Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Online adversarial knowledge distillation for graph neural networks

Can Wang, Zhe Wang, Defang Chen, Sheng Zhou *, Yan Feng, Chun Chen

College of Computer Science, Zhejiang University, ZJU-Bangsun Joint Research Center, Shanghai Institute for Advanced Study of Zhejiang University, Hangzhou, 310013, Zhejiang, China

ARTICLE INFO

Keywords:
Knowledge distillation
Graph neural networks
Dynamic graph
Online distillation

ABSTRACT

Knowledge distillation, a technique recently gaining popularity for enhancing model generalization in Convolutional Neural Networks (CNNs), operates under the assumption that both teacher and student models are trained on identical data distributions. However, its effect on Graph Neural Networks (GNNs) is less than satisfactory since the graph topology and node attributes are prone to evolve, thereby leading to the issue of distribution shift. In this paper, we tackle this challenge by simultaneously training a group of graph neural networks in an online distillation fashion, where the group knowledge plays a role as a dynamic virtual teacher and the structure changes in graph neural networks are effectively captured. To improve the distillation performance, two types of knowledge are transferred among the students to enhance each other: *local knowledge* reflecting information in the graph topology and node attributes, and *global knowledge* reflecting the prediction over classes. We transfer the global knowledge with KL-divergence as the vanilla knowledge distillation does, while exploiting the complicated structure of the local knowledge with an efficient adversarial cyclic learning framework. Extensive experiments verified the effectiveness of our proposed online adversarial distillation approach. The code is published at https://github.com/wangz3066/OnlineDistillGCN.

1. Introduction

Recent advances in computing devices such as graphical processing units are rendering possible to train Deep Neural Networks (DNNs) with millions of parameters. In particular, Deep Convolutional Neural Networks (DCNNs) have made dramatic breakthroughs in a variety of computer vision applications including image classification (Simonyan & Zisserman, 2015) and object detection (He et al., 2017), etc, due to its extraordinary ability on feature extraction and expression. However, these DCNNs often contain tens or even hundreds of layers with millions of trainable parameters. For example, Inception V3 network (Szegedy et al., 2015) requires about 24M trainable parameters while ResNet152 (He et al., 2016) contains about 60M trainable parameters. Training such cumbersome models is highly time-consuming and space-demanding, thus their deployment on resource-limited devices such as mobile phones or tablets is restricted. For years, many approaches have been proposed to transform the heavy neural network into a lightweight one, such as network pruning (Wu et al., 2016) and low-rank factorization (Denton et al., 2014). Knowledge Distillation (KD) is also proposed to compress and accelerate those cumbersome *teacher* models by training a lightweight *student* model to align with the teacher' predictions (Hinton et al., 2015). The predictions from teacher are used as a positive regularization to improve the generalization ability of the student network. Besides prediction alignment, knowledge distillation could also be achieved by aligning intermediate representations (Chen, Mei et al., 2021; Romero et al., 2015) and sample relations (Park et al., 2019; Passalis & Tefas, 2018) between the teacher and student models. Due to its simple implementation and effectiveness, knowledge distillation has gained much popularity.

Recently, knowledge distillation has attracted the attention of the Graph Neural Networks (GNNs) community, where a tiny student GNN model is trained by distilling knowledge from a large pre-trained teacher GNN model (Yang et al., 2021, 2020). A successful knowledge distillation requires that the data distribution of both the teacher and student models are similar. Nevertheless, directly employing the vanilla teacher–student paradigm will be plagued by the volatility of the graph data. That is, the graph topology and node attributes are likely to change over time. For example, in social networks, new nodes or new edges may be added to the graph. This indicates the distributions of training sets for the teacher and student models might vary. Consequently, the student model trained with outdated knowledge from the static teacher will probably result in sub-optimal solutions. A static teacher model therefore is not optimal for transferring knowledge

* Corresponding author.
E-mail addresses: wcan@zju.edu.cn (C. Wang), zhewangcs@zju.edu.cn (Z. Wang), defchern@zju.edu.cn (D. Chen), zhousheng_zju@zju.edu.cn (S. Zhou), fengyan@zju.edu.cn (Y. Feng), chenc@zju.edu.cn (C. Chen).

https://doi.org/10.1016/j.eswa.2023.121671
Received 20 October 2022; Received in revised form 25 August 2023; Accepted 14 September 2023

in a volatile environment. Although this distribution shift problem has been studied for Convolutional Neural Networks (CNNs) (Nguyen et al., 2021), the problem in the graph domain has not been well addressed. The challenge of this problem lies in the constant evolution of graph data. Traditional distribution alignment techniques, such as transfer learning, would be ineffective because they assume that the data distributions from the source and target domains are fixed. Besides, pretraining a large teacher model requires significantly more processing time and computation resources than training a student model, resulting in many economic and environmental problems. Considering the expensive cost and the frequency of the change in graph-structured data training, retraining the heavy-weight teacher GNN when the graph changes will be impractical. To alleviate this problem, instead of training a single heavy-weight static teacher GNN, we simultaneously train a group of student GNNs in an online fashion to effectively capture dynamic changes and leverage the group knowledge as an effective proxy for the pre-trained teacher model. In this way, each student GNN will distill the knowledge from other peers to enhance its performance. When retraining is required in case of graph updates, much less cost will be incurred by the light-weight student models than the heavy-weight teacher models.

More specifically, each student GNN is trained with the conventional cross-entropy loss from the ground truth labels as well as two types of knowledge from other peers, which corresponds to two characteristics of graph data: *Local Knowledge* and *Global Knowledge*: (1) *Local Knowledge* reflects information contained in the graph topology and node attributes, which is instantiated as the learned node embedding with aggregated attributes from neighboring nodes. To better capture the complicated structure of the local knowledge, we employ the adversarial cyclic learning to effectively achieve the embedding alignment among students. (2) *Global Knowledge* reflects the prediction over classes by the semi-supervised classifiers. We transfer this type of knowledge with KL divergence as the vanilla knowledge distillation does.

We conduct extensive experiments on benchmark datasets with different GNN architectures to verify the generalization of our proposed **O**nline **A**dversarial knowledge **D**istillation (OAD) framework. Ablation studies are also designed to demonstrate the necessity of each proposed component.

In summary, the contributions of this paper are concluded as follows:

- A novel Online Adversarial Distillation (OAD) framework is proposed to simultaneously train a group of student GNN models in an online fashion in which they can learn from peers and effectively capture structure updates in graph neural networks.
- Student GNNs are trained with both global and local knowledge in the GNNs group for better distillation performance. To learn the complicated structure of the local knowledge, adversarial cyclic learning is employed to achieve more accurate embedding alignment among students.
- Extensive experiments including transductive/inductive node classification and object recognition on benchmark datasets as well as dynamic graphs demonstrate the effectiveness of our framework.

The structure of this paper is organized as follows. In Section 2, we review the literatures that closely related to the topic of this paper. In Section 3, we introduce the details of the proposed Online Adversarial knowledge Distillation (OAD) for GNNs. In Section 4, we compare the performance of proposed OAD model with other baseline methods to demonstrate its effectiveness. The conclusions and future works of this paper is presented in Section 5.

2. Related works

2.1. Knowledge distillation

Knowledge distillation is a model compression technique which aims to boost the performance of a lightweight student model leveraging the 'dark knowledge' of a over-parameterized teacher model. The first work of knowledge distillation is introduced by Hinton et al. (2015), which regards the prediction of teacher model as knowledge and transfers it to student model using Kullback–Leibler (KL) divergence. To further enhance the performance, subsequent works attempted to align feature maps from intermediate layers (Chen, Mei et al., 2021; Romero et al., 2015; Sepahvand et al., 2022; Zagoruyko & Komodakis, 2017) or align sample relations based on their representations from the penultimate layer (Chen et al., 2022; Park et al., 2019; Passalis & Tefas, 2018). To circumvent the pretraining step for obtaining a large teacher model, online knowledge distillation provides a more economic solution by simultaneously training a group of student models and encouraging each student to distill from other peers (Chen et al., 2020; Lan et al., 2018; Zhang et al., 2018, 2022). The reason behind these works is that we can dynamically construct one or several "virtual" teacher(s) with higher accuracy to guide the student during training. The most straightforward way to obtain the "virtual" teacher(s) is to simply average the predictions of other peers (Zhang et al., 2018), which is later improved by adaptively learning the weights through gate or self-attention mechanism (Chen et al., 2020; Lan et al., 2018). Although online knowledge distillation has been extensively studied on CNNs, to the best of our knowledge, our work is the first attempt on GNNs.

