Box Aggregation Distillation: Fractal Feature Aggregation for Model Distillation With IoT Potential

Can Cui, *Graduate Student Member, IEEE*, Dongyan Li, Zilong Fu, Sixiang Sun, Yuanyuan Li, Wu Deng, *Member, IEEE*, and Penghe Huang

***Abstract*—In the context of Internet of Things (IoT) deployments, edge nodes are commonly constrained by limited computational and storage resources, posing resource constraints as a key challenge in edge deployment strategies. Graph knowledge distillation (GKD) effectively extracts and transfers knowledge from large teacher models, enabling model compression and performance enhancement. This provides robust solutions for edge deployments in applications, such as autonomous driving systems, real-time fault detection, and wearable sensors. Nevertheless, current GKD methods have limitations in fully exploiting the multiscale information and deep feature potential of the networks, limiting the improvement of distillation performance. Therefore, we propose a novel graph neural network distillation method based on multiscale fractal feature extraction, termed box aggregation distillation (BAD). BAD divide the network into boxes based on different scales and aggregates features of the nodes within the box to extract the deep structure information. Furthermore, we introduced the Box Aggregation Loss to quantify the differences in intermediate layers between the two models. By minimizing this disparity, we effectively transfer multiscale and deep network knowledge from the teacher model. Experimental results indicate that the BAD method improves the F1 score by 0.51% to 5.83% compared to baseline methods while maintaining model compression. Additionally, it accelerates convergence speed during the warm-up phase by at least 2.96%.**

***Index Terms*—Box aggregation distillation (BAD), edge deployment, graph knowledge distillation (GKD), Internet of Things (IoT), multiscale fractal features.**

I. Introduction

THE RAPID advancement of deep learning has driven the development of the Internet of Things (IoT), facilitating the widespread adoption of neural network models in areas, such as smart cities, healthcare monitoring, industrial automation, and smart homes [1], [2], [3], [4], [5]. However, the deployment of these complex models on resource-constrained edge devices remains challenging due to their high computational costs. Knowledge distillation offers a solution for efficiently transferring knowledge from larger teacher models to smaller student models [6], [7], [8]. Large models perform batch processing on more powerful machines, generating checkpoint files. A smaller model is deployed on edge devices, performing reduced batch computations using checkpoint files. This facilitates the transfer of knowledge from the larger model while significantly alleviating the computational load on terminal devices. This technique, first introduced by Hinton et al. [9], and facilitates efficient knowledge transfer by regularizing the soft outputs between teacher and student models. Subsequently, methods such as FitNet [10] and attention transfer (AT) [11] further refine the distillation process by aligning the additional fully connected layers that are introduced in the intermediate layers and employing the interlayer attention mechanism, respectively. However, these methods primarily target convolutional neural networks (CNNs) for processing Euclidean data, and their effectiveness in processing graph data within non-Euclidean spaces remains limited.

Currently, graph neural networks (GNNs) [12] excel in processing unstructured data. Many fields are transforming various data types into graph structures and integrating multimodal technologies to achieve more comprehensive and accurate predictions [13]. Traditional GNNs have progressively demonstrated limitations in industrial applications. In response, numerous innovative and efficient GNN models have been developed, exhibiting significant innovations in both architecture and functionality. For instance, Liang et al. [14] proposed the Parallel CNN (P-CNN), which integrates time-domain and time-frequency domain features to enable efficient and robust bearing fault identification even when utilizing small datasets. Chen et al. [15] developed the interpretable time-frequency network (TFN), which embeds trainable time-frequency convolution layers within CNNs to enhance the performance and decision transparency of industrial fault diagnosis. Yan et al. [16] proposed the coarse-fine dual-scale time-frequency attention fusion network (CDTFAFN), which fuses vibration and acoustic heterogeneous signals with robust feature learning capabilities, thereby enabling high-precision and noise-resistant mechanical fault diagnosis in multisensor environments. Additionally, several other innovative GNN

Received 6 October 2024; revised 2 December 2024 and 19 January 2025; accepted 12 February 2025. Date of publication 19 February 2025; date of current version 9 June 2025. This work was supported in part by the National Natural Science Foundation of China under Grant U2133205. *(Corresponding author: Penghe Huang.)*

Can Cui, Dongyan Li, Zilong Fu, Sixiang Sun, Yuanyuan Li, and Penghe Huang are with the School of Railway Intelligent Engineering, Dalian Jiaotong University, Dalian 116028, China (e-mail: 15583367303@163.com; lidy@djtu.edu.cn; f17610136029@163.com; ssx1026@126.com; forkp@djtu.edu.cn; hph@djtu.edu.cn).

Wu Deng is with the College of Electronic Information and Automation, Civil Aviation University of China, Tianjin 300300, China (e-mail: dw7689@163.com).

Digital Object Identifier 10.1109/JIOT.2025.3543640

models have been proposed [17], [18], [19]. To fully leverage multisource knowledge, the scale of these GNN models continues to expand, thereby intensifying the challenges associated with deploying them on edge devices.

Graph-based knowledge distillation (GKD) has emerged as an effective solution to this problem by transferring knowledge from complex, high-parameter teacher GNNs to simpler, low-parameter student models, thereby efficiently compressing the model. Yang et al. [20] introduced the local structure preservation (LSP) method, the first successful application of knowledge distillation in GNNs, which captures local topology to enable effective knowledge transfer. The pioneering success of the LSP method spurred the development of subsequent graph-structure-based knowledge distillation techniques. Ma and Mei [21] introduced GraphNAS, which optimizes the output layer and intermediate feature representations through neural architecture search and knowledge distillation, while Joshi et al. [22] proposed a method to further enhance this process by adjusting the corresponding layers of the student model. The MUSTAD method, proposed by Kim et al. [23], mimics multiple graph convolutional network (GCN) layers within a single effective layer in the student model, thereby extracting the teacher's multihop feature aggregation knowledge and transferring the teacher's final hidden feature embeddings to the student. Wang et al. [24] proposed a novel domain-aware K-nearest neighbors (KNN)-KD method, which filters domain-relevant neighborhood knowledge during the distillation process for learning, thereby enhancing the distillation performance. Other notable approaches, including SCGCN [25], GCL [26], HGNN-D [27], and GNN-SP [28], have advanced GKD, thereby enabling the efficient deployment of GNNs on edge devices.

Existing GKD approaches typically focus on aligning output layers and intermediate features, but often neglect the rich graph structure information, including local and global topological features in both 2-D and 3-D spaces, as well as the distillation of information regarding the deep network structure. In fractal theory, researchers view networks in nature as complex systems with self-similarity and multiscale characteristics. A core feature of these networks is "self-similarity," where the network exhibits similar topological structures at different scales. These features are difficult to observe through superficial examination. Furthermore, traditional local feature aggregation methods make strong assumptions about data shape and distribution, limiting their effectiveness, particularly when dealing with complex, nonuniform, and hierarchical data structures. These limitations restrict the performance of student models in downstream tasks. Therefore, a comprehensive exploration of this complex information in graph structures is crucial for achieving efficient knowledge transfer from teacher models to student models.

This study introduces a novel model distillation method, termed box aggregation distillation (BAD). By aggregating nodes with fractal characteristics, BAD overcomes the limitations of traditional GKD techniques in handling node features and transmitting deep information. BAD utilizes a box feature aggregator (BFA) module to effectively capture and consolidate multiscale graph structural information. It not only preserves the internal relationships of the original graph, but also organizes and partitions the nodes, making them more cohesive within each box. This enhances the efficiency of knowledge transfer from the teacher model to the student model while preserving the graph's topological information and structural integrity. Additionally, a confidence threshold strategy is employed to minimize the impact of irrelevant feature boxes, thereby improving model performance. Extensive experimental evaluations on three benchmark datasets (PPI, Cora, PubMed) demonstrate that BAD achieves superior F1 scores compared to all baseline distillation methods and accelerates early convergence by 2.96% to 38.26%. Moreover, ablation studies confirm BAD's superiority in extracting fractal and deep network features.

The following are our contributions.

