Process.*** In order to supervise the training of node-specific temperatures, we partition the labeled node set V_L into two disjoint sets V_{Dis} and V_{Temp}: V_{Dis} is still used in the second term of Eq. (4) for distillation, while V_{Temp} is used for evaluating distilled students and learning node temperatures. Formally, the loss for the distillation part can be written as

$$\begin{aligned} \mathcal{L}_{Dis}(\Theta_S, \Omega) = &\sum_{v \in V} \mathcal{L}_{CE}(\mathbf{p}_{v;\Theta_T}(\tau^T_{v;\Theta_S,\Theta_T,\Omega}), \mathbf{p}_{v;\Theta_S}) \\ &+ \lambda \sum_{v \in V_{Dis}} \mathcal{L}_{CE}(\mathbf{y}_v, \mathbf{p}_{v;\Theta_S}), \end{aligned} \tag{8}$$

and the loss for evaluating distilled students and supervising temperatures is

$$\mathcal{L}_{Temp}(\Theta_S) = \sum_{v \in V_{Temp}} \mathcal{L}_{CE}(\mathbf{y}_v, \mathbf{p}_{v;\Theta_S}). \tag{9}$$

However, due to the isolation between distillation and evaluation, the evaluation loss $\mathcal{L}_{Temp}$ is only related to the parameters Θ_S of student model and the partial derivative $\partial\mathcal{L}_{Temp}/\partial\Omega$ does not exist, which makes it impossible to learn the temperatures via back-propagation.

To address this problem, we propose a novel iterative learning process by alternatively performing the following preparation, distillation and learning steps:

Preparation step: We first calculate the temperature $\tau^T_{v;\Theta_S,\Theta_T,\Omega}$ for each node v according to Eq. (6), and then set up the distillation loss as Eq. (8).

Distillation step: For model distillation, we update the parameters Θ_S through a single step of back-propagation:

$$\Theta'_S := \Theta_S - \alpha \frac{\partial \mathcal{L}_{Dis}(\Theta_S, \Omega)}{\partial \Theta_S}, \tag{10}$$

where α is the learning rate for distillation step.

Learning step: We evaluate $\mathcal{L}_{Temp}$ with the updated parameter Θ'_S, and then perform back-propagation on Ω by the chain rule:

$$\Omega' := \Omega - \beta \frac{\partial \mathcal{L}_{Temp}(\Theta'_S)}{\partial \Theta'_S} \frac{\partial \Theta'_S}{\partial \Omega}, \tag{11}$$

where β is the learning rate for learning step.

Algorithm 1 Learning to Distill GNNs

Input: Graph $G = (V, E)$, node features $\mathbf{X}$, labeled node set V_L, unlabeled node set V_U, teacher GNN model with pretrained parameters Θ_T;
Output: Distilled GNN student with learned parameters Θ_S;

```
Randomly initialize Θ_S and Ω;
Randomly split labeled node set V_L into two disjoint sets V_Dis and V_Temp;
Initialize the parameterized temperatures with Ω by Eq. (6);
while warmup do
    Update Θ_S according to Eq. (8) and (10);
end while
while not converge do
    Compute the parameterized temperatures by Eq. (6);
    Compute the loss of distillation part as Eq. (8);
    Update Θ_S as Eq. (10) and obtain Θ'_S;
    Compute the loss of learning part as Eq. (9) with Θ'_S;
    Update Ω according to Eq. (11) and obtain Ω';
    Overwrite Θ_S and Ω with Θ'_S and Ω';
end while
return distilled student parameter Θ_S.
```

Here we decompose the partial derivative of $\mathcal{L}_{Temp}$ with respect to Ω into the product of $\partial\mathcal{L}_{Temp}(\Theta'_S)/\partial\Theta'_S$ and $\partial\Theta'_S/\partial\Omega$, which can be calculated by the partial derivative of Eq. (9) and (10), respectively. By iteratively executing the preparation, distillation and learning steps, we can train node-specific temperatures parameterized by Ω towards better prediction performance of distilled students.