2.2. Graph neural networks

Graph learning has many useful real-world applications such as hyperspectral image classification (Hong et al., 2020), remote sensing (Hong, Yokoya, Chanussot et al., 2019; Hong, Yokoya, Ge et al., 2019). Recently, deep learning based model such as Graph Neural Networks have dominated many graph learning tasks. Early works of GNN are mostly spectral based, which firstly transforms graph signal into spectral domain via graph Fourier Transformation and conducts convolution on the spectral domain (Bruna et al., 2014; Henaff et al., 2015). However, the computation complexity of spectral GNN is $O(N^3)$ due to eigen decomposition of Laplacian matrix, which is time-consuming. ChebyNet (Defferrard et al., 2016) simplifies spectral GNN via K-order polynomial approximation of the graph Laplacian matrix. Graph Convolutional Network (GCN) (Kipf & Welling, 2017) further introduces the first-order approximation of ChebyNet, which can also be viewed as the simplest spatial-based GNN. Spatial-based GNN uses a message passing process by aggregating the messages of neighbors recursively, thus the information from higher-order neighbors will be passed to lower-order neighbors. Graph Attention Network (GAT) introduced self-attention mechanism to allow more flexible importance assignment to neighbors (Veličković et al., 2018). GraphSAGE generated embeddings by sampling and aggregating neighbor features, enabling inductive learning on large-scale graphs (Hamilton et al., 2017). Graph Isomorphism Network (GIN) (Xu et al., 2019) proved that the message-passing GNN is at most as powerful as 1-Weisfeiler Lehman Test, based on which they design the most powerful GNN that can distinguish any non-isomorphic neighborhoods. Some other GNNs are designed for 3D point cloud recognition tasks where the graphs are generated with k nearest neighbors (Landrieu & Simonovsky, 2018; Wang et al., 2019).

Although the GNN is effective in handling graph data, it still suffers from the problem caused by redundant network parameters. This problem has been addressed by some network compression techniques, such as pruning (Chen, Sui et al., 2021). Some works also explored the potential of knowledge distillation to improve the performance of

GNNs (Yang et al., 2021, 2020). The student models of Yang et al. (2021) are parameterized label propagation module instead of GNNs. LSP (Yang et al., 2020) transferred local structure knowledge of a teacher GNN to a student GNN with less parameters. The main differences between the proposed OAD and existing knowledge distillation with GNN methods are two-fold: Firstly, existing knowledge distillation with GNN methods are mostly offline, where a teacher model is pretrained and a student model is trained to distill the knowledge of teacher. Since the graph-structured data changes frequently, the data distributions of teacher and student may vary. Therefore, the knowledge from the pretrained teacher GNN may be ineffective for the student GNN. Our method uses an online knowledge distillation framework for GNNs, where multiple student models are trained simultaneously, which ensures that the data distribution of students and "virtual teacher" is always aligned. Secondly, existing knowledge distillation with GNN methods such as LSP (Yang et al., 2020) mostly align the intermediate feature maps via hand-crafted distance measure such as L2 distance or KL divergence. However, many hand-crafted distance measures are not defined in highly correlated intermediate features (Tian et al., 2020), thus they may be ineffective in transferring the knowledge of intermediate layers. Instead, in our proposed OAD framework, we employ a cyclic GAN framework for local knowledge distillation, where some discriminators are employed to implicitly supervise the distribution alignment of teacher and student model. Therefore, the intermediate feature of students and virtual teachers are automatically aligned under the supervision of discriminators.

2.3. Generative adversarial networks

Generative Adversarial Network (GAN) is a generative model initially proposed by Goodfellow et al. (2014). It is consisted by a generator and a discriminator, where the generator attempts to synthesis new data samples similar to original dataset, and the discriminator is adopted to evaluate the authenticity of data instances. Due to its strong ability to generate high-quality samples, GANs have received significant attention. GAN has been improved from different aspects. To better control the mode of generated data, some works proposed to condition GAN on the latent variables (Chen et al., 2016; Mirza & Osindero, 2014). Some works adopt more elaborate distance metrics to increase the training stability (Gulrajani et al., 2017; Mao et al., 2017). Recently, in the KD field, the discriminator is included to enforce the student network to mimic the teacher network's pattern (Liu et al., 2018; Wang et al., 2018). Some existing works have employed GAN to generate new graph with node features and adjacency matrices (Bojchevski et al., 2018; De Cao & Kipf, 2018) similar to original graphs. Our works are significantly different from aforementioned works. Specifically, our OAD leverages GAN to enable local knowledge transfer among different student GNNs while these works use GAN to generate new data samples.

3. Method

In this section, we will elaborate on our proposed **O**nline **A**dversarial knowledge **D**istillation (OAD) framework for graph neural networks. Generally, we will utilize the online knowledge distillation paradigm to simultaneously train a group of student models and encourage each student to learn from other peers mutually.

More specifically, we start with a group of untrained student models with randomly initialized parameters. Intuitively, these students are expected to capture different characteristics of graph data, which can be mainly grouped into two types: (1) Local knowledge combining the local neighborhood structure and the node features, which is naturally represented by the learned node embeddings. (2) Global knowledge referring to the label distribution predicted by the GNN models, deciding the corresponding categories for each node.

Both of these two knowledge types are transferred among students while they differ in the way of distillation. As the global knowledge is represented by the distribution over the labels, we follow the vanilla knowledge distillation and use the Kullback–Leibler divergence (KL) to measure the distance. In contrast, the intermediate layers of GNN models are formulated by aggregating the messages from K-hop neighbors, which encodes rich local subgraph information and is more complicated than the label distribution. How to ditill the knowledge of intermediate layers can be referred as a "local knowledge distillation" process. More importantly, each student model is expected to capture different characteristics of the graph. Strictly aligning the embedding with L_2 or KL measurement will impair the diversity and degenerates the knowledge transferred among student models. Inspired by the success of adversarial training in learning complicated distribution, we train several discriminators to distinguish the difference between the learned embedding distribution, which yields more accurate results. Additionally, we utilize the cyclic training strategy to improve the training efficiency. Fig. 1 illustrates the proposed OAD framework, which contains three major components: local knowledge distillation, global knowledge distillation, and adversarial cyclic training. In the following contents, we will first recap some fundamental concepts about knowledge distillation and graph neural networks. Then, we introduce the details about the proposed model.

3.1. Supervised learning

In a supervised C-class classification problem, we are given a dataset with m labeled samples $\mathcal{D} = \{\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}\}_{i=1}^{m}$, where $\boldsymbol{x}^{(i)} \in \mathcal{R}^d$ is the input feature sampled from an unknown data distribution $\mathcal{X}$ and $\boldsymbol{y}^{(i)} \in \mathcal{R}^C$ is the one-hot ground truth label. Our goal is to learn a classifier parameterized by Θ: $f(\cdot;\Theta) : \mathcal{X} \rightarrow [0,1]^C$. The output of mapping function f is known as "logits" $\boldsymbol{z}$, which is subsequently transferred into probability distribution $\boldsymbol{p} = \sigma(\boldsymbol{z}) \in \mathcal{R}^C$ with a softmax function $\sigma(\cdot)$:

$$p_i = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)} \tag{1}$$

The classifier is typically trained to minimize the cross entropy (CE) between probability distribution $\boldsymbol{p}$ and ground truth labels $\boldsymbol{y}$:

$$\mathcal{L}_{CE}(\boldsymbol{p}, \boldsymbol{y}) = \sum_{i=1}^{C} y_i \log p_i \tag{2}$$

3.2. Knowledge distillation

Knowledge distillation is generally composed by two neural networks with distinct architectures: A teacher network with larger parameters and a student network with fewer parameters. Besides standard cross entropy loss between student predictions $\boldsymbol{p}_s$ and ground truth label $\boldsymbol{y}$, student network is also trained to minimize the Kullback–Leibler (KL) divergence between softened student probability $\sigma(\boldsymbol{z}_s/T)$ and softened teacher probability $\sigma(\boldsymbol{z}_t/T)$:

$$\mathcal{L}_{KD} = \mathcal{L}_{CE}(\boldsymbol{p}_s, \boldsymbol{y}) + \alpha T^2 KL(\sigma(\boldsymbol{z}_s/T), \sigma(\boldsymbol{z}_t/T)) \tag{3}$$

where T is a hyper-parameter known as temperature. The higher T leads to more significant softened effect.

Besides output logits, the teacher networks' intermediate layers also encoded highly representative knowledge to the given dataset, which may be potential to boost the students performance further. Many works have been proposed to match the raw or transformed hidden layers. For example, Fitnet (Romero et al., 2015) randomly selects a hidden layer from both teacher network and student network and minimize their mean square error (MSE) loss:

$$\mathcal{L}_{Fitnet} = \frac{1}{2} \| r(\boldsymbol{h}_s) - \boldsymbol{h}_t \|^2 \tag{4}$$

where $r(\cdot)$ is a dimension transformation operator such as Multi-Layer Perception.