1) The fractal characteristics of the dataset networks were explored and analyzed. By examining the relationship between the minimum number of boxes and the radius r_b and the fitting error, we revealed the network's clear self-similarity.
2) Proposing a Fractal Feature-Based Aggregation Method, BAD. After partitioning the network on multiple scales and aggregating its features through BFA, the redundant information in the network is again removed through the confidence thresholding strategy, which enhances the representativeness of the core nodes and improves the distillation efficiency at the same time.
3) Extensive experiments on three benchmark datasets of varying degrees of difficulty demonstrate the effectiveness of BAD in node classification. The ability of BAD in extracting fractal and deep network features is also confirmed by ablation studies.

The structure of this article is as follows. Section II reviews GNNs and classical graph distillation methods; Section III presents the implementation details of the BAD method, including the renormalization process, the BFA aggregator, and the complete BAD methodology; Section IV provides a detailed exposition of the experimental results; and Section V concludes the research presented in this article.

II. Related Work

In this section, we provide a concise overview of the research areas pertinent to our study: GNNs and traditional GKD techniques.

A. Graph Neural Networks

GNNs have emerged as powerful tools for processing unstructured data in real-world applications and have been widely adopted across various fields. GNNs are typically categorized into spectral-based and nonspectral-based methods, depending on their information propagation mechanisms and feature update strategies. Spectral methods leverage graph spectral theory by applying Fourier transforms to the graph Laplacian operator, thereby defining convolution operations from a frequency-domain perspective. These methods primarily focus on capturing global frequency information [29], [30]. Conversely, spatial methods define

convolution operations directly within the graph's spatial structure. They update node representations by aggregating information from neighboring nodes, emphasizing local neighborhood details [31], [32], [33]. While existing GNN models excel in their respective domains, this article primarily focuses on the mechanisms of knowledge transfer between different GNN models, without introducing new GNN architectures.

B. Classical Distillation Algorithms Based on Graph Structures

Yang et al. [20] developed the LSP module, integrating knowledge distillation into GNNs. This pioneering work in traditional GKD facilitates the transfer of knowledge by effectively capturing local topological structures. A GNN at the kth layer can be represented as $\mathcal{G}^{(k)} = \{\mathcal{V}, \mathcal{E}, \mathcal{H}^{(k)}\}$, where $\mathcal{V}$, $\mathcal{E}$, and $\mathcal{H}^{(k)}$ denote the set of nodes, the set of edges, and the feature matrix at the kth layer, respectively. These nodes and edges have been transformed into learned embeddings. Additionally, we use $\mathbb{Z} = \{z_i \in \mathbb{R}^F \mid i = 1, 2, \ldots, n\}$ to represent the feature vectors of this intermediate feature map, where z_i denotes the feature vector of the ith node, $\mathbb{R}^F$ represents the F-dimensional real space formed by all the feature vectors, and i indexes the nodes from 1 to n.

The local structure is represented as a d-dimensional real vector $\mathbb{LS} = \{\mathrm{LS}_i \in \mathbb{R}^d \mid i = 1, 2, \ldots, n\}$, where d is the degree of node i. The formula for each element is given by [20]

$$\mathrm{LS}_{ij} = \frac{e^{\|z_i - z_j\|_2^2}}{\sum_{j:(j,i) \in \mathcal{E}} e^{\|z_i - z_j\|_2^2}} \tag{1}$$

where $\|z_i - z_j\|_2^2$ is defined as the squared Euclidean $L2$ norm of the difference vector between z_i and z_j, used to quantify their similarity in high-dimensional space. The feature values of all nodes pointing to the local central node are exponentiated, then normalized to convert them into a probability distribution. This enables LS_i to accurately represent the local structure of each node i using (1).

The local structures of the teacher model and student model are computed separately and denoted as LS^s and LS^t, respectively. The similarity distribution between the local structures of the teacher and student models is measured using Kullback–Leibler divergence (KL divergence), which can be expressed as [20]

$$S_i = \sum_{j:(j,i) \in \mathcal{E}} \mathrm{LS}_{ij}^s \log\left(\frac{\mathrm{LS}_{ij}^s}{\mathrm{LS}_{ij}^t}\right) \tag{2}$$

where S_i is inversely proportional to the local structure distribution.

Thus, by aggregating these similarity measures, an overall metric is obtained, referred to as the LSP loss, which is defined as [20]

$$\mathcal{L}_{\mathrm{LSP}} = \frac{1}{N} \sum_{i=1}^{N} S_i. \tag{3}$$

The total loss is composed as follows [20]

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda \mathcal{L}_{\mathrm{LSP}} \tag{4}$$

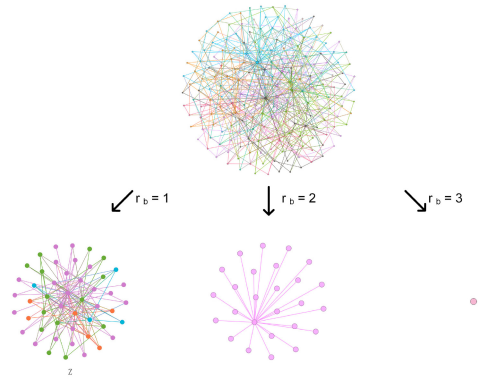

Fig. 1. Illustration of the box covering process.

where $\mathcal{L}_{\mathrm{CE}}$ represents the cross-entropy loss function between the predicted values and the true labels of the student model, and λ is a hyperparameter that balances these two losses. The standard backpropagation algorithm is used with (4) to converge the model and update its weights.

III. BAD Method

In this section, we introduce the proposed BAD method. To address the limitations of traditional graph structure distillation methods in exploring deep network information, we first employ a box partitioning module to segment the network. The process is shown in Fig. 1.

Next, we design a BFA that explicitly captures and aggregates high-confidence box structural information across multiple scales. The overall loss consists of two components: 1) the cross-entropy loss between the prediction vector generated by the student GAT and the true labels, and 2) the BAD loss derived from the BFA, which measures the difference between the teacher and student models. The general framework of the BAD approach is shown in Fig. 2.

A. Box Covering for Pretreatment

To initialize the graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, the initial mass m_i and candidate weight c_i for each $i \in V$ are set to 1. For each candidate central (hub) node i, the infection radius begins at 1 and increases incrementally by 1, progressively affecting its neighboring nodes. If a neighbor has not been visited, it is marked as visited, added to the new infection set, and the mass m_i is updated. This process is repeated until the radius of infection exceeds r_b. The formula for calculating the mass after infection is

$$m_i = \sum_{j \in N(i, r_b)} w_{ij} \tag{5}$$

where $N(i, r_b)$ represents the set of all nodes within the infection radius r_b of node i, and w_{ij} denotes the contribution of node j to the mass of node i. In most cases, w_{ij} is set to 1.

Among all candidate central nodes, the node with the highest mass is selected as the hub node, as shown as follows:

$$h_i = \arg\max_{i \in \mathcal{V}} m_i \quad \text{such that} \quad c_i = 1 \tag{6}$$