3.3.3 ***Implementation Details.*** We name our proposed framework as LTD (Learning To Distill). The pseudo code of LTD is shown in Alg. 1. We bisect the labeled node set V_L into V_{Dis} and V_{Temp}, *i.e.,* $|V_{Dis}| = |V_{Temp}|$. Compared with the traditional distillation loss in Eq. (4), V_{Temp} will affect the parameters in the GNN student indirectly and thus alleviate the overfitting issue. We will run 20 epochs of distillation without updating Ω as warmup, and then perform the iterative learning process. The time complexity of each iteration in LTD is linear with respect to the number of nodes and edges. Code and more implementation details are provided in https://github.com/BUPT-GAMMA/LTD.

3.4 Scalable Variant

For faster distillation, we introduce a simplified version of LTD, named LTD$_+$, by heuristically updating node-specific temperatures.

3.4.1 ***Heuristic Temperature Update.*** In order to calculate an appropriate temperature for each node, we need to use the features considered in the temperature parameterization of LTD. Formally, the temperature τ_v^T of node v is calculated as:

$$\tau^T_{v;\Theta_S,\Theta_T,\Omega} = \mu \cdot \frac{1}{\mathrm{Max}(\mathbf{f}_{v;\Theta_S})} + \nu \cdot \frac{1}{||\mathbf{f}_{v;\Theta_S}||_2} + \gamma \cdot e_{v;\Theta_T}, \qquad (12)$$

where $\mu, \nu, \gamma > 0$ are balance hyper-parameters, and $\mathrm{Max}(\cdot)$ returns the maximum value of a vector. The formation is inspired by the learned temperatures of LTD, where nodes with smaller $\mathrm{Max}(\mathbf{f}_{v;\Theta_S})$, $||\mathbf{f}_{v;\Theta_S}||_2$ and larger $e_{v;\Theta_T}$ will have higher temperatures.

Table 1: Dataset statistics.

Dataset	Citeseer	Cora	Pubmed	A-Computers	A-Photo
# Nodes	2,110	2,485	19,717	13,381	7,487
# Edges	3,668	5,069	44,324	245,778	119,043
# Features	3,703	1,433	500	767	745
# Classes	6	7	3	10	8

3.4.2 ***Connections between LTD$_+$ and LTD.*** The full model LTD has a more elaborately designed learning process of node temperatures, and can empirically help student models achieve better performance. Inspired by the learned temperatures of LTD, we propose LTD$_+$ as a simple yet efficient variant, which enjoys a faster model training speed at the cost of prediction accuracy. As a trade-off between speed and accuracy, LTD$_+$ is particularly suitable to large scale graphs or deep GNN models.

4 EXPERIMENTS

In this section, we conduct experiments on five benchmark datasets to answer the following research questions (RQs):

- RQ1: Can the GNN students distilled by our LTD outperform those distilled by other knowledge distillation frameworks? How about the efficiency of LTD and LTD$_+$ compared with other distillation frameworks?
- RQ2: How does our LTD perform under different settings (*i.e.,* ablation studies, distinct combinations of GNN teacher/student)?
- RQ3: What patterns can we observe from learned parameters of LTD (*i.e.,* node-specific temperatures)?

4.1 Experimental Setup

4.1.1 ***Datasets.*** In our experiments, we employ five benchmark datasets widely used in previous works [19, 29]. The statistical information of the five datasets is shown in Table 1.

4.1.2 ***Teacher/Student Models.*** We use GCN, GAT, GraphSAGE, APPNP, SGC as teacher/student models and set them as [29].

4.1.3 ***Experimental Settings.*** We conduct experiments on the most popular task for evaluating GNNs, *i.e.,* semi-supervised node classification. For each dataset, we use 40 nodes per class as the training data, 10 nodes per class as the validation data, and all the rest nodes for testing. For each combination of GNN model and dataset, we will pretrain a GNN model as the teacher and fix its parameters. After the distillation, we will evaluate the distilled students learned by different frameworks. For evaluation metric, we will report the classification accuracy as previous work [13, 23].

