Toward Multiple Specialty Learners for Explaining GNNs via Online Knowledge Distillation

1st Tien-Cuong Bui
Department of ECE
Seoul National University
Seoul, South Korea
cuongbt91@snu.ac.kr

2nd Van-Duc Le
Department of ECE
Seoul National University
Seoul, South Korea
levanduc@snu.ac.kr

3rd Wen-syan Li
Graduate School of Data Science
Seoul National University
Seoul, South Korea
wensyanli@snu.ac.kr

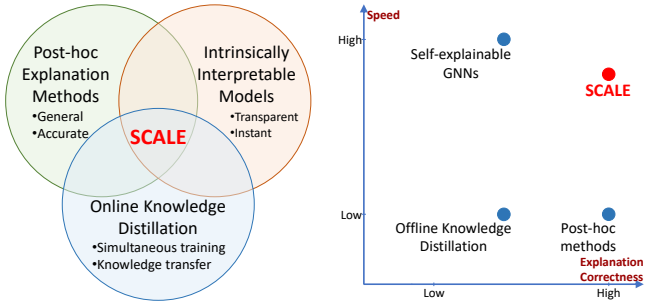
4th Sang Kyun Cha
Graduate School of Data Science
Seoul National University
Seoul, South Korea
chask@snu.ac.kr

***Abstract*—Graph Neural Networks (GNNs) have become increasingly ubiquitous in numerous applications and systems, necessitating explanations of their predictions, especially when making critical decisions. However, explaining GNNs is challenging due to the complexity of graph data and model execution. Despite additional computational costs, post-hoc explanation approaches have been widely adopted due to the generality of their architectures. Intrinsically interpretable models provide instant explanations but are usually model-specific, which can only explain particular GNNs. Therefore, we propose a novel GNN explanation framework named SCALE, which is general and fast for explaining predictions. SCALE trains multiple specialty learners to explain GNNs since constructing one powerful explainer to examine attributions of interactions in input graphs is complicated. In training, a black-box GNN model guides learners based on an online knowledge distillation paradigm. In the explanation phase, explanations of predictions are provided by multiple explainers corresponding to trained learners. Specifically, edge masking and random walk with restart procedures are executed to provide structural explanations for graph-level and node-level predictions, respectively. A feature attribution module provides overall summaries and instance-level feature contributions. We compare SCALE with state-of-the-art baselines via quantitative and qualitative experiments to prove its explanation correctness and execution performance. We also conduct a series of ablation studies to understand the strengths and weaknesses of the proposed framework.**

***Index Terms*—Graph Neural Networks, Explainable AI, Online Knowledge Distillation**

I. Introduction

As graph neural networks (GNN) [1, 2] have become increasingly ubiquitous in real-world applications and large-scale systems, understanding the causes behind predictions is essential for assessing trust, especially regarding critical decisions. However, generating explanations for GNN is challenging due to the following reasons. Unlike tabular data, text, and grid-like data, graphs consist of multiple components, such as graph structures, node features, and edge features. Therefore, influential factors of a certain prediction can be complicated interactions among these factors. Furthermore, graph datasets are diverse, wherein each may contain a different set of components, thus making it hard to measure exact contributions. Moreover, it is not easy to use general explainable AI (XAI) tools [3, 4] for explaining GNNs due to the complexity of graphs.



(a) Relationships of Approaches (b) Positioning Existing GNN Explanation Approaches

Fig. 1. SCALE inherits the best features from post-hoc explanation methods and intrinsically interpretable models for GNNs. It trains multiple specialty learners simultaneously with a black-box GNN to explain GNNs based on online knowledge distillation [5]. It can instantly provide accurate explanations for GNNs' predictions.