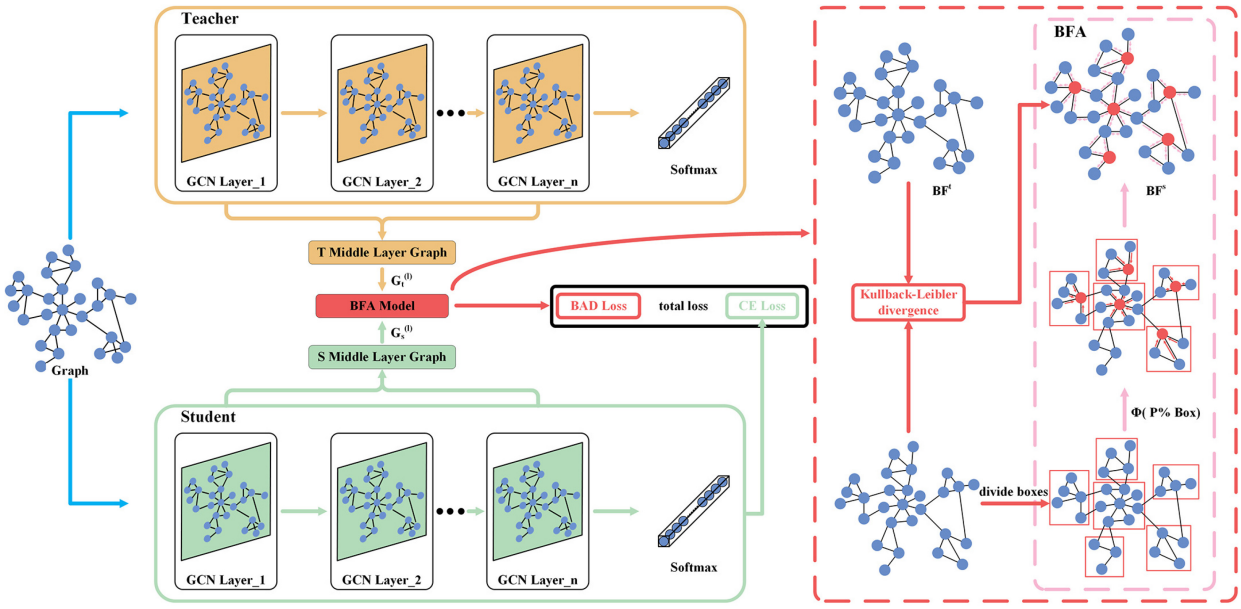

Fig. 2. Overall framework of BAD.

where node i must satisfy the condition $c_i = 1$, meaning that node i must be a candidate central node. The operation arg max represents "the argument that maximizes the value."

The subgraph consisting of the central node h_i and its neighbors within a radius of r_b is referred to as a box. This subgraph is removed from the graph $\mathcal{G}$, and the number of unassigned nodes is updated. This operation is expressed as

$$\mathcal{G} = \mathcal{G} \setminus \mathcal{B}_{h_i} \tag{7}$$

$$\mathcal{B}_{h_i} = \{j \mid \text{dist}(h_i, j) \ \leq r_b\} \tag{8}$$

where $\mathcal{B}_{h_i}$ represents the set (box) of all nodes within a radius r_b centered at node i. The symbol $\setminus$ denotes the set difference operation, i.e., the removal of the subgraph. The function $\text{dist}(h_i, j)$ indicates the shortest path distance between the hub node h_i and node j. This shortest path distance is calculated using the breadth-first search (BFS) algorithm, and is defined as

$$\text{dist}(h_i, j) \ = \min_{P \in \mathcal{P}(h_i, j)} \left\{ \sum_{(u,v) \ \in P} w(u, v) \right\} \tag{9}$$

where $\mathcal{P}(h_i, j)$ represents the set of all possible paths from the hub node h_i to node j, P is a specific path, (u, v) denotes an edge in path P, and $w(u, v)$ is the weight of edge (u, v). In an unweighted graph, the weight $w(u, v) \ = 1$, meaning the shortest path distance between nodes h_i and j is the path with the fewest edges.

By aggregating all hub nodes and their respective boxes, two sets are obtained, denoted as $\mathbb{H} = \{h_1, h_2, \ldots, h_k\}$ and $\mathbb{B} = \{\mathcal{B}_{h_1}, \mathcal{B}_{h_2}, \ldots, \mathcal{B}_{h_k}\}$. Here, $\mathbb{H}$ represents the set of all central nodes, and $\mathbb{B}$ represents the set of boxes, where each box is centered at a central node h_i with a radius of r_b.

B. BFA Method

Feature updates are performed on the intermediate output feature map at layer l, denoted as $\mathcal{G}^{(l)} = \{\mathcal{V}, \mathcal{E}, \mathcal{H}^{(l)}\}$, where $\mathcal{V}$ and $\mathcal{E}$ represent the sets of nodes and edges, respectively. The matrix $\mathcal{H}^{(l)} \in \mathbb{R}^{N \times F_l}$ represents the node feature matrix, with N being the number of nodes and F_l the dimensionality of the intermediate features at layer l. Let $\mathcal{B}_{h_i}$ be the box centered at the hub node h_i, with its node feature vector denoted as $\mathbf{f}_{h_i}^{(l)} \in \mathbb{R}^{F_l}$. The feature update for the hub node of the i th box is then expressed as

$$\mathbf{f}_{h_i}^{(l)*} = \mathbf{f}_{h_i}^{(l)} + \Phi \left(\{\mathbf{f}_j^{(l)} \mid j \in \bigcup_{\mathcal{B}_{h_i} \in \text{Top}_{p\%}(|\mathcal{B}_{h_i}|)} \mathcal{B}_{h_i}\} \right) \tag{10}$$

where $\mathbf{f}_{h_i}^{(l)*}$ is the hub node feature both after the update, and Φ denotes the feature aggregation function used to integrate the features within the selected box. Specifically, $\text{Top}_{p\%}(|\mathcal{B}_{h_i}|)$ represents the selection of the top $p\%$ of boxes after sorting them in descending order based on the number of nodes, where larger boxes have a more significant impact on the hub node and are thus considered more important. The optimal selection of confidence levels varies depending on the neural network model. Several approaches can be taken in selecting the aggregation function Φ, and in this article, we adopt the following aggregation strategies:

$$\Phi(\cdot) = \begin{cases} \sum_{j \in \mathcal{B}_{h_i}} \mathbf{f}_j^{(l)}, & \text{SUM} \\ \frac{1}{|\mathcal{B}_{h_i}|} \sum_{j \in \mathcal{B}_{h_i}} \mathbf{f}_j^{(l)}, & \text{MEAN} \\ \sigma\left(\sum_{j \in \mathcal{B}_{h_i}} a_{h_i j} \mathbf{W} \mathbf{f}_j^{(l)} \right), & \text{GNN.} \end{cases} \tag{11}$$

The three aggregation methods are: 1) summation aggregation (SUM); 2) mean aggregation (MEAN); and 3) aggregation based on GNNs. In the experimental section, we systematically compare these feature aggregation methods with those that do not use feature aggregation. For the weighted summation and mean aggregation methods, we applied strategies that aggregate either to the hub node or to all nodes within the box.

The GNN-based aggregation method employs a GCN with a single convolutional layer for feature aggregation.

After updating the features, we use $\mathbb{BF} = \{\mathbf{BF}_i \in \mathbb{R}^{F_l} \mid i = 1, 2, \ldots, n\}$ to represent the set of BFA feature vectors, where each feature vector can be expressed as

$$\mathrm{BF}_{ij} = \frac{\exp\left(-\lambda \cdot \|\mathbf{h}_i^{(l)} - \mathbf{h}_j^{(l)}\|_2\right)}{\sum_{n \in \mathcal{B}_{h_i}} \exp\left(-\lambda \cdot \|\mathbf{h}_i^{(l)} - \mathbf{h}_n^{(l)}\|_2\right)}. \tag{12}$$

In this context, $\|\mathbf{h}_i^{(l)} - \mathbf{h}_j^{(l)}\|_2$ represents the Euclidean distance used to measure the dissimilarity between nodes, where a smaller distance indicates higher similarity between nodes. A negative exponential function is applied to emphasize this property, ensuring that neighbors with smaller distances contribute more significantly to the feature update of the central node. The parameter λ serves as a tuning factor, controlling the sensitivity of the weight to the distance. Eq. (12) normalizes the weights within the node neighborhood $\mathcal{B}_{h_i}$, ensuring that the sum of the weights of all neighbors within the box for node i equals 1.

C. BAD Method

Since the input graph for both the teacher model and the student model is identical, for a given intermediate feature map $\mathcal{G}^{(l)}$, we compute the box features for both the teacher and student models, denoted as BF_t and BF_s, respectively. For each hub node h_i, the similarity between their box features can be calculated as

$$\mathrm{SIM}_i = \mathcal{D}_{\mathrm{KL}}\left(\mathbf{BF}_i^s \| \mathbf{BF}_i^t\right) = \sum_{j:(j,i) \in \mathcal{E}} \mathbf{BF}_{ij}^s \cdot \log\left(\frac{\mathbf{BF}_{ij}^s}{\mathbf{BF}_{ij}^t}\right) \tag{13}$$

where $\mathcal{D}_{\mathrm{KL}}$ represents the KL divergence between the feature distributions of the two models at node i.