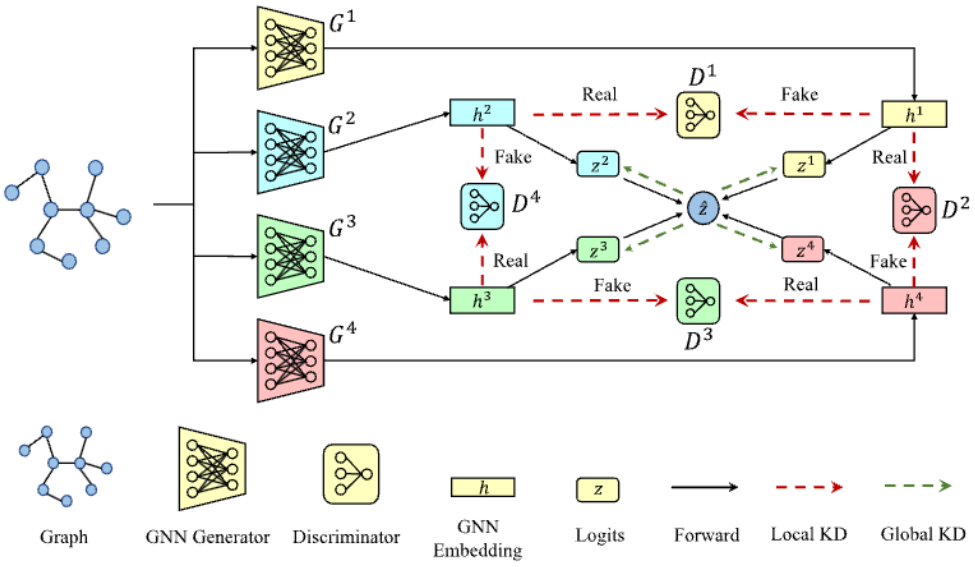

Fig. 1. Overview of the proposed online adversarial knowledge distillation framework for GNNs. We show the case with four student GNN models. The student network is a GNN generator. Each student model is assigned with a discriminator. The local knowledge is distilled by a cyclic generator-discriminator framework, denoted as red dot lines. The global knowledge is distilled by the KL divergence between student logits and "virtual" teacher's logits, denoted as green dot lines.

3.3. Graph neural networks

In this subsection, we briefly recall the definition of GNNs. The input of a GNN is a graph denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is the node set and E is the edge set. The node features are given by a matrix $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\} \in \mathcal{R}^{N\times D}$, where $\boldsymbol{x}_i \in \mathcal{R}^D$ is the node feature of v_i. The adjacency matrix of G is denoted by $A^{N\times N}$ where $A_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$ and $A_{ij} = 0$ if $(v_i, v_j) \notin \mathcal{E}$. The connected neighborhood set of node v_i is denoted as $\mathcal{N}_i$.

Graph neural networks aim at learning a mapping function $f : \mathcal{R}^{N\times N} \times \mathcal{R}^{N\times D} \to \mathcal{R}^{N\times D'}$. The core operation of graph neural networks is aggregating features from the neighborhood nodes. A typical GNN with L layers is defined as:

$$G(\boldsymbol{x}_i) = \boldsymbol{h}_i^{(L)}, \quad \boldsymbol{h}_i^{(l+1)} = \psi(\boldsymbol{h}_i^{(l)}, \mathcal{AGG}_{j\in\mathcal{N}_i}\phi(\boldsymbol{h}_i^{(l)}, \boldsymbol{h}_j^{(l)})) \tag{5}$$

where $l = 1, 2, \ldots, L-1$, $\boldsymbol{h}_i^{(l)}$ is the embedding of node v_i in the lth hidden layer of the GNN encoder G, $\phi(\cdot,\cdot)$ is a pairwise message passing function, $\mathcal{AGG}$ function is used to aggregate messages from all neighbors of v_i, and ψ function updates the state of v_i based on its current state as well as the messages propagated by local neighbors. Existing GNN methods have mainly varies on the message passing functions $\phi(\cdot,\cdot)$ as well as the aggregation function $\mathcal{AGG}$. To verify the generalization of our proposed framework for the graph neural networks, we test three widely adopted graph neural networks named GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018) and GraphSAGE (Hamilton et al., 2017).

3.4. Local knowledge distillation

The aforementioned definition of GNNs demonstrates that the learned embedding $\boldsymbol{h}_i$ can effectively represent the local information around node v_i. Inspired by the idea of intermediate-layer association in knowledge distillation, a straightforward strategy to enhance representation ability of $\boldsymbol{h}_i$ is to distill the intermediate layer or its transformation from a better generalized teacher GNN. However, we propose that this method suffers from following two drawbacks: (1) It requires a teacher GNN usually with much more parameters being pretrained, which needs expensive computations. (2) More importantly, the graph-structured data can be more frequently changed than static data such as images. For example, in a social network, there are millions of new following relationships generated everyday. Thus, the dataset distributions of teacher and student network may vary given the drastic changes of graph, and the knowledge learned by teacher network may not be helpful for the student network. The experiments on dynamic graph prove this point, see Section 4.4.

Instead, we propose Online Adversarial Distillation (OAD) for GNN to address above two drawbacks. In OAD framework, a group of lightweight student GNNs are trained simultaneously and each student is expected to distill the knowledge from a "virtual" teacher formulated by the remaining group members, thus a pretrained teacher network is not needed. Since the total number of parameters in OAD framework is less than pretrained teacher network, it is more spatially efficient than vanilla knowledge distillation framework. Besides, as all the networks are trained on the same graph, the problem of domain shift caused by the dynamicity of the graph is solved.

Next, we introduce the local knowledge distillation module of OAD. The mth student GNN is expected to distill the knowledge from the intermediate layers of the remaining $M-1$ GNNs. We adopt a GAN framework to facility the distillation procedure. For the mth student network, its graph convolution layers G^m are utilized as a generator to generate embedding $\boldsymbol{h}^m$. Then, to distill the knowledge of embedding $\boldsymbol{h}_q$ generated by another GNN members G^q, a discriminator $D^{mq} : \mathcal{R}^{N\times D'} \to \mathcal{R}^N$ is included. The discriminator inputs h^m or h^q and outputs a scalar between 0 (fake) and 1 (real). When the input is $\boldsymbol{h}^m$, D^{mq} is trained to output 0 (fake). When the input is $\boldsymbol{h}^q$, D^{mq} is trained to output 1 (real). The GNN generator G^m is trained to confuse the discriminator and obtain high score (close to real) after discrimination.

However, discriminating all pairs of M student models takes $M * (M-1)/2$ times, which is unfriendly to efficient training, especially when the number of student model grows. To improve the training efficiency while maintaining the performance, we follow Chung et al. (2020) and utilize the cyclic training strategy to reduce the number of discrimination into M. For D^m, it takes G^m and G^{m+1} as input and learns to discriminate G^m as fake and G^{m+1} as real. In this way, each student GNN generator is trained to confuse the D^m and generate the embedding that follows the distribution of other student models. For the entire students group, the knowledge of node embeddings are transferred in a cycle: $M \to M-1$, $M-1 \to M-2$, $\ldots$, $1 \to M$. We use the original GAN framework where both discriminators and generators

are trained to minimize the following loss:

$$\mathcal{L}_D = \sum_{m=1}^{M-1} [\log D^m(G^m(X)) - \log D^m(G^{m+1}(X))] + \log D^M(G^M(X)) - \log D^M(G^1(X)) \quad (6)$$
$$\mathcal{L}_G = \sum_{m=1}^{M} -\log D^m(G^m(X))$$

In the training phase, sequentially updating discriminator D^m and GNN encoder G^m will destroy the computation graph since G^m will also join the loss calculation of D^{m+1}. Instead, we fix generators and update all the discriminators firstly, then we fix discriminators and update all the generators. This group optimization process will not increase the training burden.

3.5. Global knowledge distillation

Given the embedding $\boldsymbol{h}_i^m$ of node v_i learned by the mth GNN generator, we use the graph convolutional layer Θ and softmax function to transform the representation into the logits $\boldsymbol{z}_i^m$, which is computed as:

$$z_i^{mk} = \frac{exp(\Theta(h_i^{mk})/\mathcal{T})}{\sum_{j=1}^{K} exp(\Theta(h_i^{mj})/\mathcal{T})} \quad (7)$$

where z_i^{mk} is the probability of node v_i in class k predicted by mth student model, $\mathcal{T}$ is the temperature. Note that $\mathcal{T}$ is greater than 1 and we set $\mathcal{T} = 3.0$ for all the experiments in this paper.