Lately, several approaches have been proposed for explaining GNNs [6], as there is no one-size-fits-all solution in XAI. These methods look at the problem from different angles and provide multiple ways to explain GNNs. Most existing approaches fall under the post-hoc explanation category and focus on instance-level explanations [6]. Among them, perturbation methods such as GNNExplainer [7] and PGExplainer [8] have been widely adopted since they introduced benchmark datasets for explanation tasks and achieved outstanding results. However, these methods need to train explanation models for target instances after training black-box GNNs resulting in additional computational resources and execution time. Unlike post-hoc methods, intrinsically interpretable (self-explainable) models such as GAT [9] can provide explanations for predictions instantly. Several novel self-explainable GNN architectures [10, 11] have been proposed based on distance-based objective functions. However, they usually evaluate their proposed methods using citation graphs [12], which lack generalization to other datasets. Moreover, similarity functions can be a computational burden in many scenarios when graphs are complex. Besides, existing methods mostly concentrate on structural explanations while overlooking feature attributions.

A promising but challenging research problem is training

explainers with a black-box GNN, wherein each explainer learns to analyze a subset of interactions within an input graph. This approach enables explainers to be as general as post-hoc explanation methods and as quick as self-explainable models when making explanations. However, designing an effective training paradigm, which allows explainable models to achieve equivalent performance as the black box, is not a straightforward problem. It is because explainable models' performances can be inferior in a standalone training mode resulting in untrustworthy explanations. One solution for this problem is online knowledge distillation [5], wherein a black-box GNN is regarded as a teacher, while explainable models are considered students. The teacher guides the learning processes of students via distillation losses.

We propose SCALE, the first explanation framework training multiple **S**pe**C**i**A**lty **L**earners to **E**xplain GNNs. As depicted in Figure 1a, SCALE combines the advantages of post-hoc explanation methods, intrinsically interpretable models, and online knowledge distillation. Following [7, 8], our explanation framework concentrates on finding important factors from graph structures, node features, and edge features that affect predictions the most. Here, we relax the complexity of the problem by discarding edge features. Since numerous interactions exist among these components, creating a single explainer to examine attributions is complicated. Therefore, SCALE trains multiple specialty learners to explain predictions in different aspects. A learner is a model that learns from a black-box GNN to capture a subset of interactions. As shown in Figure 2a, structural importance and feature transformation learners are trained simultaneously with the black-box GNN based on an online knowledge distillation paradigm. This paradigm enables the black-box GNN model to guide learners to achieve approximate predictive performance.

In the explanation phase, instant explainers output predictions with explanations in different aspects and levels based on special procedures. First, an edge selection procedure eliminates unimportant edges and nodes for a graph-level prediction using learnable masks of a structural importance learner. Second, a customized version of the random walk with restart (RWR) algorithm, which has been widely applied to recommender systems and search engines [13]–[16], is employed to provide structural explanations for node-level predictions. In our scenario, a target node that needs an explanation is considered a query vertex. A random walker includes neighboring nodes and edges in the explanation as it jumps through these elements. Moreover, we execute DeepLIFT [17, 18] on top of a trained multilayer perceptron (MLP) learner in the feature attribution module to provide instance-level feature attributions and overall attribution summaries. This library is selected since it is an effective and fast feature attribution method for deep neural networks.

Evaluating the efficiency of explanation methods is difficult since ground-truth explanations do not exist in most graph datasets. Therefore, we follow [7, 8] to formulate structural explanations as binary classification tasks, wherein influential nodes and edges of predictions are included in explanations. Then, we make quantitative and qualitative comparisons between SCALE and baselines on the correctness of structural explanations and execution performance. Moreover, we evaluate the efficiency of the feature attribution module using a real-world dataset with intelligible features. Since no ground-truth information exists, we compare our observations with findings from a state-of-the-art method [19] for evaluation purposes. Finally, a series of ablation studies are conducted to understand algorithms in SCALE better. Through extensive experiments and analyses, we conclude that SCALE is effective in providing explaining GNNs in both explanation capability and execution performance.