This metric effectively captures the detailed differences in the complex feature structures within high-dimensional feature spaces, facilitating local feature alignment.

A lower SIM_i value indicates smaller differences between the two distributions in the feature space. By calculating the similarity across all nodes i, the box aggregation loss can be expressed as

$$\mathcal{L}_{\mathrm{BAD}} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{SIM}_i. \tag{14}$$

In addition to the box aggregation loss, the supervision for the student network can also come from the node label predictions made by the student model. The binary cross-entropy loss is used for model training

$$\mathcal{L}_{\mathrm{CE}}^s = \sum_{i=1}^{N} (y_i \log(\sigma(z_i)) + (1 - y_i) \log(1 - \sigma(z_i))) \tag{15}$$

where $\sigma(z_i)$ represents the Sigmoid function applied to the model output z_i. Therefore, the total loss for BAD can be expressed as

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}}^s + \mu \mathcal{L}_{\mathrm{BAD}} \tag{16}$$

Algorithm 1 Algorithm of BAD

Input: graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, box radius r_b, teacher GNN middle graph $\mathcal{G}_t^{(l)}$, student GNN middle graph $\mathcal{G}_s^{(l)}$, teacher GNN middle feature $\mathcal{H}_t^{(l)}$, student GNN middle feature $\mathcal{H}_s^{(l)}$, balance weight μ;

Output: Box aggregation loss $\mathcal{L}_{BAD}$;

1: Clone graph $\mathcal{G}$ to a new graph $\mathcal{G}_{\mathrm{clone}}$;
2: **while** $\mathcal{G}_{\mathrm{clone}}$ is not empty **do**
3: Set $m_i = 1$, $c_i = 1$, $\forall i \in \mathcal{V}$;
4: **for** $i = 1, 2, \ldots, n$ **do**
5: Compute node mass m_i, updating current peer nodes' mass according to (5);
6: **end for**
7: Select hub node h_i with maximum m_i according to Eq. (6);
8: Add $\mathcal{B}_{h_i}$ to set $\mathbb{B}$;
9: Remove $\mathcal{B}_{h_i}$ from $\mathcal{G}_{\mathrm{clone}}$ according to Eq. (7);
10: **end while**
11: **for** $\mathcal{B}_{h_i} = \mathcal{B}_{h_1}, \mathcal{B}_{h_2}, \ldots, \mathcal{B}_{h_k}$ **do**
12: Update teacher middle feature $\mathcal{H}_t^{(l)}$ for hub node $\mathbf{BF}_i^t$ according to Eq. (10) to Eq. (12);
13: Update student middle feature $\mathcal{H}_s^{(l)}$ for hub node $\mathbf{BF}_i^s$ according to Eq. (10) to Eq. (12);
14: **end for**
15: Add box aggregation loss according to Eq. (14) to $\mathcal{L}_{BAD}$;
16: **return** Box aggregation loss $\mathcal{L}_{BAD}$;

where μ is a hyperparameter that balances the contributions of the binary cross-entropy loss and the box aggregation loss. By applying this loss to the GNN model for backpropagation, the gradient descent mechanism of the neural network gradually reduces the gap between the teacher and student models, thus achieving effective knowledge transfer.

Algorithm 1 summarizes the BAD algorithm. First, we clone the original graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ to create a new graph $\mathcal{G}_{\mathrm{clone}}$. This step ensures that subsequent operations do not affect the data in the original graph (Line 1). For each node in the cloned graph, we set its initial mass $m_i = 1$ and candidate coefficient $c_i = 1$ (Line 3). Then, we iterate through the entire cloned graph, gradually calculating and updating the mass of each node m_i. Based on the calculated node mass m_i, the node with the highest mass is selected as the hub node h_i, and these nodes are aggregated into "boxes" $\mathcal{B}_{h_i}$ (Lines 4–9). Next, for each formed box $\mathcal{B}_{h_i}$, the intermediate features $\mathcal{H}_t^{(l)}$ and $\mathcal{H}_s^{(l)}$ of both the teacher GNN and student GNN are updated (Lines 11–14). Finally, the box aggregation loss $\mathcal{L}_{\mathrm{BAD}}$ is used to optimize the model's performance (Lines 15 and 16).

IV. Experiments

We evaluated the effectiveness of the BAD method through node classification tasks on three benchmark datasets. This section begins by introducing the datasets used in the experiments, followed by a detailed description of the baseline models, training configurations, and pertinent details. Finally, we provide an in-depth discussion of the experimental results.

TABLE I
DETALS OF DATASETS USED IN THE EXPERIMENTS

Data Sets	Nodes	Edges	Class	Feature Dimension	Train Set	Val Set	Test Set
Cora	2,708	5,429	7	1,433	140	500	1000
PubMed	19,717	44,338	3	500	180	500	1000
PPI	56,944	818,716	121	50	20	2	2

TABLE II
COMPARISON OF PARAMETERS OF STUDENT MODEL AND TEACHER MODEL ON THE DATASETS

Dataset	Epoch	Layers	Attention Heads	Intermediate Layer Features	Student Model Parameters	Teacher Model Parameters
Cora	100	3	2, 2, 1	16, 16, 7	0.047M	2.60M
PubMed	200	3	2, 2, 1	2, 2, 3	0.002M	1.60M
PPI	500	5	2, 2, 2, 2, 2	68, 68, 121	0.13M	2.59M

A. Datasets

We evaluated our approach on three real-world networks, Cora [34], PubMed [35], and protein-protein interaction (PPI) [36] datasets. These three networks are widely used in the field of graph learning and have become standard benchmark datasets for evaluating the performance of GNN models. The Cora and PubMed datasets have nodes for academic papers and edges for citation relationships between papers, while the PPI dataset has nodes for proteins and edges denoting interactions between them. These datasets vary in size, labeling dimensions, and classification tasks, thus allowing for a variety of studies in node classification and graph analysis. All datasets were partitioned into training, validation, and testing sets using standard dataset division methods. A summary of the dataset statistics is provided in Table I.

B. Baselines and Training Details

All datasets used in this article were implemented using the GAT model [32] under the PyTorch framework, serving as both the teacher and student models for the experiments.

The baseline methods used in this study are as follows.

1) *FitNet Method:* This method uses deep supervision, where intermediate layer features are set as targets to guide the student model in progressively learning the intermediate feature representations of the teacher model.
2) *AT Method:* This method guides the student model to focus on key feature regions by transferring information about the salient regions in the teacher model's attention mechanism.
3) *LSP Method:* The LSP technique is employed to transfer the local topological features of the teacher model to the student model, thereby enhancing its performance.
4) *MustaD Method:* By using a single effective layer to retain the multihop feature aggregation of deep GCNs, the knowledge of the teacher model is transferred to the student model. The final loss is given by

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda_{\text{emb}}\mathcal{L}_{\text{emb}} + \lambda_{\text{pred}}\mathcal{L}_{\text{pred}}. \tag{17}$$

5) *KNN-KD Method:* By incorporating a KNN module during the transfer process, the student model's generalization ability is enhanced by leveraging the similarity between instances.

TABLE III
TIME REQUIRED FOR BOX COVERING ON THREE DATASETS

Data Sets	$r_b = 1$	$r_b = 2$	$r_b = 3$	$r_b = 4$
Cora	0. 3569 s	0.3342 s	0.3148 s	0.4567 s
PubMed	16.5536 s	13.9161 s	16.2680 s	21.8852 s
PPI	0.1431 s	0.2535 s	1.6397 s	3.2754 s

We use a binary cross-entropy function as the loss function for all models. To fully validate the generalization ability of the models, a warm-up phase of 30 rounds was set up for all our training. This design aims to leverage knowledge distillation to prevent the student model from inefficient learning in the early stages of training, by smoothing the learning process and guiding feature learning, allowing the model to quickly transition into an effective learning state.