Note that network parameters of M students are initialized differently, thus their logits are different during the training process. For the mth student GNN, the remaining M-1 different students can act as an "ensemble" teacher, which is generally more effective than single model. We verify this point in Section 4.5. Therefore, In the global knowledge distillation, for the mth student GNN model, we formulate the global knowledge of this "ensemble" teacher by averaging their logits $\boldsymbol{z}_i^j$ as:

$$\widehat{\boldsymbol{z}_i^m} = \frac{1}{M-1} \sum_{j,j\neq m}^{M} \boldsymbol{z}_i^j \quad (8)$$

We use the KL divergence between these two softened class distribution as the global knowledge distillation loss:

$$\mathcal{L}_B = \mathcal{T}^2 \sum_{i=1}^{N} \sum_{m=1}^{M} KL(\boldsymbol{z}_i^m, \widehat{\boldsymbol{z}_i^m}) \quad (9)$$

where $\mathcal{T}^2$ is multiplied to keep the gradient this loss term roughly unchanged when $\mathcal{T}$ changes (Hinton et al., 2015).

3.6. Overall training process

The overall training loss for student GNN models is formulated as:

$$\mathcal{L}_{total} = \mathcal{L}_{CE} + \alpha\mathcal{L}_G + \beta\mathcal{L}_B \quad (10)$$

where $\mathcal{L}_{CE}$ is the supervised cross entropy loss, α and β are the coefficients for the corresponding distillation losses. In our experiments, each student do not distill the knowledge from other peers at the initial epochs of training, i.e., they only learn from the ground-truth labels. After warmed-up phase, each student has its own perception to the mapping structure, then the meaningful knowledge can be distilled to others.

3.7. Complexity analysis

In this subsection, we analyze the computation complexity of the proposed OAD framework. Let T_G^i and T_Θ^i be the computing time of one forward propagation on graph encoder and output layers of the ith student, respectively. Let T_D^i be the computing time of one forward propagation on the ith discriminator. The overall computing time of one forward propagation of proposed model is:

$$T_{OAD} = \sum_{i=1}^{M} T_G^i + \sum_{i=1}^{M} T_\Theta^i + \sum_{i=1}^{M} 2T_D^i = O(\sum_{i=1}^{M} T_G^i) + O(\sum_{i=1}^{M} T_\Theta^i) + O(\sum_{i=1}^{M} T_D^i) \quad (11)$$

Thus, the computing time of the proposed OAD model increases linearly w.r.t. the number of group members. Meanwhile, let N_G and N_Θ be the parameter number of graph encoder and output layers for a single student GNN, respectively. Let N_D be the parameter number of a single discriminator. Thus the total parameter number of OAD model is:

$$S_{OAD} = M(N_G + N_\Theta + N_D) \quad (12)$$

when M is limited, the total number of parameters is much less than a cumbersome GNN.

Algorithm 1 Online adversarial distillation for Graph Neural Network

Input: Graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. Node attribution matrix $X = \{x_1, x_2, ..., x_N\} \in \mathcal{R}^{N\times D}$. Adjacency matrix $A \in \mathcal{R}^{N\times N}$. M student graph neural networks $\{f_m\}_{m=1}^M$. M discriminators $\{D_m\}_{m=1}^M$. Epochs of independent learning $epoch_t$, epochs of online knowledge distillation $epoch_o$, weights of loss α, β.

Output: Well-trained graph neural network ensemble.

```
for t = 1 to epoch_t do
    Forward M student GNN models to generate predictions {f_m(X)}_{m=1}^M.
    Compute cross entropy loss L_CE for each GNN.
    Update the parameters of M students by backward propagation of L_CE.
end for
for t = 1 to epoch_o do
    Forward M student GNN models to generate predictions {f_m(X)}_{m=1}^M and hidden embedding {G_m(X)}_{m=1}^M.
    Forward M discriminators {D_m}_{m=1}^M.
    Compute L_D.
    Update the parameters of M discriminators by back propagating L_D.
    Compute cross entropy loss L_CE for each student GNN.
    Compute global knowledge distillation loss L_B.
    Compute local knowledge distillation loss L_G.
    Update the parameters of M student GNNs by back propagation of L_total.
end for
```

4. Experiment

To evaluate the performance of proposed OAD framework, we conduct node classification tasks on citation datasets, Protein–Protein Interaction (PPI) dataset, and object recognition task on ModelNet40 dataset. A variety of classic GNN architectures are tested to evaluate the generalization of the OAD framework, including GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018) and GraphSAGE (Hamilton et al., 2017). Besides, we compare the performance of OAD and vanilla KD method on dynamic graphs. The result indicates that proposed OAD method is more robust to the drastic changes of graphs than vanilla KD method. We also conduct an ablation study to evaluate the effectiveness of each module and the group mutual learning strategy in the OAD framework. Furthermore, we analyze the impact of group size used in online knowledge distillation to the performance of model. Finally, we visualize the feature space generated by different models, which

demonstrates that proposed OAD framework can generate much more effective representations. The proposed OAD model is implemented using Pytorch-1.9.1 Library. All the experiments are conducted on a Linux Ubuntu 18.04 Server with one GeForce RTX 2080 Ti GPU of 11.0 GB memory.

4.1. Transductive node classification on citation datasets

We firstly evaluate our model by transductive node classification task on three citation datasets: Cora, Pubmed and Citeceer. In these datasets, nodes represent the publications and edges represent citations between them. The Cora dataset contains 2708 nodes, 5429 edges, 1433 features and seven possible classes. The Pubmed dataset has 19 717 nodes, 44 338 edges, 500 features and three possible classes. The Citeceer dataset contains 3327 nodes, 4732 edges, 3703 edges and six possible classes. Following the same experimental settings of existing methods (Yang et al., 2016), 20 nodes per class are selected for training, 500 nodes are for validation and 1000 nodes are for testing. The nodes in all of these datasets have only one label, so we use Top-1 Accuracy as the evaluation metric. In the transductive training setting, the model can access all nodes on the whole graph.

We compare the performance of vanilla GNNs and the corresponding OAD variants, denoted as "a specific model + OAD". Each student GNN with the OAD framework has the same network architecture as its corresponding vanilla GNN. The results of OAD are the average performance of the student models in the group. The results of "A specific model + KD" are obtained as follows: we firstly pre-train a strong enough teacher model as save its parameters. Then, we use this teacher model the supervise the training of a small-size student model, along with the supervision of ground truth label.

We also include the performance of a GNN ensemble with the same group size and network architecture as OAD framework. All the hyperparameters settings of "Ensemble" are same as "single GNN" except different group size. Note that at inference phase, *an ensemble model averages all the students' predictions while the output of proposed OAD method is the prediction of single student GNN.*

Table 1 summarizes the network architectures and their parameter numbers in citation experiment. We set the group size in this experiment as 4. As can be seen from Table 1, *the total parameters of student networks and extern discriminators are less than one teacher network*. We adopt two-layer GNNs in all the experiments. For GCN and GraphSAGE, the dimension of hidden layer is 16. For GAT, following the settings in original paper (Veličković et al., 2018), the hidden dimension is 8 and the attention heads are set as [8,1] for Cora and Citeceer, and [8,8] for Pubmed, respectively. The aggregation type of GraphSAGE is "mean". For all the experiments, we fix learning rate, weight decay and optimizer as 0.005, 0.0005 and Adam, respectively. The OAD model is warmed up for 100 epoch, and further mutual learned for 100 epoch. For other baselines, the total traning epoch is 200. For OAD, we adopt GCNs as skeleton of discriminators. The input of discriminator is the last hidden layers (the layer before last convolution layer) of GNNs. The discriminators used in GCN experiments are a single-layer GCN, while the discriminators used in GAT and GraphSAGE experiments are two-layer GCNs with hidden dimensions of 16. We set the loss coefficient $\alpha = 1$ and $\beta = 1$ for all the network-dataset combinations. Note that we evaluate the performance of all the student models and report the best case for our model.

We compare the classification accuracy of different methods. The accuracy is defined as the proportion of number of correctly predicted samples to the number of all samples. Table 2 illustrates the experimental results of the transductive node classification task on citation datasets. The values of the best performance are marked in bold. The baseline methods include the single GNN and vanilla KD method. We observe that all the GNNs with the proposed OAD framework gain improvement over vanilla KD method and single GNNs. Specifically, on the Citeceer dataset, our model can bring 0.74% and 2.05% performance improvement for GCN and GraphSAGE, respectively. This demonstrates the effectiveness and the generalization of the proposed OAD framework.

Table 1
Network architecture used in citation networks.