4.2 Analysis of Main Results (RQ1)

4.2.1 ***Comparison with the Traditional Distillation Framework.*** First of all, to validate our motivation of learning node-specific distillation temperatures, we will test the following frameworks based on different GNN teacher/student models:

- FT (Fixed Temperature): All nodes use the same temperature as a hyper-parameter, which is adopted in most knowledge distillation frameworks.
- LTD$_{\text{w/o LS}}$: The proposed LTD method without learning steps, *i.e.,* the parameter Ω is never updated after initialization.

Table 2: Classification accuracies with GNN models as GAT and GraphSAGE.

Dataset	GNN	Framework variants				+Impv.	GNN	Framework variants				+Impv.
	GAT	FT	$\text{LTD}_{\text{w/o LS}}$	LTD	LTD_{+}		SAGE	FT	$\text{LTD}_{\text{w/o LS}}$	LTD	LTD_{+}	
Citeseer	0.7525	0.7564	0.7044	**0.7735**	0.7630	2.26%	0.7276	0.7409	0.7613	**0.7746**	0.7613	1.75%
Cora	0.8520	0.8534	0.8356	**0.8656**	0.8614	1.43%	0.8426	0.8501	0.8482	**0.8703**	0.8576	2.38%
Pubmed	0.7944	0.8048	0.8144	**0.8274**	0.8098	1.60%	0.8189	0.8271	0.7362	**0.8401**	0.8347	1.57%
A-Computers	0.8091	0.8079	0.7823	**0.8304**	0.8234	2.63%	0.7829	0.7999	0.7934	**0.8144**	0.8019	1.81%
A-Photo	0.9094	0.9194	0.9145	**0.9316**	0.9248	1.33%	0.9146	0.9194	0.9059	**0.9306**	0.9232	1.22%

Table 3: Classification accuracies with GNN models as APPNP and SGC.

Dataset	GNN	Framework variants				+Impv.	GNN	Framework variants				+Impv.
	APPNP	FT	$\text{LTD}_{\text{w/o LS}}$	LTD	LTD_{+}		SGC	FT	$\text{LTD}_{\text{w/o LS}}$	LTD	LTD_{+}	
Citeseer	0.7530	0.7508	0.7641	**0.7851**	0.7691	2.75%	0.7238	0.742	0.7536	**0.7873**	0.7602	4.47%
Cora	0.8581	0.8585	0.8333	**0.8693**	0.8660	1.26%	0.8454	0.8562	0.8534	**0.8660**	0.8632	1.14%
Pubmed	0.8301	0.8313	0.8258	**0.8436**	0.8391	1.48%	0.8205	0.7944	0.7939	**0.8405**	0.8249	2.44%
A-Computers	0.8095	0.8141	0.8061	**0.8363**	0.8250	2.73%	0.8047	0.8326	0.7821	**0.8528**	0.8330	2.43%
A-Photo	0.9225	0.9304	0.9261	**0.9337**	0.9316	0.35%	0.9118	0.9165	0.9084	**0.9297**	0.9254	1.44%

Table 4: Classification accuracies with GNN model as GCN.

Dataset	GNN	Framework variants				+Impv.
	GCN	FT	$\text{LTD}_{\text{w/o LS}}$	LTD	LTD_{+}	
Citeseer	0.7359	0.7547	0.7586	**0.7851**	0.7613	3.49%
Cora	0.8534	0.8600	0.8614	**0.8721**	0.8651	1.24%
Pubmed	0.7989	0.8029	0.7897	**0.8191**	0.8118	2.02%
A-Computers	0.8594	0.8468	0.8443	**0.8645**	0.8600	0.59%
A-Photo	0.9223	0.9231	0.9032	**0.9324**	0.9275	1.01%

• LTD: The proposed method.