For all models, the learning rate was fixed at 0.005, the slope of the leaky ReLU was set to 0.2, and the batch size was 2. For the Cora and PubMed datasets, the dropout rate for both the input and output layers was set to 0.6, and the weight decay was set to 0.0001. In contrast, for the PPI dataset, these three values were set to 0. The teacher models all had a 3-layer architecture, with four attention heads in the input layer and six attention heads in the output layer, and a hidden dimension of 256. Although the student models had a deeper architecture, they used fewer attention heads and had smaller hidden dimensions. The specific configurations of the student models for different datasets are detailed in Table II. The student models for Cora, PubMed, and PPI datasets are approximately 55.32 $\times$, 800 $\times$, and 19.92 $\times$ smaller than their respective teacher models. Specifically, the λ_{emb} and λ_{pred} for MustaD are 0.01 and 0.1, respectively. In the KNN-KD method, the initial value of $k$is set to 5. All experiments were conducted on a single RTX 2080 Ti GPU.

C. Analysis of Fractal Features of the Datasets

We performed a basic fractal characterization of the three dataset networks used in the same way. By focusing on the PPI dataset as an illustrative example, five of the 24 protein networks were randomly chosen, and their fractal dimensions were evaluated using logarithmic coordinates, as shown in the Fig. 3. As the scale parameter r_b was incrementally increased, the network was fully encompassed by a box at $r_b = 4$, effectively renormalizing it into a single node. The linear

TABLE IV
PERFORMANCE OF DIFFERENT KNOWLEDGE DISTILLATION METHODS

Dataset	Method	Macro-F1 Score						Accuracy (%)		Macro-Precision		Macro-Recall	
		Epoch 0	Epoch 10	Epoch 20	Epoch 30	Final	ΔF1	Final	ΔAcc	Final	ΔPrec	Final	ΔRec
Cora	Teacher	/	/	/	/	68.60	/	/	/	/	/	/	/
	Student	15.70	18.20	21.80	32.30	64.50	2.79%↑	67.71	1.44%↑	66.68	1.76%↑	66.58	1.59%↑
	Fitnet	15.60	18.00	21.30	33.00	65.90	0.61%↑	67.92	1.12%↑	66.69	1.75%↑	65.33	3.54%↑
	AT	15.30	19.00	25.20	32.20	65.40	1.38%↑	67.92	1.12%↑	67.55	0.46%↑	66.71	1.40%↑
	LSP	15.40	19.10	21.70	32.10	65.90	0.61%↑	67.81	1.27%↑	66.76	1.65%↑	66.63	1.51%↑
	MustaD	15.60	18.40	22.30	32.90	65.96	0.51%↑	68.03	0.95%↑	65.58	3.47%↑	68.04	0.59%↓
	KNN-KD	15.00	17.00	22.70	33.30	65.20	1.63%↑	68.68	≈0%	67.12	1.11%↑	66.82	1.23%↑
	BAD($r_b = 1$)	**15.50**	**18.00**	**23.60**	**32.90**	**66.30**	/	**68.67**	/	**67.86**	/	**67.64**	/
	BAD($r_b = 2$)	15.50	17.90	24.10	32.10	66.10	/	68.45	/	67.18	/	66.42	/
PubMed	Teacher	/	/	/	/	84.93	/	/	/	/	/	/	/
	Student	40.36	54.99	60.15	61.03	76.36	1.49%↑	74.91	2.23%↑	80.09	1.79%↑	70.21	2.04%↑
	Fitnet	40.36	54.97	60.15	61.02	76.38	1.46%↑	76.38	0.26%↑	81.42	0.12%↑	71.33	0.43%↑
	AT	40.35	52.11	61.13	60.50	76.30	1.57%↑	74.93	2.20%↑	80.54	1.22%↑	69.65	2.86%↑
	LSP	40.44	57.80	58.24	62.74	76.86	0.83%↑	75.05	2.04%↑	79.93	1.99%↑	70.55	1.55%↑
	MustaD	37.99	42.00	47.29	50.48	78.21	0.91%↓	78.62	2.59%↓	82.34	1.00%↓	75.28	4.84%↓
	KNN-KD	40.33	39.46	44.30	45.33	73.23	5.83%↑	75.10	1.97%↑	79.96	1.95%↑	70.64	1.42%↑
	BAD($r_b = 1$)	**40.39**	**53.91**	**57.90**	**62.66**	**77.50**	/	**76.58**	/	**81.52**	/	**71.64**	/
	BAD($r_b = 2$)	40.38	57.01	58.60	63.30	77.49	/	76.38	/	81.42	/	71.33	/
PPI	Teacher	/	/	/	/	94.08	/	/	/	/	/	/	/
	Student	30.61	37.50	49.77	57.21	87.33	0.81%↑	/	/	89.17	0.76%↑	86.02	1.28%↑
	Fitnet	29.64	38.94	48.82	57.96	86.92	1.29%↑	/	/	88.59	1.42%↑	86.15	1.13%↑
	AT	31.66	43.40	52.77	56.23	86.70	1.54%↑	/	/	89.08	0.86%↑	86.96	0.18%↑
	LSP	30.92	43.53	54.25	63.14	87.59	0.51%↑	/	/	89.48	0.41%↑	86.61	0.59%↑
	MustaD	37.72	45.60	53.68	57.78	67.20	31.01%↑	/	/	56.02	60.39%↑	83.94	3.79%↑
	KNN-KD	30.48	45.08	56.85	64.01	87.33	0.81%↑	/	/	88.77	1.22%↑	85.93	1.38%↑
	BAD($r_b = 1$)	30.46	46.47	57.80	63.78	87.53	/	/	/	90.24	/	85.45	/
	BAD($r_b = 2$)	30.48	42.66	59.76	63.45	87.66	/	/	/	88.59	/	87.57	/
	BAD($r_b = 3$)	30.48	45.43	54.93	63.77	87.35	/	/	/	88.50	/	87.44	/
	BAD($r_b = 2$, 50%)	**30.48**	**45.09**	**57.42**	**64.45**	**88.04**	/	/	/	**89.85**	/	**87.12**	/

arrangement of data points on the logarithmic scale, coupled with high R^2 values, indicates self-similarity. Based on the fractal dimension formula $D_f = -(\text{slope}/\log b)$, the calculated fractal dimensions D_f range between 1.7 and 1.8. When applying the same methodology to the Cora and PubMed datasets, which are complete network graphs, the need for sampling was eliminated. These datasets were renormalized to a single node at a radius of 9, with fractal dimensions of 1.264 and 3.868, respectively. This is consistent with Barabási and Albert's self-similar network theory [37], indicating significant self-similarity and modularity within these networks.

The time used for box segmentation is shown in Table III. It is worth noting that the complexity of box covering is influenced by the scale and type of the dataset. Although the PubMed dataset is of moderate size, the relatively loose connectivity between its nodes results in longer box covering times compared to the Cora and PPI datasets.

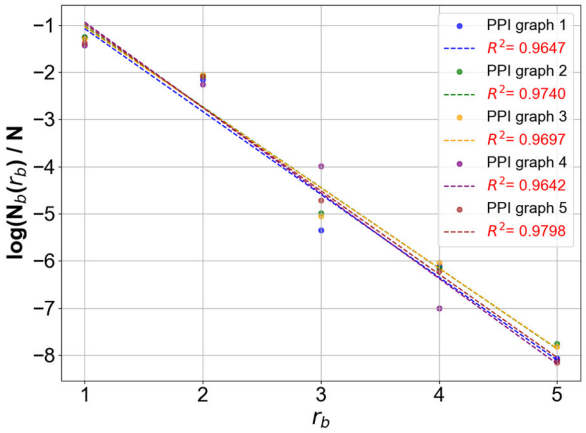

Fig. 3. Fractal dimension and fitting error of the networks in three datasets.

D. Comparison With Baseline Methods

To comprehensively evaluate the performance improvements of the BAD method in comparison to existing baseline methods, we conducted extensive experiments on three datasets. During the training of all models, the teacher and student networks were pretrained on a consistent training set. Table IV summarizes the improvements in various metrics of the BAD method compared to each baseline method and the independent student model. Student represents the results of training the student model using only the true labels, without using the teacher's knowledge. BAD refers to our method, with results shown in bold. For other methods, the best results are underlined. $\Delta F1_{\text{Cora}}$, $\Delta F1_{\text{PubMed}}$, and $\Delta F1_{\text{PPI}}$ indicate the improvement of the BAD method compared to the baseline methods.