Dataset	Network	Layers	Features	Attention heads	Parameters
Cora	GCN-Teacher	2	128,7	/	0.18M
	GCN-Student	2	16,7	/	0.02M
	GCN-Discriminator	1	1	/	0.017K
	GAT-Teacher	2	128,7	8,1	1.47M
	GAT-Student	2	8,7	8,1	0.09M
	GAT-Discriminator	2	16,1	/	1.06K
	GraphSAGE-Teacher	2	128,7	/	0.37M
	GraphSAGE-Student	2	16,1	/	0.05M
	GraphSAGE-Discriminator	2	16,1	/	0.29K
Citeceer	GCN-Teacher	2	128,6	/	0.47M
	GCN-Student	2	16,6	/	0.06M
	GCN-Discriminator	1	1	/	0.017K
	GAT-Teacher	2	128,6	8,1	3.80M
	GAT-Student	2	8,6	8,1	0.24M
	GAT-Discriminator	2	16,1	/	1.06K
	GraphSAGE-Teacher	2	128,6	/	0.95M
	GraphSAGE-Student	2	16,6	/	0.12M
	GraphSAGE-Discriminator	2	16,1	/	0.29K
Pubmed	GCN-Teacher	2	128,3	/	0.06M
	GCN-Student	2	16,3	/	0.008M
	GCN-Discriminator	1	1	/	0.017K
	GAT-Teacher	2	128,3	8,8	0.54M
	GAT-Student	2	8,3	8,8	0.03M
	GAT-Discriminator	2	16,1	/	1.06K
	GraphSAGE-Teacher	2	128,3	/	0.13M
	GraphSAGE-Student	2	16,3	/	0.02M
	GraphSAGE-Discriminator	2	16,1	/	0.29K

Table 2
Top-1 accuracy (%) of different methods on three citation datasets. The results are obtained under the statistical significance level $\alpha < 0.05$. The results of teacher and student ensemble are also included. The best results are in bold.

Model	Cora	Pubmed	Citeceer
GCN	80.0	78.9	65.7
GCN+KD	80.4	79.0	66,5
GCN+OAD	**80.9**	**79.3**	**67.5**
Improve %	0.62%	0.38%	1.50%
GCN Ensemble	80.3	79.0	66.0
GCN Teacher	81.5	79.2	68.2
GAT	81.9	77.0	68.9
GAT+OAD	**82.2**	**78.5**	**69.6**
GAT+KD	81.2	78.0	68.6
Improve %	1.23%	0.77%	1.01%
GAT Ensemble	82.1	77.5	69.9
GAT Teacher	82.2	77.9	69.9
GraphSAGE	80.3	76.5	68.0
GraphSAGE+OAD	**81.6**	**77.4**	**69.7**
GraphSAGE+KD	81.0	76.9	68.3
Improve %	0.74%	0.65%	2.05%
GraphSAGE Ensemble	81.2	77.1	69.4
GraphSAGE Teacher	81.7	78.3	69.9

4.2. Inductive node classification on PPI dataset

In this section, we evaluate the proposed OAD framework by inductive node classification task on Protein-to-Protein Interaction (PPI) dataset (Zitnik & Leskovec, 2017). PPI dataset contains 24 graphs corresponding to different human issues. The average number of nodes per graph is 2372. Each node has 50 features that encode information about the positional gene set and immunological characters. The total number of edges is 818 716, constructed by prepossessing data provided by Hamilton et al. (2017). The dimension of the label for each node is 121. We follow the same data splitting protocol as Veličković et al. (2018), in which the number of graphs for training, validation and testing are 20, 2 and 2, respectively. Note that the testing graphs are completely undetectable during training. For PPI dataset, each node

Table 3
Network architectures used on PPI dataset.

Network	Layers	Features	Attention heads	Parameters
GAT-Teacher	3	256,256,121	4,4,6	3.64M
GAT-Student	5	64,64,64,64,121	2,2,2,2,2	0.17M
GAT-Discriminator	2	64,1	/	0.008M
GraphSAGE-Teacher	3	256,256,121	/	0.22M
GraphSAGE-Student	5	64,64,64,64,121	/	0.047M
GraphSAGE-Discriminator	2	64,1	/	0.004M

may belong to multiple categories and we evaluate the overall F1 score of the entire graph.

Table 3 summarizes the students' and teacher's network structures used in the experiment. The group size is set as 4. As can be seen from Table 3, *the total parameters of student networks and extern discriminators are less than one teacher network*. For our model, students in the group warmup with cross entropy loss for 50 epochs, then they learn mutually for 50 epochs. For the single GNN and LSP (Yang et al., 2020), the total number of training epoch is 100. For all the comparisons, the learning rate, weight decay and optimizer are set as 0.005, 0 and Adam, respectively. The aggregation type of GraphSAGE is "mean". For our model, we adopt two-layer GCNs as discriminators in feature level distillation with 64 hidden neurons. The input of discriminator is the last hidden layers (the layer before last convolution layer) of GNNs. The loss coefficients is set as $\alpha = 1, \beta = 0.1$ for both GAT and GraphSAGE. All the results are generated by our implementations except LSP (Yang et al., 2020) which we run the codes released by authors. We also include the performance of the teacher model and an ensemble model. The group size of the ensemble model is same as OAD. Other hyperparameters settings of ensemble are the same as single GNN except group size. Note that we evaluate the performance of all the student models and report the best case for our model. We repeat every experiment 4 times with different random seeds to initialize and report the average and variance.

Our baselines include some offline knowledge distillation methods: KD (Hinton et al., 2015), FitNet (Romero et al., 2015), AT (Zagoruyko & Komodakis, 2017) and LSP (Yang et al., 2020). For these methods, we firstly train a common large enough teacher model as described in Table 3. Then, these KD methods will distill the knowledge of this teacher. The baselines also include an classic online knowledge distillation method - DML (Zhang et al., 2018). The evaluation of DML is same as OAD by averaging the performance of all group members. We adopt GAT (Veličković et al., 2018) and GraphSAGE (Hamilton et al., 2017) as the skeleton models. The student architecture of all the baseline methods are same.

We compare the F1 Score of different methods. The F1 score is defined as:

$$F1\_Score = \frac{2 * P * R}{P + R} \tag{13}$$

where P is precision and R is the recall. Table 4 summarizes the results on the PPI dataset. The values of the best performance are marked in bold. We observe that training with a pretrained teacher model will negatively influence the performance of GNN, while our proposed OAD can boost its performance. Specifically, compared to single GNN model, proposed OAD method has 0.85% and 2.27% improvement for GAT and GraphSAGE, respectively. Moreover, OAD exceeds other vanilla knowledge distillation variants by more than 2% for both GAT and GraphSAGE, indicating "virtual" teacher paradigm is more effective for GNN. Compared to DML, proposed OAD method capture both local and global knowledge of GNN and improves by 0.35% and 1.27% for GAT and GraphSAGE, respectively.

Table 4
F1 scores of different methods on PPI dataset. The results are obtained under the statistical significance level $\alpha < 0.05$. The results of teacher and student ensemble are also included. The best results are in bold.

Method	GAT	GraphSAGE
Student	90.13 ± 0.30	76.03 ± 0.61
Fitnet	87.63 ± 0.37	67.75 ± 0.96
AT	85.60 ± 1.20	74.45 ± 1.18
LSP	89.60 ± 0.62	74.83 ± 0.66
KD	90.35 ± 0.33	76.28 ± 0.66
DML	90.63 ± 0.50	77.03 ± 0.33
OAD	**90.98 ± 0.46**	**78.30 ± 0.36**
Teacher	97.0	86.3
Ensemble	94.8	80.1

4.3. Object recognition on ModelNet40 dataset

ModelNet40 dataset is adopted for the object recognition task, consisting of 12 311 meshed CAD models from 40 categories (9843 for training and 2468 for testing). Each CAD model belongs to one category, thus we use Top-1 and balanced accuracy as the evaluation metric. Each model is constructed by a series of points with (x,y,z) coordinates. Following the experimental settings of Qi et al. (2017), we uniformly sample 1024 points for each model and then discard the original one. We use DGCNN as the skeleton model and construct the graph in the feature space by KNN algorithm as the previous work (Wang et al., 2019). It is worth noting that the graph varies in different layers and different training stages. For a fair comparison, we use the same student architecture as that of LSP (Yang et al., 2020).

We adopt the same teacher and student architectures as Yang et al. (2020), which are summarized in Table 5. We set the group size of our OAD model as 4. As can be seen from Table 5, *the total parameters of student networks and extern discriminators are less than one teacher network*. For online knowledge distillation, the epoch of warmup learning and mutual learning are set as 100 and 150, respectively. For fair comparison, the single DGCNN is trained for 250 epochs. For all settings, we use SGD with momentum rate of 0.90 as optimizer. The learning rate, weight decay, dropout rate are 0.1 and 0.0001 and 0.5, respectively. The discriminators for proposed model are two-layer MLPs with batch normalization, which take the last hidden layer (the layer before output layer) of DGCNN as input. We set $\alpha = 1, \beta = 0.05$ as loss weights. We cite the results of Fitnet, AT and LSP in the original paper (Yang et al., 2020). We also include the performance of the teacher model and an ensemble model. The group size of the ensemble model is same as OAD. Other hyperparameters settings of ensemble are the same as single GNN except group size. Note that we evaluate the performance of all the student models and report the best case for our model. We repeat every experiment 4 times with different random seeds to initialize and report the average.