Our contributions are summarized as follows:

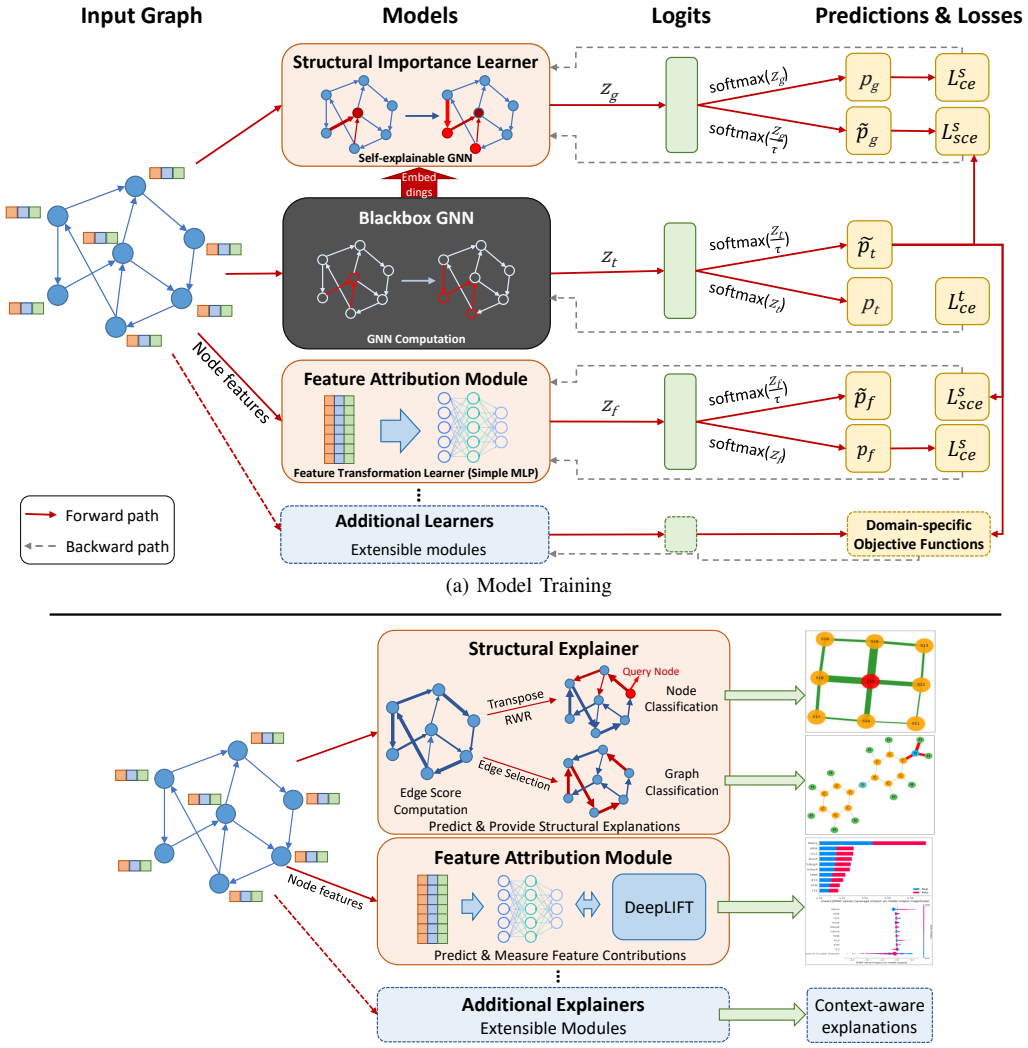
- We propose a novel explanation framework that utilizes multiple specialty learners to provide accurate explanations instantly for GNNs.
- Our proposed framework is general and can explain any message-passing-based GNN architectures. Moreover, it provides instant explanations for input graphs.
- We are the first to use RWR for explaining node-level predictions, which can specify different contributions of neighbors to a target node.
- We conduct extensive experiments and a series of ablation studies to demonstrate SCALE's superiority in explanation capability and execution performance.