The experimental results show that, except for the MustaD method on the PubMed dataset, the other datasets and methods demonstrate significant advantages. For the AT method, we used the attention formula $\alpha_{ij} = (\exp(\sigma(e_{ij}))/\sum_{k\in\mathcal{N}(i)}\exp(\sigma(e_{ik})))$ and applied LeakyReLU as the attention activation function. The Cora dataset employed the renormalization strategy with a scale of $r_b = 1$, the PubMed dataset used a scale of $r_b = 2$, and the PPI dataset adopted a scale of $r_b = 2$ with a 50% confidence level.

On the PubMed dataset, the MustaD method outperforms the BAD method. This is primarily because the PubMed dataset is highly sparse, which affects the feature aggregation performance of the BAD method, though this impact is acceptable. However, on the PPI dataset, MustaD experiences a sharp decline in F1 score and precision, while its recall remains high. This phenomenon is due to the imbalanced class distribution in the PPI dataset. In this context, MustaD introduces a significant amount of redundant information when using linear layers to match the dimensions of the student and teacher models, thereby affecting the overall performance of the model. Similarly, due to the class imbalance in the PPI dataset, the accuracy metric is no longer suitable for classification tasks on this dataset. Compared to the Student method, an improvement in standard deviation of approximately 4.78% indicates more stable performance across different datasets, especially when handling complex graph structures, such as those in the PPI datasets. Our method demonstrates greater performance improvements compared to other knowledge distillation methods, effectively extracting and transferring the local topological information of the teacher model, while also capturing deep information and multiscale structures, significantly enhancing the student model's performance.

E. Convergence Analysis of Different Distillation Methods

The core objective of knowledge distillation is to enable the student model to effectively learn from the teacher model. When the student model can converge quickly and capture the teacher's key information accurately, accelerating convergence can enhance model performance.

This section shows a further analysis of the convergence of the model in the warm-up phase. Since we utilized the teacher model's knowledge during the first 30 epochs for learning, we analyzed the convergence of each method by monitoring the model's performance throughout the warm-up phase across the three datasets, as shown in Figs. 4 and 5.

The experimental results show that the BAD model exhibits fast and stable convergence characteristics across all datasets. In the Cora dataset, as shown in Figs. 4(a) and 5(a), the BAD model with $r_b = 1$ achieves an average convergence speed during the warm-up phase that is 2.96% to 4.19% faster than other baseline methods. Figs. 4(b) and 5(b) shows that in the PubMed dataset, apart from having a similar convergence speed to the LSP method, the BAD model with $r_b = 1$ improves the average convergence speed by 7.79% to 10.52% compared to other baselines. As shown in Figs. 4(c) and 5(c), in the PPI dataset, the BAD model with $r_b = 2$ significantly enhances the average convergence speed by 5.43% to 38.26%.

TABLE V
F1 SCORES FOR EACH STAGE OF THE PPI NODE CLASSIFICATION TASK

Agg. Method	Origin	Epoch 10	Epoch 20	Epoch 30	Final
SUM	**30.46**	**46.47**	**57.8**	**63.78**	**87.53**
MEAN	30.46	42.77	56.65	60.72	87.04
GNNLayer_1	30.92	44.28	53.31	56.64	87.5
GNNLayer_2	30.52	38.76	53.17	57.31	87.37

TABLE VI
PERFORMANCE OF THE SUM AGGREGATION METHOD AT DIFFERENT SCALES ACROSS VARIOUS TRAINING STAGES, EVALUATED USING THE F1 SCORE

R-Scale	Origin	Epoch 10	Epoch 20	Epoch 30	Final
$r_b = 1$	30.46	46.47	57.80	63.78	87.53
$r_b = 2$	30.48	42.66	59.76	63.45	87.04
$r_b = 3$	30.48	44.43	54.93	63.77	87.35

These results demonstrate that our method effectively utilizes the knowledge from the teacher model, achieving faster and more stable convergence across multiple datasets. Particularly in complex datasets, the BAD model can quickly reach high performance in the early stages, proving its superiority in handling complex structures, especially in networks with fractal structures.

F. Ablation Experiments

In this section, we will examine the impact of the BAD method on the model from the perspectives of different aggregation strategies, scale radii, and confidence thresholds. We will also conduct ablation experiments to analyze the specific contribution of each strategy to the model's performance.

1) Comparison With Different Aggregation Methods: To evaluate different aggregation strategies, we employed summation, averaging, and GNN-based feature aggregation to aggregate intermediate layer node features in the PPI network, with experiments conducted at scale parameter $r_b = 1$. Table V presents F1 scores from the initial state (origin), through the warm-up phase (epochs 10-30), to the final phase. GNNLayer_1 and GNNLayer_2 denote single- and double-layer GNN aggregation. The optimal aggregation strategies are highlighted in bold.

Results indicate that initial F1 scores for all methods are relatively low (around 30), reflecting early learning stages. The SUM aggregation method shows the fastest convergence, significantly outperforming others during the first 30 epochs and achieving the highest F1 score in the final evaluation. SUM's average convergence rate in the first 30 epochs is 9.90% to 29.07% higher than other methods, suggesting it better captures the fractal network structure for effective feature representation learning.

The same methodology was applied to Cora and PubMed datasets to validate generality. SUM consistently demonstrated faster convergence and higher final F1 scores across all datasets, indicating robustness of the proposed method.

2) Effects of Scale on Model Performance: To investigate scale effects on model performance, we employ the SUM aggregation method as a baseline to evaluate performance under various confidence level strategies in the PPI node classification task. The results are summarized in Table VI.

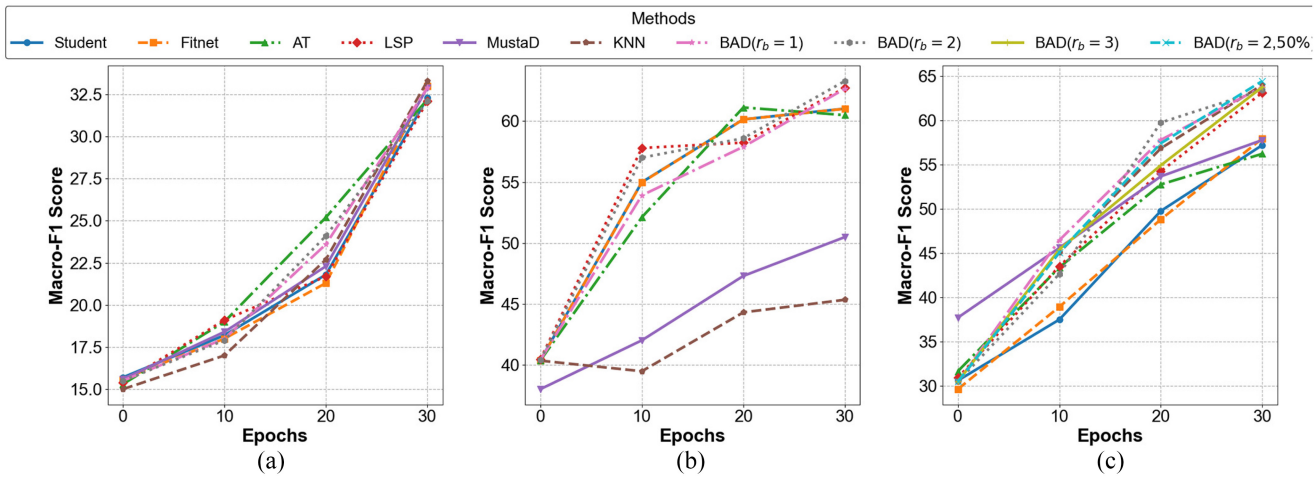

Fig. 4. Line plots of the convergence patterns of F1 scores over the first 30 epochs for various baselines in three datasets. (a) Cora. (b) PubMed. (c) PPI.