We compare the recognition accuracy of different methods. Table 6 summarizes the results of object recognition on the ModelNet40 dataset. The values of the best performance are marked in bold. We can observe that the performance of our proposed model exceeds all the comparison methods. Besides, OAD can obtain a similar Top-1 accuracy score and a higher balanced accuracy score than the teacher model. This indicates that stacking several lightweight models with a significantly less number of parameters can perform almost as well as the teacher model. Another interesting observation is that most baseline methods perform worse than the student model, which is explainable as the graphs varies during training. This further indicates the advantage of the OAD framework in adapting the changes of the graph.

4.4. Dynamic graph

In this section, we compare the performance of a single GNN, conventional knowledge distillation and our proposed OAD model on

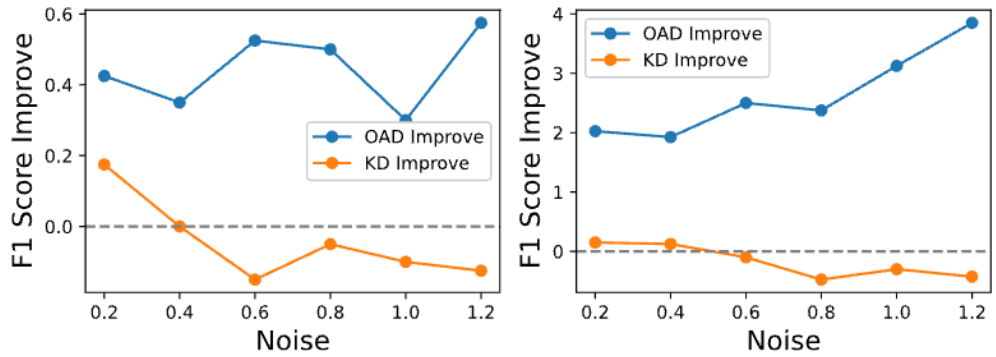

Fig. 2. F1 score improvement w.r.t. a single GNN after adding random noise on the node attributions of PPI dataset. The blue lines denote "OAD_Improve" and the orange lines denote "KD_Improve". The left subplot shows the experiment on GAT. The right subplot shows the experiment on GraphSAGE.

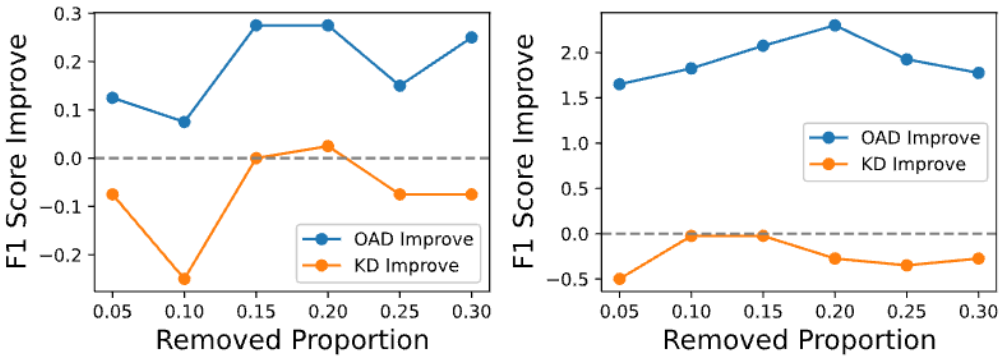

Fig. 3. F1 score improvement w.r.t. a single GNN after removing certain proportions of edges of the graphs of PPI dataset. The blue lines denote "OAD_Improve" and the orange lines denote "KD_Improve". The left subplot shows the experiment on GAT. The right subplot shows the experiment on GraphSAGE.

Table 5
Network architectures used on ModelNet40 dataset.

Network	Layers	Feature maps	MLPs	K	Parameters
Teacher	5	64,64,128,256,1024	512,256	20	1.81M
Student	4	32,32,64,128	256	10	0.1M
Discriminator	2	32,1	/	/	4.25K

Table 6
Testing accuracy of different methods on ModelNet40 dataset. The results are obtained under the statistical significance level $\alpha < 0.05$. The results of teacher and student ensemble are also included. The best results are in bold.

Method	Top-1 accuracy	Balanced accuracy
Student	92.3	88.8
Fitnet	91.1	87.9
AT	91.6	87.9
LSP	91.9	88.6
OAD	**92.7**	**89.3**
Teacher	92.9	89.3
Ensemble	93.2	90.0

dynamic graphs. The changes of a graph may be caused by node attributes variation or graph structure evolution. Corresponding to above two circumstances, we conduct two simulation experiments on PPI dataset. Firstly, we add random noises on each node's attribution (This experiment is denoted as "Dynamic-A"). The noise has zero means and frequencies ranging from 0.2 to 1.2. Then, we randomly remove certain proportions of edges on the graph (This experiment is denoted as "Dynamic-B"). The removed proportion ranges from 0.05 to 0.3. Consider that GNNs may fail to train after its edges being removed, we add self loops on the isolated nodes. We adopt GAT and GraphSAGE as the skeleton networks. Their architectures are summarized in Table 3. For knowledge distillation, we pretrain a much larger teacher GNN on original graph and then transfer its knowledge about logits distribution to a student GNN trained on dynamic graphs. For OAD and the single GNN, the training process is completely on dynamic graphs. Other experimental details are the same as Section 4.2. We compute the improvement of KD compared to the single student GNN (denoted as "KD_Improve") and improvement of OAD compared to the single student GNN (denoted as "OAD_Improve") as follows:

$$\begin{aligned} KD\_Improve &= F1\_Score(KD) - F1\_Score(single) \\ OAD\_Improve &= F1\_Score(OAD) - F1\_Score(single) \end{aligned} \tag{14}$$

where $F1\_Score(single), F1\_Score(KD), F1\_Score(OAD)$ are F1 scores of single GNN, KD and OAD, respectively. The results of "Dynamic-A" and "Dynamic-B" are presented in Figs. 2 and 3, respectively. From Fig. 2, we observe that as the graph changes more drastically, the value of "KD Improve" drops. When the frequency of random noise exceeds 0.6, the knowledge transferred from the static teacher is even harmful to the student. On the other hand, our proposed OAD model exceeds student GNN on all dynamic graphs, especially when the dynamic graph is greatly different from the original graph. Note that there are no explicit relationships between "F1 score improvement" and noise rate. The value of "F1 Score Improve" fluctuates with the increase of noise rate, as OAD model and student GNN may have different parameter initializations on different noise rate and edge removal proportion settings. From Fig. 3, we observe that for both GAT and GraphSAGE, the values of "KD Improve" are marginally around 0, indicating the knowledge obtained from the old graph is not effective on the new graph. Meanwhile, OAD maintains the superiority on the dynamic graphs. The reason behind this phenomenon may be that all students are trained on the graph of same dynamic frequency simultaneously in our model, they can capture different characteristics of the dynamic graph and knowledge consistency is retained.

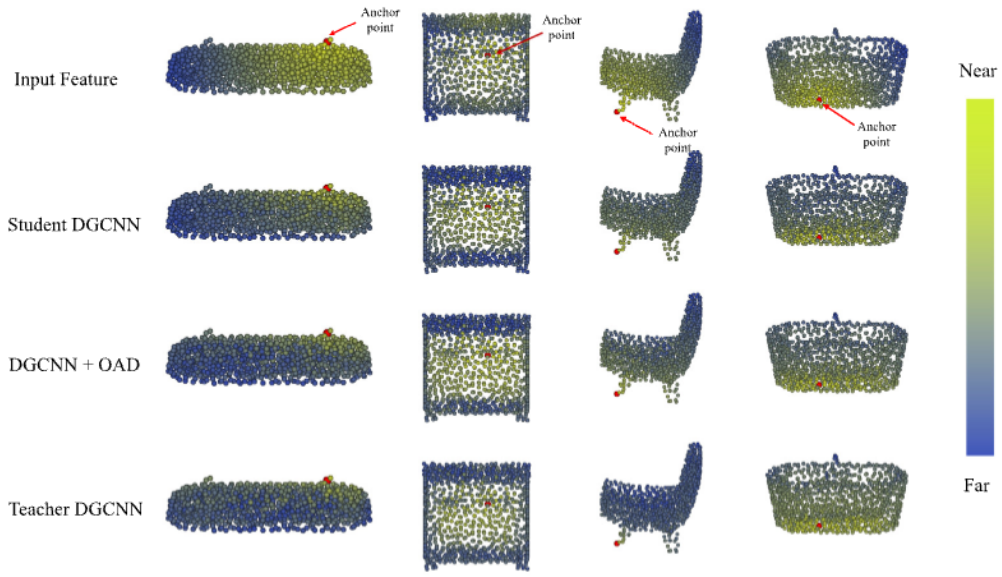

Fig. 4. Visualization of features learned by different models, deprecated by the distance between anchor points (colored in red) and other points. From left to right: original input features; features learned by student DGCNN; features learned by DGCNN with OAD framework and features learned by teacher DGCNN.