Here is the structure of the paper. Section II describes related work. Section III presents background concepts. We describe the methodology in Section IV. Experiments are presented in Sections V and VI. We discuss detailed limitations, improvements, and possible applications in Section VII. Finally, we conclude our work in Section VIII.

II. Related Work

A. Post-hoc Explanation Methods

Most existing GNN explanation methods are post-hoc approaches, which consider pre-trained models black boxes. Lately, a good survey of these methods has been provided by Yuan et al. [6]. Instance-level explanation receives much attention from the research community, which includes four main topics: gradient-based, decomposition, surrogate, and perturbation. While the first three approaches try to fit the existing XAI methods to the graph domain [20, 21], the perturbation one, first proposed by Ying et al. [7], is an active research area with several following research papers [8, 22, 23]. The perturbation methods focus on finding important subgraphs via either edge mask learning or MCTS algorithms. However, these methods encounter overfitting issues due to the size of perturbed samples. They also do not consider feature attributions thoroughly. Moreover, post-hoc methods cannot provide instant explanations due to additional computations. Unlike them, SCALE trains specialty learners with a black-box GNN simultaneously to provide accurate explanations instantly without retraining. Besides, post-hoc explanation methods [7, 8] often transform node classification problems into graph classification problems via subgraph (K-hop) sampling. This approach is suboptimal when graphs



(a) Model Training

(b) Inference & Explanation

Fig. 2. An illustration of SCALE Architecture. The upper figure demonstrates the training process of SCALE, while the bottom one presents how SCALE provides explanations based on trained learners. Additional learners and explainers can be implemented depending on particular requirements.

contain numerous small cycles. SCALE implements different algorithms to explain node-level and graph-level predictions.

B. Self-explainable Graph Neural Networks

Intrinsically interpretable (or self-explainable) models can overcome the performance problems of post-hoc explanation methods. Specifically, self-explainable models can provide explanations instantly based on their trained weights. Graph Attention Networks (GAT) [9] can be considered a self-explainable model wherein attention matrices are used for constructing explanations. GCN-LPA [24] is another approach that integrates label propagation into GCN [25] and replaces the normalized adjacency matrix with a learnable matrix. Lately, several intrinsically interpretable GNN models have been proposed based on similarity functions [10, 11]. Zhang et al. [11] model the feature and label similarity in GNN execution. However, these methods aim to achieve high prediction accuracy on node classification datasets with the homophily property. Therefore, they may have inferior prediction performance on other datasets without this property, such as the ones in [7], leading to less trustworthy explanations. Moreover, explanation procedures in these models are either not discussed or too straightforward for general datasets. Even though [11] can explain GNNs using prototypes extracted during training, it is significantly slow. Our proposed method aims to provide instant explanations and can be applied to all message-passing GNN architectures.

C. Knowledge Distillation Methods for GNNs

Knowledge distillation (KD) [5] was originally a model compression method [26]. Lately, many explanation methods based on it have been proposed to make student models interpretable [27]. Most KD approaches for GNNs [28]–[31] are building small student models which can achieve or surpass the prediction accuracy of pre-trained teacher models. Students typically use the same datasets as the original model, as graph-free KD [28] is costly. Even though [30, 31] propose explanation approaches for GNNs, they merely concentrate on the

prediction accuracy of student models on citation graphs [12] or similar ones with the homophily phenomenon. Their proposed solutions for GNN explanations are too straightforward, making it hard to achieve significant results on other datasets, such as the ones in [7, 8]. Moreover, all methods implement offline knowledge distillation paradigms that require additional computational resources. To the best of our knowledge, we are the first to use an online knowledge distillation paradigm to train specialty learners to explain GNNs. Our proposed framework consists of multiple algorithms that explain node-level and graph-level predictions from different perspectives.

III. Preliminaries

An input graph $G = \{V, E, X\}$ consists of a set of vertices V, a set of edges E, and a d-dimensional node feature matrix $X = \{x_1, ..., x_n\}, x_i \in \mathbb{R}^d$. We study the problem of explaining GNNs in node and graph classification tasks.