• LTD_{+}: The proposed scalable variant of LTD.

For the framework variant FT, we conduct a careful grid search of global temperature τ from $\{0.001, 0.01, 0.1, 1, 4, 8, 12, 16, 20, 24\}$ and balance hyper-parameter λ from $\{0.1, 1, 50, 100, 200\}$, and employ Adam optimizer with learning rate 0.01 for updating parameters. For a fair comparison, we ensure that the number of hyper-parameter trials in FT is more than that of our LTD.

We present the results on five benchmark datasets with five GNN models in Table 2, 3 and 4. We **bold** the best results among the teacher model and the three distilled students learned by different framework variants. The relative improvements over the better model between teacher and FT are also reported. From the experimental results, we have the following observations:

(1) The GNN students distilled by our proposed LTD framework can achieve consistent improvements over all five GNN models on the five datasets. Compared with the pretrained teacher model and the student distilled by a fixed temperature hyper-parameter, on average our distilled GNN student has 3.12% and 2.40% relative improvements in prediction accuracy, respectively. This observation verifies our motivation of learning node-specific temperatures and demonstrates the effectiveness of our method.

(2) The scalable variant LTD_{+} consistently outperforms FT and has a relative improvement of 1.11%. Also, the full model LTD has a 1.2% relative advantage over LTD_{+}. Therefore, both temperature parameterization and adaptive learning are effective for prediction performance.

(3) The performance of the framework variant $\text{LTD}_{\text{w/o LS}}$ is very unstable due to the removal of learning steps. Compared with $\text{LTD}_{\text{w/o LS}}$, our method has 3.92% relative improvements on average, which demonstrates the necessity of temperature learning. Therefore, our iterative learning process can guide the distillation process towards better prediction performance of distilled students.

(4) Though we conduct a careful grid search for FT to select the best temperature hyper-parameter, FT still performs worse than the pretrained teacher in many cases, *e.g.,* APPNP on Citeseer and SGC on Pubmed. Hence learning node-specific temperatures is a more reasonable choice for distilling GNNs than employing a fixed global temperature.

(5) The performance of $\text{LTD}_{\text{w/o LS}}$ can outperform the teacher GNN model in some cases, *e.g.,* GAT and GraphSAGE on Citeseer dataset. A possible reason is that our constructed node feature provides informative prior knowledge for parameterizing node-specific temperatures.

4.2.2 ***Comparison with State-of-the-art Distillation Frameworks.*** To validate our motivation that modeling the distillation process is important in distilling GNNs, we also compare our LTD with two state-of-the-art distillation frameworks for GNNs:

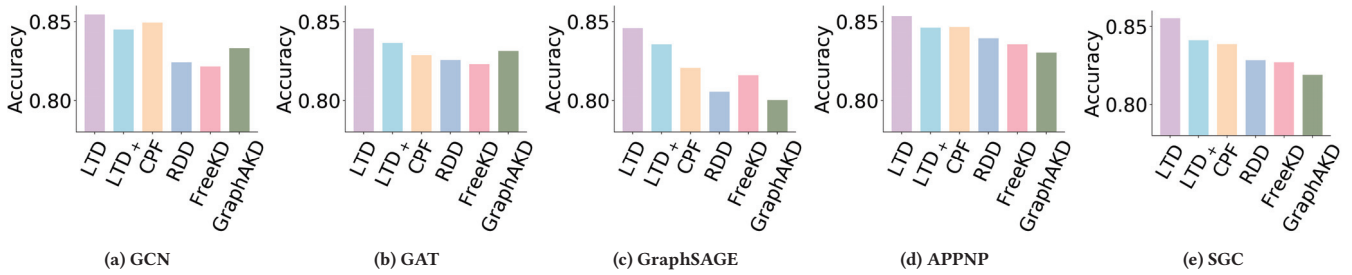
• CPF [29]: Given a pretrained GNN teacher, CPF designs a student model based on label propagation and feature transformation to take advantage of prior knowledge.