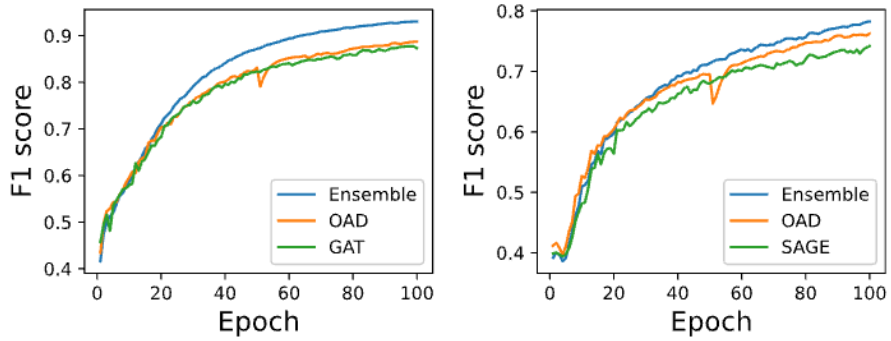

Fig. 5. The training F1 scores of Ensemble, OAD and a vanilla GNN on PPI dataset.

4.5. Benefit of mutual learning

In this section, we further analyze the benefit of group-based mutual learning. In Fig. 5, we compare the validation micro F1 scores of "ensemble", OAD and a vanilla GNN on the PPI dataset during the training phase. The experimental setup is the same as Section 4.2. We plot the validation F1 scores from epoch 0 to epoch 100. Our OAD model is warmed up from epoch 0 to 50 without group knowledge transfer. From 50 to 100 epoch, additional discriminators are introduced for mutual learning. It can be seen that the F1 scores of "ensemble" are consistently higher than the GNN trained with OAD, which indicates that the generated group knowledge may act like a "virtual" teacher to help improve the performance of the GNN model. We observe that the performance drops slightly at around epoch 50, because the parameters of newly attached discriminators are randomly initialized. After several epochs, as the discriminators learn the difference of GNN students' features, they can aid the training process.

4.6. Ablation study

To further evaluate the effectiveness of each component in our OAD framework, we perform an ablation study on the PPI dataset and ModelNet40 dataset. The results are presented in Tables 7 and 8. We analyze the contributions of each component as follows:

(1) w/o $\mathcal{L}_B$ refers to removing the global knowledge distillation module ($\mathcal{L}_B$) of the OAD framework. On the PPI dataset, the ablation of this module decreases the micro-F1 scores of GAT and GraphSAGE by 0.4% and 1.2%, respectively. As for the ModelNet40 dataset, DGCNN obtains 0.3% lower top-1 accuracy and 0.2% lower balanced accuracy without this module. This demonstrates that global knowledge distillation can bring benefits to the proposed framework.

Table 7
Ablation studies on PPI dataset.

Method	GAT	GraphSAGE
w/o $\mathcal{L}_B$	90.58 ± 0.35	77.10 ± 0.56
w/o $\mathcal{L}_G$	90.80 ± 0.34	77.08 ± 0.54
OAD	90.98 ± 0.46	78.30 ± 0.36

(2) w/o $\mathcal{L}_G$ refers to removing the local knowledge distillation module ($\mathcal{L}_G$) of the OAD framework. As a result, all the external discriminators are removed and knowledge transfer on node embedding space is prohibited. This setting decreases the performance of GAT by 0.18% and GraphSAGE by 1.22%, respectively. On the ModelNet40 dataset, after removing local knowledge distillation module, Top-1 accuracy and balanced accuracy of DGCNN drops by 0.3%. Thus, including the local knowledge distillation module can improve the performance of the OAD framework.

4.7. Comparisons of processing time

Compared to vanilla KD approaches, a pretrained teacher GNN model is not required by OAD method. Thus, OAD can reduce the

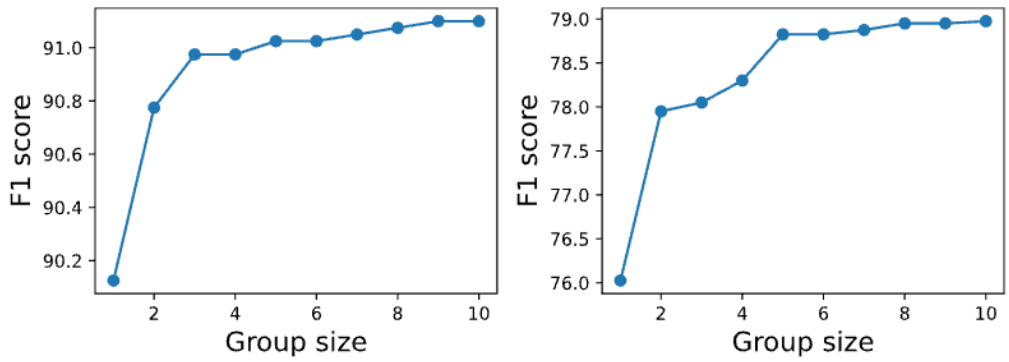

Fig. 6. The F1 scores on PPI dataset with different group size. Left: GAT. Right: GraphSAGE.

Table 8
Ablation studies on ModelNet40 dataset.

Method	Top-1 accuracy	Balanced accuracy
w/o $\mathcal{L}_B$	92.4	89.0
w/o $\mathcal{L}_G$	92.2	88.7
OAD	92.7	89.3

Table 9
Processing time of different methods on PPI dataset (seconds).

	GAT	GraphSAGE
Teacher	679.8	610.4
Student	154.5	126.1
OAD ($M = 2$)	320.9	285.2
OAD ($M = 4$)	526.1	468.0
OAD ($M = 6$)	894.4	825.5
OAD ($M = 8$)	1153.2	1061.8

Table 10
F1 score of GraphSAGE on PPI dataset w.r.t. various temperatures.

T	1.0	2.0	3.0	4.0	5.0	6.0
OAD	78.30	78.63	78.30	78.87	78.83	78.67
KD	76.67	76.50	76.28	76.03	75.97	76.43

Table 11
F1 score of GraphSAGE on PPI dataset w.r.t. various loss weights.

α	0.5	1.0	1.5	2.0	2.5	3.0
OAD	78.37	78.30	78.70	78.20	78.57	78.47
β	0.05	0.10	0.15	0.20	0.25	0.30
OAD	78.57	78.30	78.70	78.20	78.57	78.47

computation load and save the processing time. To verify this point, we compare the training time of KD and OAD on PPI dataset using the same experiment settings as Section 4.2. The time of data loading and evaluating models is excluded. The results are shown in Table 9. Note that the training time of KD methods is the sum of teacher's training time and student's training time. It is shown in Table 9 that the processing time of OAD is approximately 2/3 of the processing time of KD.

4.8. Sensitivity analysis

Group size. We evaluate the impact of group size (number of student GNN models) on the performance of our proposed OAD framework. Fig. 6 illustrates the performance concerning the group size ranging from 1 to 10. We observe that the F1 scores of GAT and GraphSAGE will generally increase as the student group is enlarged. We also observe that when the group size exceeds a specific threshold, the benefit becomes gradually marginal until convergence due to the capacity saturation.

Temperature. We then evaluate the influence of temperature T to our OAD method. We conduct the experiments on PPI dataset using GraphSAGE as the student model. The hyperparameters setting is same as Section 4.2 except temperature used in global knowledge distillation. The results are shown in Table 10. We observe that the F1 scores of OAD are substantially higher than vanilla KD method in all temperature settings. Thus, we conclude that proposed OAD method is not sensitive to the temperature chosen.

Loss coefficients. We finally evaluate the influence of loss coefficients α and β. We testify the F1 score of GraphSAGE model on PPI dataset. The hyperparameters setting is same as Section 4.2 except α and β settings. Note that $\alpha = 1$ is fixed when varying β and $\beta = 0.1$ is fixed when varying α. The results are shown in Table 11 We observe that the performance of OAD slightly fluctuates with different loss weights. Thus OAD is robust to loss weight selection.