A. Background on Graph Neural Networks

GNNs usually adopt message-passing paradigms which iteratively execute propagation, aggregation, and update to compute node embedding vectors. A message $m_{ij}^l = \text{Message}(h_i^{l-1}, h_j^{l-1})$, where h_i^{l-1} and h_j^{l-1} are the representation vectors of nodes i and j at layer $l-1$, respectively, is sent along the edge from j to i. At each node, GNNs aggregate information as follows: $m_i^l = \text{Aggregate}(m_{ij}^l | j \in \mathcal{N}_i)$. A new representation vector at layer l is computed using the following formula $h_i^l = \text{Update}(m_i^l, h_i^{l-1})$. The last layer's embedding vectors h^L are then used for downstream tasks such as node and graph classification.

Node Classification is a fundamental task in graph analytics, wherein the role is to assign correct labels to nodes in a graph. Let f denote a labeling function $f : V \mapsto \{1, ..., C\}$ that maps each node in V to a class in C. In GNNs, node embedding vectors h^L are fed to feed-forward networks to obtain node labels.

Graph Classification refers to the problem of classifying input graphs into groups. Let f denote a labeling function $f : G \mapsto \{1, ..., C\}$. A pooling layer is attached on the top of node embedding vectors h^L to output an embedding vector with the same dimension for an input graph. This vector is the input of feed-forward networks for classification.

B. Constructing Explanations for GNNs

A GNN's prediction $\hat{y}$ is made by a trained GNN model f operating on an input graph G_c. Let us denote G_c as the computation graph. In effect, we only need to analyze G_c to explain $\hat{y}$, meaning we should focus on finding essential subgraphs and crucial node features. Here, we ignore edge features for simplicity. Formally, SCALE constructs an explanation for a prediction $\hat{y}$ as (G_s, Φ_x), where G_s is a subgraph of G_c and Φ_x is an attribution vector of node features. Specifically, $\Phi_x = \{\phi_1, \phi_2, ..., \phi_d\}$ represents the contribution of node features to the prediction $\hat{y}$. Moreover, G_s's edges contain importance scores corresponding to their contribution to $\hat{y}$.

IV. Methodology

A. Framework Overview

We develop SCALE based on observations of existing GNN explanation methods. First, most post-hoc explanation methods aim at finding important factors that affect particular predictions from graph structures and node features. Therefore, they design explainable models based on prior knowledge of a pre-trained GNN, such as node embeddings or predicted labels. However, post-hoc computations hinder them from providing instant explanations. Even though intrinsically interpretable models can provide instant explanations without retraining, they are mainly based on model-specific approaches evaluated with simple datasets, thus lacking generalization to broader scenarios. Moreover, all methods mainly concentrate on structural explanations while overlooking feature attributions.

SCALE is a model-agnostic framework that provides instant explanations without retraining explainers. It is like an adapter with various transformation engines that convert message-passing-based GNN architectures into explainable versions. Specifically, it trains multiple specialty learners simultaneously with a black-box GNN to explain different aspects of an input graph based on an online knowledge distillation paradigm. After training, multiple instant explainers operate on trained learners to provide predictions and explanations simultaneously. For clarity, here are the brief descriptions of learners and explainers.

- **Learner**: a learner is a model guided by a black-box GNN model in training to capture special interactions in the GNN.
- **Explainer**: an explainer is a procedure that provides explanations for predictions based on a trained learner.

In the following subsections, we describe the training paradigm and present procedures for providing explanations. Then, we analyze the computational complexity of SCALE's internal engines in training and inference (including explanation) to demonstrate its capabilities and limitations.