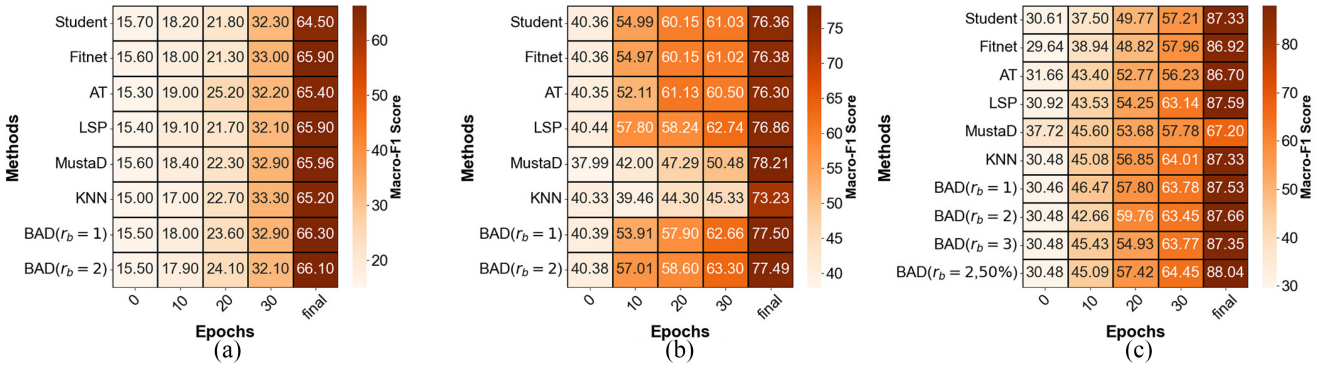

Fig. 5. Heat map of the convergence patterns of F1 scores over the first 30 epochs for various baselines in three datasets. (a) Cora. (b) PubMed. (c) PPI.

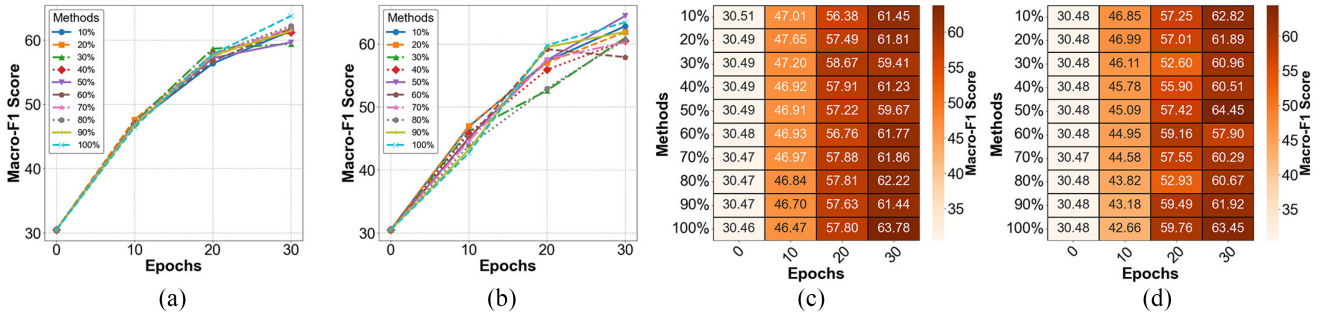

Fig. 6. F1 score for different confidence levels at scale in the PPI dataset. (a) Line graphs under scale $r_b = 1$. (b) Line graphs under scale $r_b = 2$. (c) Heat map under scale $r_b = 1$. (d) Heat map under scale $r_b = 2$.

Specifically, the $r_b = 1$ method shows a slight advantage in final F1 score, whereas $r_b = 2$ and $r_b = 3$ demonstrate faster convergence (3.9% and 2.3% on average, respectively). This implies the $r_b = 2$ scale has the greatest impact on model performance. At $r_b = 4$, the PPI network collapses into a single node, rendering aggregation meaningless.

In contrast, multiscale segmentation results on the Cora and PubMed datasets are less significant. Under equivalent conditions, $r_b = 1$ achieves optimal performance, while slightly underperforms. At larger scales, performance remains comparable to $r_b = 2$. This phenomenon arises due to the sparsity of graph structures in Cora and PubMed, leading to weak node connectivity. As r_b increases, more redundant information is aggregated. Therefore, complex adjustments to r_b are unnecessary, and small-scale aggregation is sufficient. In contrast, large-scale aggregation may actually impair model performance. Although the model's performance is sensitive to the scale parameter r_b, its value is actually limited and controllable. Even at smaller scales, the model's performance can still be improved.

3) Effects of Confidence Levels on Model Performance: The self-similar nature of the network results in a power-law distribution of box sizes during the box covering process, where most boxes contain fewer nodes, and a few contain

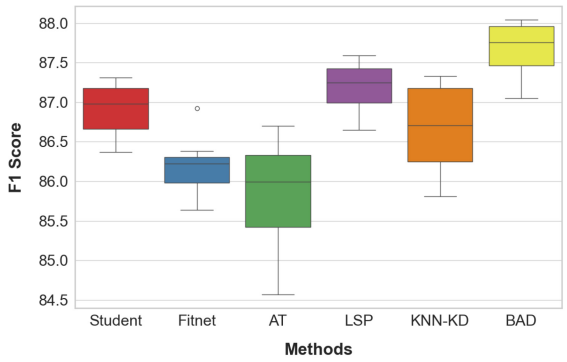

Fig. 7. Statistical tests of BAD method compared to baseline methods on the PPI dataset.

TABLE VII
SUMMARY OF SIGNIFICANCE TEST RESULTS FOR BAD AND BASELINE METHODS ON PPI DATASET

Method	Mean	SD	d	p (Bonferroni)	Significant
Student	86.907	0.356	0.898	3.793×10^{-4}	T
Fitnet	86.173	0.359	1.735	5.986×10^{-8}	T
AT	85.825	0.680	1.410	1.479×10^{-5}	T
LSP	87.208	0.303	0.574	1.589×10^{-2}	T
KNN-KD	86.661	0.570	0.903	1.002×10^{-3}	T
BAD	87.689	0.328	-	-	-

more. In this scenario, hub nodes of smaller boxes exhibit weaker aggregation capabilities, potentially affecting model performance. To evaluate performance at different confidence levels, we extracted boxes at varying levels and sorted them by size. The classification task on the PPI network was analyzed in 10% increments, with results shown in Fig. 6.

Fig. 6(a) and (b) shows that $r_b = 2$ is more sensitive to confidence than $r_b = 1$, indicating that the larger the scale, the more the learning efficiency will be affected by the predicting articulation-free point features. Fig. 6(c) and (d) illustrates convergence trends, with the model converging fastest and achieving the highest F1 score at scale $r_b = 2$ and 50% confidence. At scale $r_b = 3$, the reduced number of boxes limits the applicability of the confidence extraction method, suggesting that scale selection significantly affects learning performance, particularly at larger scales where fewer boxes constrain stability and performance.

Applying the same confidence selection strategy to the Cora and PubMed graphs yielded similar results to using 100% of the boxes. This is due to the PPI network's complexity and pronounced power-law distribution during renormalization, whereas the Cora and PubMed networks are simpler with minimal fractal characteristics. Thus, confidence selection is more effective in complex networks, with limited impact on simpler ones.

4) Significance Test: To validate the performance and stability of the BAD method, experiments were conducted on the PPI dataset using ten different random seeds for independent testing of each baseline method. A t-test was performed to assess the statistical significance of the results. To ensure the rigor of the statistical analysis, Bonferroni correction was applied to control the false discovery rate due to multiple comparisons. Notably, since the MustaD method performed poorly on imbalanced class samples, its results may not accurately reflect the method's true capabilities and were therefore excluded from the significance analysis. The experimental results are shown in Fig. 7 and Table VII. The t-test reveals that the performance differences between the BAD method and all baseline methods are statistically significant (p-values < 0.05). Furthermore, the effect size further quantifies the advantage of the BAD method, with a larger effect size indicating a more pronounced performance gap compared to the baseline methods.