4.9. Visualization

In this section, we visualize the features learned by different student GNN models. We extract the last layer of the student GNN models as features and visualize the distance between a randomly selected anchor point (colored in red) and other points on the model. Fig. 4 illustrates the Euclidean distance of points in the input space, from top to bottom are feature spaces of single student DGCNN model, DGCNN with OAD framework and the teacher DGCNN model. From left to right are five different samples in ModelNet40 dataset. We use the feature space of teacher DGCNN model as the reference. We can observe that the feature structure of the proposed model is similar to the feature structure of the teacher. This demonstrates the capability of the representations learned by the OAD framework.

5. Conclusions and future works

Knowledge distillation has become a promising technique to improve the performance of CNNs, under the assumption that teacher and student models are trained on the identical data distribution. However, since the topological structure and node attributes of graph data are likely to evolve, this fundamental assumptions of knowledge distillation may not hold for Graph Neural Networks (GNNs), leading to the sub-optimal solution. In this paper, we propose online knowledge distillation for graph neural networks to tackle this challenge. More specifically, we train a group of student GNN models simultaneously and with the guidance of a virtual dynamic teacher, the performance of each member is improved by distilling both the local and global knowledge. Adversarial cyclic learning is utilized to effectively and efficiently exploit the complicated information contained in the graph topology and node attributes. Extensive experiments on citation dataset, PPI dataset and ModelNet40 dataset demonstrate the effectiveness of the proposed OAD framework.

Future works. There are some possible future directions of this work. Firstly, Graph Transformer models have shown great potential on graph learning. Compared with GNN, Graph Transformers are heavier models, thus applying knowledge distillation on Graph Transformer is more meaningful than GNN. Therefore, we will study how to distill the knowledge of Graph Transformer. Secondly, although online knowledge distillation may alleviate the problem of distribution shift on graph data, its processing speed is slower as the group size increases. How to maintain the performance of OAD while making it more efficient should be studied in the future work.

CRediT authorship contribution statement

Can Wang: Writing – review & editing. **Zhe Wang:** Conceptualization, Methodology, Software, Writing – original draft. **Defang Chen:** Conceptualization, Writing – review & editing. **Sheng Zhou:** Conceptualization, Writing – review & editing. **Yan Feng:** Supervision, Funding acquisition. **Chun Chen:** Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The datasets used in this paper is publicly available.

Acknowledgments

This work is supported by the Starry Night Science Fund of Zhejiang University Shanghai Institute for Advanced Study (Grant No: SN-ZJU-SIAS-001), National Natural Science Foundation of China (Grant No: U1866602), National Natural Science Foundation of China (Grant No: 621062219), Zhejiang Provincial Natural Science Foundation of China under Grant No. LTGG23F030005, Ningbo Natural Science Foundation (Grant No: 2022J183) and the advanced computing resources provided by the Supercomputing Center of Hangzhou City University.

References

Bojchevski, A., Shchur, O., Zügner, D., & Günnemann, S. (2018). Netgan: Generating graphs via random walks. In *Proceedings of the international conference on machine learning* (pp. 610–619). PMLR.

Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2014). Spectral networks and locally connected networks on graphs. In *Proceedings of the international conference on learning representations*.

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Proceedings of the neural information processing systems*.

Chen, D., Mei, J.-P., Wang, C., Feng, Y., & Chen, C. (2020). Online knowledge distillation with diverse peers. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 3430–3437).

Chen, D., Mei, J.-P., Zhang, H., Wang, C., Feng, Y., & Chen, C. (2022). Knowledge distillation with the reused teacher classifier. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 11933–11942).

Chen, D., Mei, J.-P., Zhang, Y., Wang, C., Wang, Z., Feng, Y., & Chen, C. (2021). Cross-layer distillation with semantic calibration. *vol. 35*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 7028–7036).

Chen, T., Sui, Y., Chen, X., Zhang, A., & Wang, Z. (2021). A unified lottery ticket hypothesis for graph neural networks. In *Proceedings of the international conference on machine learning* (pp. 1695–1706). PMLR.

Chung, I., Park, S., Kim, J., & Kwak, N. (2020). Feature-map-level online adversarial knowledge distillation. In *Proceedings of the international conference on machine learning* (pp. 2006–2015). PMLR.

De Cao, N., & Kipf, T. (2018). Molgan: An implicit generative model for small molecular graphs. CoRR abs/1805.11973.

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. *vol. 29*, In *Advances in neural information processing systems*.

Denton, E. L., Zaremba, W., Bruna, J., LeCun, Y., & Fergus, R. (2014). Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems* (pp. 1269–1277).

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial networks. In *Advances in neural information processing systems* (pp. 2672–2680).

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of wasserstein gans. In *Proceedings of the neural information processing systems*.

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Proceedings of the neural information processing systems*.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. B. (2017). Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980–2988).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Henaff, M., Bruna, J., & LeCun, Y. (2015). Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. CoRR abs/1503.02531.

Hong, D., Gao, L., Yao, J., Zhang, B., Plaza, A., & Chanussot, J. (2020). Graph convolutional networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing, 59*(7), 5966–5978.

Hong, D., Yokoya, N., Chanussot, J., Xu, J., & Zhu, X. X. (2019). Learning to propagate labels on graphs: An iterative multitask regression framework for semi-supervised hyperspectral dimensionality reduction. *ISPRS Journal of Photogrammetry and Remote Sensing, 158*, 35–49.

Hong, D., Yokoya, N., Ge, N., Chanussot, J., & Zhu, X. X. (2019). Learnable manifold alignment (LeMA): A semi-supervised cross-modality learning framework for land cover and land use classification. *ISPRS Journal of Photogrammetry and Remote Sensing, 147*, 193–205.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of the international conference on learning representations*.

Lan, X., Zhu, X., & Gong, S. (2018). Knowledge distillation by on-the-fly native ensemble. In *Advances in neural information processing systems* (pp. 7528–7538).

Landrieu, L., & Simonovsky, M. (2018). Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4558–4567).

Liu, R., Fusi, N., & Mackey, L. (2018). Teacher-student compression with generative adversarial networks. CoRR abs/1812.02271.

Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., & Paul Smolley, S. (2017). Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 2794–2802).

Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. CoRR abs/1411.1784.

Nguyen, D., Gupta, S., Nguyen, T., Rana, S., Nguyen, P., Tran, T., Le, K., Ryan, S., & Venkatesh, S. (2021). Knowledge distillation with distribution mismatch. In *Machine learning and knowledge discovery in databases. Research track: European conference, ECML PKDD 2021, Bilbao, Spain, september 13–17, 2021, proceedings, part II 21* (pp. 250–265). Springer.

Park, W., Kim, D., Lu, Y., & Cho, M. (2019). Relational knowledge distillation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3967–3976).

Passalis, N., & Tefas, A. (2018). Learning deep representations with probabilistic knowledge transfer. In *Proceedings of the European conference on computer vision* (pp. 283–299).

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652–660).

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., & Bengio, Y. (2015). Fitnets: Hints for thin deep nets. In *Proceedings of the international conference on learning representations*.

Sepahvand, M., Abdali-Mohammadi, F., & Taherkordi, A. (2022). Teacher–student knowledge distillation based on decomposed deep feature representation for intelligent mobile applications. *Expert Systems with Applications, 202*, Article 117474.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Proceedings of the international conference on learning representations*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–9).

Tian, Y., Krishnan, D., & Isola, P. (2020). Contrastive representation distillation. In *Proceedings of the international conference on learning representations*.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2018). Graph attention networks. In *Proceedings of the international conference on learning representations*.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., & Solomon, J. M. (2019). Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics*, *38*(5), 1–12.

Wang, X., Zhang, R., Sun, Y., & Qi, J. (2018). KDGAN: Knowledge distillation with generative adversarial networks.. In *Advances in neural information processing systems* (pp. 783–794).

Wu, J., Leng, C., Wang, Y., Hu, Q., & Cheng, J. (2016). Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4820–4828).

Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How powerful are graph neural networks? In *Proceedings of the international conference on learning representations*.

Yang, Z., Cohen, W., & Salakhudinov, R. (2016). Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the international conference on machine learning* (pp. 40–48). PMLR.

Yang, C., Liu, J., & Shi, C. (2021). Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework. In *Proceedings of the web conference 2021* (pp. 1227–1237).

Yang, Y., Qiu, J., Song, M., Tao, D., & Wang, X. (2020). Distilling knowledge from graph convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7074–7083).

Zagoruyko, S., & Komodakis, N. (2017). Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *Proceedings of the international conference on learning representations*.

Zhang, Y., Xiang, T., Hospedales, T. M., & Lu, H. (2018). Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4320–4328).

Zhang, Y.-L., Zhou, J., Shi, Q., & Li, L. (2022). Exploring the combination of self and mutual teaching for tabular-data-related semi-supervised regression. *Expert Systems with Applications*, Article 118931.

Zitnik, M., & Leskovec, J. (2017). Predicting multicellular function through multi-layer tissue networks. *Bioinformatics*, *33*(14), 190–198.