• RDD [34]: Given a pretrained GNN teacher, RDD considers the node/edge reliability and proposes to correct the teacher outputs.

• GraphAKD [10]: With the help of adversarial training, GraphAKD designs a learnable distance function to measure the distribution discrepancy of teacher and student.

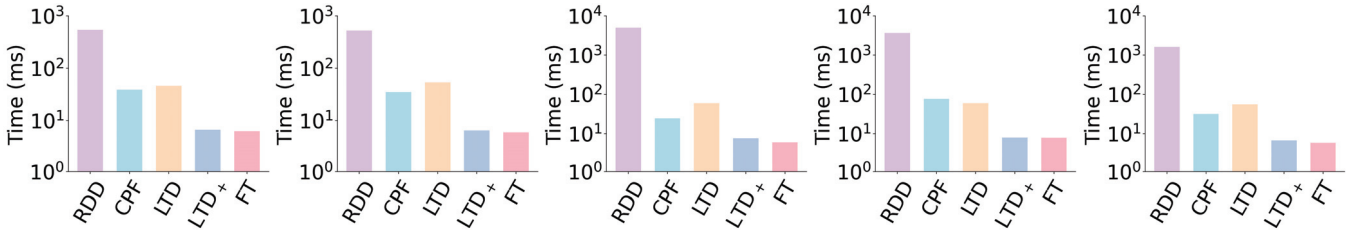
• FreeKD [5]: Based on reinforcement learning, FreeKD enhances the performance of GNNs without requiring an optimal teacher.

Figure 2 shows the average accuracies of the three frameworks. We can see that LTD consistently performs the best for all five GNNs. The relative improvement against the best performed baseline CPF is 2.02%. These methods all employed predefined distillation processes, while LTD can learn a distillation objective towards better



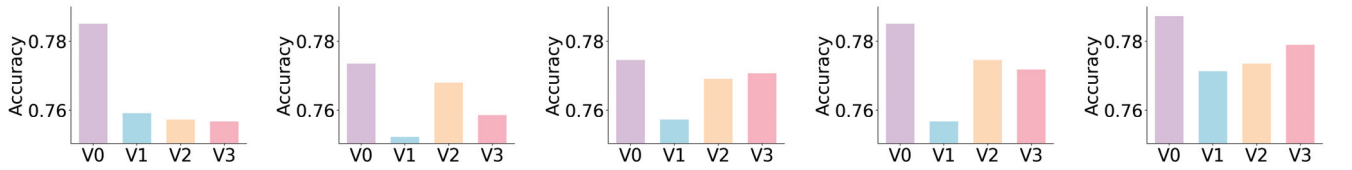
(a) GCN (b) GAT (c) GraphSAGE (d) APPNP (e) SGC

Figure 2: Average classification accuracies of different distillation frameworks on five GNN models. The accuracies are averaged over five datasets.



(a) Cora (b) Citeseer (c) Pubmed (d) A-Computers (e) A-Photo

Figure 3: Running time (in log scale) of different distillation frameworks with GCN teacher/student on five datasets.



(a) GCN (b) GAT (c) GraphSAGE (d) APPNP (e) SGC

Figure 4: Classification accuracies of different ablated models on Citeseer dataset. V0: our proposed method; V1: logit vector $\mathbf{f}_v$ is removed; V2: L2-norm of $\mathbf{f}_v$ is removed; V3: the neighborhood entropy e_v is removed.

Figure 5: The accuracy gains of LTD when the GNN teacher and student have distinct architectures.

generalization ability. Meanwhile, LTD$_+$ is also very competitive, which has a 0.74% relative improvement over CPF. This experiment shows the superiority of our method in distilling GNNs.