B. Online Knowledge Distillation Paradigm

SCALE trains multiple learners to capture specific aspects of a black-box model, wherein each one is then used to construct explainers. For simplicity, we implement two learners for structural explanations and node feature attributions. As presented in Figure 2a, structural importance and feature transformation learners are trained together with a black-box GNN based on an online knowledge distillation paradigm, which is customized from [26]. Specifically, learners are students in the training scheme, while a black-box GNN model serves as the teacher to guide student learning. Moreover, SCALE separates students' computational paths so that they cannot affect the teacher's performance during training. The teacher model is trained with a cross-entropy loss function as follows:

$$\mathcal{L}_{ce}^t = -\frac{1}{N} \sum_{i=1}^{N} y_i \cdot \log(softmax(z_i^t)), \tag{1}$$

where z_i^t is the i^{th} output vector, and y_i is the i^{th} one-hot label vector. N denotes the number of training samples. Student models are trained with a joint objective function as follows:

$$\mathcal{L}^s = \mathcal{L}^s_{ce} + \lambda \mathcal{L}^s_{sce}, \tag{2}$$

where $\mathcal{L}^s_{ce}$ is a cross-entropy loss function, and $\mathcal{L}^s_{sce}$ is a soft cross-entropy objective function based on soft targets derived from predictive distributions. λ is a balancing factor regulating the amount of distilled information used in a student model. Here, the soft cross-entropy loss function is used instead of the KL divergence loss function due to practical performance achievements in experiments. $\mathcal{L}^s_{ce}$ and $\mathcal{L}^s_{sce}$ are as follows:

$$\begin{aligned} \mathcal{L}^s_{ce} &= -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log(softmax(z_i^s)), \\ \mathcal{L}^s_{sce} &= -\frac{1}{N}\sum_{i=1}^{N} softmax(\frac{z_i^t}{\tau}) \cdot \log(softmax(\frac{z_i^s}{\tau})), \end{aligned} \tag{3}$$

where z_i^s is the i^{th} output of the student model's prediction layer, and τ is the temperature term scaling information from corresponding models.

Batch normalization [32] is applied to multiple fully-connected layers in student networks to mitigate the changes in hidden outputs. Without batch normalization, the prediction accuracy of students is unstable in experiments leading to untrustworthy explanations. Algorithm 1 presents a joint training procedure of a black-box model and learners using the objective functions above.

Algorithm 1 Joint Training Procedure

Input: Training dataset $\mathbb{T}$, Number of epochs T
Output: f, g

1: **for** $epoch = 0 \rightarrow T$ **do**
2: f = train($\mathbb{T}$, $\mathcal{L}^t_{ce}$) {Train teacher f}
3: g = distill($\mathbb{T}$, f, $\mathcal{L}^s$)) {Distill knowledge to students}
4: **end for**

C. Providing Structural Explanations

Structural explanations are the main focus of most existing GNN explanation methods [6]. They aim to specify important nodes and edges to a prediction. For simplicity, we select GCN [25] as the black-box GNN model and describe how to design structural importance learners (self-explainable GNNs) for node and graph classification problems. The simplest form of a layer-wise propagation rule is as follows:

$$f(H^l, A) = \sigma(AH^lW^l), \tag{4}$$

where A is the adjacency matrix of an input graph, H^l is a representation matrix of nodes, W^l is a trainable weight matrix, and σ is a non-linear function.

Graph Classification. A self-explainable GNN is constructed by adding a mask matrix M to the propagation formula Equation (4) as follows:

$$f(H^l, A, M) = \sigma((A \odot M)H^lW^l). \tag{5}$$

Inspired by [8], the mask matrix is initialized via an MLP network in which inputs are edge embedding vectors constructed by concatenating embedding vectors of source and target nodes taken from the black-box GNN model. Specifically, each element of M is computed as follows:

$$m_{ij} = sigmoid(\text{MLP}([h_i, h_j])), \tag{6}$$

where h_i and h_j are representation vectors of the source and target nodes of an edge. A batch normalization layer follows each layer in the MLP model to mitigate the covariate shift problem caused by the weight updates of the black-box model. The last layer outputs probabilities of whether edges are selected or not. A self-explainable GNN model is trained with the black-box GNN using Algorithm 1, as illustrated in Figure 2a. In the inference and explanation phase, M is used to provide a structural explanation for a particular prediction of an input graph, wherein each element tells us whether the corresponding edge is important to the prediction or not.