V. CONCLUSION

In this article, we propose a novel fractal feature-based model distillation method, BAD, which aims to address the challenge of deploying models on edge devices. The performance of small-scale student models is enhanced by BAD through leveraging and transferring multiscale fractal knowledge along with inherent deep network information. Specifically, we conducted an in-depth analysis of the fractal characteristics of the datasets used. By observing the differences of networks at multiple scales, the superiority of the BAD method in capturing multiscale fractal features was demonstrated. Moreover, by minimizing the box aggregation loss, the student model is able to maximize its learning from the knowledge provided by the teacher model. The integration of aggregation and confidence threshold strategies helps to mitigate the negative effects of irrelevant features, thereby enhancing the transfer of knowledge. Extensive node classification experiments across datasets of varying difficulty and complexity show that, even with model size reductions ranging from 19.92 to 800 times, the student model's F1 score increased significantly by 0.51% to 5.83% compared to baseline methods. Additionally, the average convergence speed during the initial warm-up phase improved by 2.96% to 38.26%. Furthermore, ablation studies confirmed the impact of different scales on model performance and demonstrated that the BAD method effectively eliminates redundant information through aggregation and confidence threshold strategies, enabling more efficient transfer of deep and fractal structural knowledge. In IoT applications, this method offers significant advantages for edge device deployment due to its fast convergence speed, high accuracy, and high model compression ratio.

In future research, the BAD method will be further optimized by enhancing convergence speed, resource efficiency, scalability for IoT applications, and dynamic graphs. The effectiveness and interpretability of BAD will also be explored in transmembrane models. Furthermore, efforts will be made to overcome the high complexity associated with fully connected neural networks, with the goal of developing an efficient distilled neural network model that integrates the BAD method into the KAN network. These advancements will enable the BAD method to be deployed quickly and efficiently in various IoT scenarios.

REFERENCES

[1] S. A. Ali, S. A. Elsaid, A. A. Ateya, M. ElAffendi, and A. A. A. El-Latif, "Enabling technologies for next-generation smart cities: A comprehensive review and research directions," *Future Internet*, vol. 15, no. 12, p. 398, 2023.

[2] X. Li, H. Zhao, J. Xu, G. Zhu, and W. Deng, "APDPFL: Anti-poisoning attack decentralized privacy enhanced federated learning scheme for flight operation data sharing," *IEEE Trans. Wireless Commun.*, vol. 23, no. 12, pp. 19098–19109, Dec. 2024.

[3] Z. Zhu, X. Li, H. Chen, X. Zhou, and W. Deng, "An effective and robust genetic algorithm with hybrid multi-strategy and mechanism for airport gate allocation," *Inf. Sci.*, vol. 654, Jan. 2024, Art. no. 119892.

[4] J. Zheng, P. Liang, H. Zhao, and W. Deng, "A broad sparse fine-grained image classification model based on dictionary selection strategy," *IEEE Trans. Rel.*, vol. 73, no. 1, pp. 576–588, Mar. 2024.

[5] W. Deng, J. Wang, A. Guo, and H. Zhao, "Quantum differential evolutionary algorithm with quantum-adaptive mutation strategy and population state evaluation framework for high-dimensional problems," *Inf. Sci.*, vol. 676, Aug. 2024, Art. no. 120787.

[6] Y. Cao, Q. Ni, M. Jia, X. Zhao, and X. Yan, "Online knowledge distillation for machine health prognosis considering edge deployment," *IEEE Internet Things J.*, vol. 11, no. 16, pp. 27828–27839, Aug. 2024.

[7] E. S. Jeon, M. P. Buman, and P. Turaga, "Uncertainty-aware topological persistence guided knowledge distillation on wearable sensor data," *IEEE Internet Things J.*, vol. 11, no. 18, pp. 30413–30429, Sep. 2024.

[8] H. Chen, H. Zhao, Z. Zhang, and K. Li, "Discriminative feature learning-based federated lightweight distillation against multiple attacks," *IEEE Internet Things J.*, vol. 11, no. 10, pp. 17663–17677, May 2024.

[9] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[10] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," 2014, *arXiv:1412.6550*.

[11] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," 2016, *arXiv:1612.03928*.

[12] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[13] L. Waikhom and R. Patgiri, "A survey of graph neural networks in various learning paradigms: Methods, applications, and challenges," *Artif. Intell. Rev.*, vol. 56, no. 7, pp. 6295–6364, 2023.

[14] M. Liang, P. Cao, and J. Tang, "Rolling bearing fault diagnosis based on feature fusion with parallel convolutional neural network," *Int. J. Adv. Manuf. Technol.*, vol. 112, no. 3, pp. 819–831, 2021.

[15] Q. Chen et al., "TFN: An interpretable neural network with time-frequency transform embedded for intelligent fault diagnosis," *Mech. Syst. Signal Process.*, vol. 207, Jan. 2024, Art. no. 110952.

[16] X. Yan, D. Jiang, L. Xiang, Y. Xu, and Y. Wang, "CDTFAFN: A novel coarse-to-fine dual-scale time-frequency attention fusion network for machinery vibro-acoustic fault diagnosis," *Inf. Fusion*, vol. 112, Dec. 2024, Art. no. 102554.

[17] S. Li, J. Ji, K. Feng, K. Zhang, Q. Ni, and Y. Xu, "Composite neuro-fuzzy system-guided cross-modal zero-sample diagnostic framework using multisource heterogeneous noncontact sensing data," *IEEE Trans. Fuzzy Syst.*, vol. 33, no. 1, pp. 302–313, Jan. 2025.

[18] Y. Xu et al., "Cross-modal fusion convolutional neural networks with online soft-label training strategy for mechanical fault diagnosis," *IEEE Trans. Ind. Informat.*, vol. 20, no. 1, pp. 73–84, Jan. 2024.

[19] K. Feng, J. Ji, Q. Ni, Y. Li, W. Mao, and L. Liu, "A novel vibration-based prognostic scheme for gear health management in surface wear progression of the intelligent manufacturing system," *Wear*, vol. 522, Jun. 2023, Art. no. 204697.

[20] Y. Yang, J. Qiu, M. Song, D. Tao, and X. Wang, "Distilling knowledge from graph convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7074–7083.

[21] J. Ma and Q. Mei, "Graph representation learning via multi-task knowledge distillation," 2019, *arXiv:1911.05700*.

[22] C. K. Joshi, F. Liu, X. Xun, J. Lin, and C. S. Foo, "On representation knowledge distillation for graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 4, pp. 4656–4667, Apr. 2024.

[23] J. Kim, J. Jung, and U. Kang, "Compressing deep graph convolution network with multi-staged knowledge distillation," *PLos One*, vol. 16, no. 8, 2021, Art. no. e0256187.

[24] Z. Wang, S. Liu, X. Liu, M. Zhang, D. Wong, and M. Zhang, "Domain-aware k-nearest-neighbor knowledge distillation for machine translation," in *Proc. Find. Assoc. Comput. Linguist.*, 2024, pp. 9458–9469.

[25] Q. Song, J. Su, and W. Zhang, "scGCN is a graph convolutional networks algorithm for knowledge transfer in single cell omics," *Nat. Commun.*, vol. 12, no. 1, p. 3826, 2021.

[26] Y. Qian, Y. Zhang, Y. Ye, and C. Zhang, "Distilling meta knowledge on heterogeneous graph for illicit drug trafficker detection on social media," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 26911–26923.

[27] J. Wang, X. Wang, B. Jin, J. Yan, W. Zhang, and H. Zha, "Heterogeneous graph-based knowledge transfer for generalized zero-shot learning," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, 2021, pp. 1859–1866.

[28] S. Zhou et al., "Distilling holistic knowledge with graph neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10387–10396.

[29] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[30] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 3844–3852.

[31] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1025–1035.

[32] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[33] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" 2018, *arXiv:1810.00826*.

[34] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–93, 2008.

[35] G. Namata, B. London, L. Getoor, B. Huang, and U. Edu, "Query-driven active surveying for collective classification," in *Proc. 10th Int. Workshop Min. Learn. Graphs*, vol. 8, 2012, pp. 1–8.

[36] M. Zitnik and J. Leskovec, "Predicting multicellular function through multi-layer tissue networks," *Bioinformatics*, vol. 33, no. 14, pp. i190–i198, 2017.

[37] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.