4.2.3 ***Analysis of Scalability.*** In order to verify the efficiency of our model, we compare the running time of a single epoch for different distillation frameworks in Figure 3. We can see that LTD$_+$ and FT are the fastest, LTD and CPF are comparable, and RDD is the slowest. It is worth noting that our scalable variant LTD$_+$ runs 321.4/7.8/5.9 times faster than RDD/LTD/CPF. To conclude, LTD offers the best prediction performance, while LTD$_+$ is the cost effective choice for large graphs.

4.3 Analysis of Extensive Studies (RQ2)

We conduct additional experiments under different settings to further demonstrate our effectiveness. As we only present the results on Citeseer dataset for convenience.

4.3.1 ***Ablation Studies.*** We conduct ablation studies to investigate the effect of each concatenated component in temperature parameterization. As shown in Figure 4, we compare our full model (V0) with three ablated models (V1-V3) where logit vector $\mathbf{f}_{v;\Theta_S}$, L2-norm $||\mathbf{f}_{v;\Theta_S}||_2$ or neighborhood entropy $e_{v;\Theta_T}$ is respectively removed. Figure 4 shows that all the three components contribute to the overall performance, which demonstrates the necessity of each component for temperature modeling.

4.3.2 ***Performance with Distinct GNN Teacher/Student.*** To further prove the generality of our LTD, we conduct additional experiments when the GNN teacher and student have distinct architectures. Figure 5 shows the results of $5 \times 5 = 25$ teacher-student combinations, and we have the following observations: (1) Our proposed LTD can effectively improve the performance of pretrained GNN teacher in all 25 combinations. (2) The simplest GNN, *i.e.,* SGC, is the best student that works well with all five GNN teachers. This observation also aligns with the assumption in [29] that simpler student models can utilize more prior knowledge. (3) The accuracies of most distilled students fall within a narrow range of $(0.775, 0.788)$, which shows that our LTD is not fastidious about the implementations of GNN teacher/student.

4.4 Learned Temperature Analysis (RQ3)

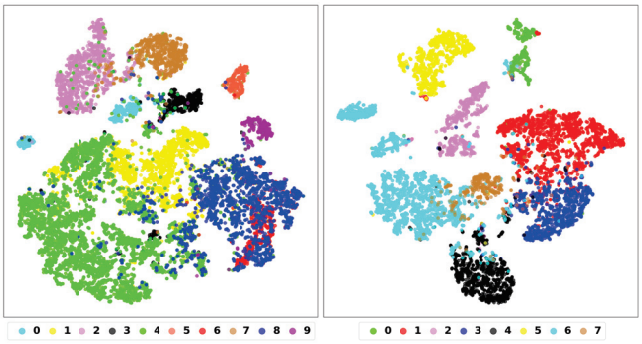
We analyze the learned node-specific temperatures in $5 \times 5 = 25$ GNN-dataset combinations, and present the following case studies based on GAT to illustrate how LTD helps learn a better distillation.

(1) First of all, to prove that the node temperatures are significantly changed and truly node-specific after the training process, we compute the Pearson correlation coefficient between randomly initialized temperatures and learned ones. Taking Cora dataset as an example, the correlation coefficient is only 0.02. The mean and standard deviation of node temperatures after training are 0.28 ± 0.26. Note that a change of 0.1 in temperature could cause a hundredfold increase in the exponential operator of softmax function. Therefore, LTD can learn diverse node-specific temperatures for distilling.

(2) We observe that nodes in a "confusing" class (*i.e.,* mixed with other classes) tend to have higher temperatures. For example, we visualize the node embeddings learned by GAT teacher via t-SNE [21] in Figure 6, and use node colors to indicate their labels. As shown Figure 6(a), the blue nodes are mixed with the red ones, and we find that 88%/83% of the red/blue nodes have higher temperatures than the average temperature of all nodes. Similarly, in Figure 6(b), 75%/76% of the red/blue nodes are higher than average. This observation indicates that LTD is not confident with the predictions of such nodes and thus will assign higher temperatures to soften their label distributions towards uniform ones.

(3) We observe that nodes with a small L2-norm $||\mathbf{f}_v||_2$ tend to have higher temperatures. Taking Cora dataset as an example, we select Top-50 nodes with the highest/lowest learned temperatures, and find that the Pearson correlation coefficient between temperature and L2-norm $||\mathbf{f}_v||_2$ in the selected 100 nodes is -0.77, which indicates a strong negative correlation. We also notice that nodes with a smaller L2-norm usually have fewer non-zero features, which could be insufficient to support confident enough predictions.

(4) Note that we allow negative node temperatures which will completely overwhelm the predictions in the pretrained teacher. We observe that nodes with negative learned temperatures are likely to be wrongly predicted by the GNN teacher. Taking A-Computers dataset as an example, its accuracy is 0.81 for all nodes and only 0.50 for the nodes with negative learned temperatures. Hence, the node-specific temperatures adopted in LTD are very flexible, and can help correct the predictions of GNN teachers.



(a) A-Computers (b) A-Photo

Figure 6: Visualization of node embeddings learned by GAT teacher on A-Computers and A-Photo. The blue and red classes overlap with each other, and their nodes have higher learned temperatures.

5 CONCLUSION

In this paper, we propose a novel knowledge distillation framework LTD that can be applied on any pretrained GNN models to further improve their prediction performance. Instead of introducing a global temperature hyper-parameter as most previous work did, we innovatively propose to learn node-specific distillation temperatures supervised by the performance of distilled students. Specifically, we parameterize each node's temperature by a function of its neighborhood's encodings and predictions, and design a novel iterative learning process for model distilling and parameter learning. As a cost effective choice, the scalable variant LTD$_+$ is proposed by heuristically updating node-specific temperatures. We conduct experiments on five benchmark datasets and show that our proposed framework can be successfully applied on five popular GNN models. Extensive studies further demonstrate the effectiveness and efficiency of our method.

For future work, we will investigate the feasibility of combining our LTD with other knowledge distillation frameworks applied on GNNs, since they all used a fixed temperature hyper-parameter. Another direction is to explore other unsupervised evaluation tasks, *e.g.,* link prediction and clustering. Also, the proposed framework could also be generalized to the distillation of other neural networks (*e.g.,* Transformer [22]) as well.

ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China (No. U20B2045, 62192784, 62172052, 62002029, 62006129).

REFERENCES

[1] Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. *arXiv preprint arXiv:1704.04675* (2017).

[2] Hongxu Chen, Hongzhi Yin, Tong Chen, Quoc Viet Hung Nguyen, Wen-Chih Peng, and Xue Li. 2019. Exploiting centrality information with graph convolutions for network representation learning. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 590–601.

[3] Yuzhao Chen, Yatao Bian, Xi Xiao, Yu Rong, Tingyang Xu, and Junzhou Huang. 2021. On Self-Distilling Graph Neural Network. In *Proceedings of IJCAI*.

[4] Bin Dong, Jikai Hou, Yiping Lu, and Zhihua Zhang. 2019. Distillation ≈ Early Stopping? Harvesting Dark Knowledge Utilizing Anisotropic Information Retrieval For Overparameterized Neural Network. *arxiv* (2019).

[5] Kaituo Feng, Changsheng Li, Ye Yuan, and Guoren Wang. 2022. FreeKD: Free-direction Knowledge Distillation for Graph Neural Networks. *arXiv preprint arXiv:2206.06561* (2022).

[6] Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born Again Neural Networks. In *Proceedings of ICML*.

[7] Jun Gao, Jiazun Chen, Zhao Li, and Ji Zhang. 2021. ICS-GNN: lightweight interactive community search via graph neural network. *Proceedings of the VLDB Endowment* 14, 6 (2021), 1006–1018.

[8] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for Quantum chemistry. In *Proceedings of ICML*. 1263–1272.

[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of NeurIPS*. 1024–1034.

[10] Huarui He, Jie Wang, Zhanqiu Zhang, and Feng Wu. 2022. Compressing Deep Graph Neural Networks via Adversarial Knowledge Distillation. *arXiv preprint arXiv:2205.11678* (2022).

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2014. Distilling the knowledge in a neural network. *Proceedings of NeurIPS*.

[12] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. 2021. Amalgamating Knowledge From Heterogeneous Graph Neural Networks. In *Proceedings of CVPR*. 15709–15718.

[13] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR*.

[14] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *Proceedings of ICLR*.

[15] Weixuan Liang, Sihang Zhou, Jian Xiong, Xinwang Liu, Siwei Wang, En Zhu, Zhiping Cai, and Xin Xu. 2020. Multi-view spectral clustering with high-order optimal neighborhood laplacian matrix. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[16] Hao Liu, Jindong Han, Yanjie Fu, Jingbo Zhou, Xinjiang Lu, and Hui Xiong. 2020. Multi-modal transportation recommendation with unified route representation learning. *Proceedings of the VLDB Endowment* 14, 3 (2020), 342–350.

[17] Diego Marcheggiani and Ivan Titov. 2017. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In *Proceedings of EMNLP*.

[18] Franco Maria Nardini, Cosimo Rulli, Salvatore Trani, and Rossano Venturini. 2022. Distilled Neural Networks for Efficient Learning to Rank. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[19] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Pitfalls of graph neural network evaluation. *arxiv* (2018).

[20] Damien Teney, Lingqiao Liu, and Anton van Den Hengel. 2017. Graph-structured representations for visual question answering. In *Proceedings of CVPR*. 1–9.

[21] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 11 (2008).

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS*. 5998–6008.

[23] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *Proceedings of ICLR*.

[24] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of ICML*. 6861–6871.

[25] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *TNNLS* (2020).

[26] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5410–5419.

[27] Kunran Xu, Lai Rui, Yishi Li, and Lin Gu. 2020. Feature normalized knowledge distillation for image classification. In *Proceedings of ECCV*. Springer, 664–680.

[28] Bencheng Yan, Chaokun Wang, Gaoyang Guo, and Yunkai Lou. 2020. TinyGNN: Learning Efficient Graph Neural Networks. In *Proceedings of SIGKDD*. 1848–1856.

[29] Cheng Yang, Jiawei Liu, and Chuan Shi. 2021. Extract the Knowledge of Graph Neural Networks and Go Beyond it: An Effective Knowledge Distillation Framework. In *Proceedings of WWW*. 1227–1237.

[30] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. 2020. Distilling Knowledge From Graph Convolutional Networks. In *Proceedings of CVPR*. 7074–7083.

[31] Lingling Zhang, Shaowei Wang, Jun Liu, Qika Lin, Xiaojun Chang, Yaqiang Wu, and Qinghua Zheng. 2022. MuL-GRN: Multi-Level Graph Relation Network for Few-Shot Node Classification. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[32] Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. 2021. Graph-less neural networks: Teaching old mlps new tricks via distillation. In *Proceedings of ICLR*.

[33] Wentao Zhang, Yuezihan Jiang, Yang Li, Zeang Sheng, Yu Shen, Xupeng Miao, Liang Wang, Zhi Yang, and Bin Cui. 2021. ROD: Reception-aware Online Distillation for Sparse Graphs. In *Proceedings of SIGKDD*. 2232–2242.

[34] Wentao Zhang, Xupeng Miao, Yingxia Shao, Jiawei Jiang, Lei Chen, Olivier Ruas, and Bin Cui. 2020. Reliable Data Distillation on Graph Convolutional Network. In *Proceedings of SIGMOD*. 1399–1414.

[35] Zhilu Zhang and Mert Sabuncu. 2020. Self-Distillation as Instance-Specific Label Smoothing. *Proceedings of NeurIPS*.

[36] Huan Zhao, Quanming Yao, and Weiwei Tu. 2021. Search to aggregate neighborhood for graph neural network. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. 552–563.