Node Classification. Most existing methods use the same approach as graph classification problems to explain a node's prediction. Specifically, a subgraph including the target node is extracted and fed to an explainable model to obtain a structural explanation. Even though this approach can select and filter out uninfluential edges and neighbors, it cannot specify the exact contributions of these factors to a prediction.

We follow the intuition that influential nodes are usually close neighbors. Therefore, if a neighbor is important to the target node's prediction, the edge between them should receive a high importance score. RWR is an efficient method for calculating relevance scores between two nodes. Inspired by its efficiency in recommender systems [14]–[16], we implement a customized version to provide structural explanations for node-level predictions. Let us recall the formula of the algorithm as follows:

$$r_{t+1} = (1-d)r_0 + d\hat{\mathcal{A}}_c r_t, \tag{7}$$

where $\hat{\mathcal{A}}_c$ is a column-wise normalized transition matrix, r_t is a probability distribution vector at time t, r_0 is the initial probability distribution vector, and d is the probability that random walkers jump to a new state.

A target node can be considered a query node in the RWR algorithm for explaining node-level predictions, wherein its corresponding element in r_0 is initialized as 1, while others are set to 0. The transition matrix $\hat{\mathcal{A}}_c$ is the transposed version of a trainable adjacency matrix $\hat{\mathcal{A}}$ taken from a structural importance learner, as presented in Equation (8).

$$f(H^l, \hat{\mathcal{A}}) = \sigma(\hat{\mathcal{A}}H^lW^l) \tag{8}$$

If two nodes i and j have an edge, the corresponding element in $\hat{\mathcal{A}}$ is computed as follows:

$$\hat{a}_{ij} = softmax(\text{MLP}([h_i, h_j])), \tag{9}$$

where h_i and h_j are representation vectors of nodes i and j taken from the black-box GNN, respectively. The softmax function is executed on each row of matrix $\hat{\mathcal{A}}$. A converged

vector r tells us the importance of nodes in the input graph to the target node's prediction. To visualize an explanation, we sample an important subgraph, including the target node and top k nodes selected by ordering probabilities in r. Note that these k vertices can include neighbors in multiple hops. The complete procedure is described in Algorithm 2.

Algorithm 2 Querying Influential Vertices and Edges of a Node's Prediction

Input: A target node v
Pre-trained Matrix $\hat{\mathcal{A}}$,
Num. of iteration T,
Jumping probability d,
Num. of Response Nodes k
Output: An explanation of v's prediction

1: $\hat{\mathcal{A}}_c$ = transpose($\hat{\mathcal{A}}$)
2: $\mathcal{P}_V$ = RWR($v, \hat{\mathcal{A}}_c, T, d$) {Node scores}
3: $\mathcal{P}_E$ = diag($\mathcal{P}_V$) $\cdot \hat{\mathcal{A}}$ {Edge scores}
4: $\mathcal{R}_V$ = top_k($\mathcal{P}_V, k$) {Top k neighbors}
5: visualize($\mathcal{P}_V, \mathcal{P}_E, \mathcal{R}_V$) {Visualize the explanation}

D. Feature Attribution Analysis

In many real-world scenarios, meaningful node features significantly benefit machine learning models. Masking techniques proposed by Ying et al. [7] cannot clarify the exact contributions of features to predictions. Due to the complexity of message-passing patterns, it is difficult to utilize general XAI tools [3, 4] to measure feature attributions. Moreover, we cannot examine feature attributions using transformation matrices of a pre-trained GNN since the model accuracy significantly drops when the graph structure is ignored, causing inconsistent results.

To solve the problems above, we construct a feature attribution module to output attributions for a particular prediction and a contribution summary for a group of predictions. This module consists of a feature transformation learner, which is a simple MLP model trained with the black-box GNN using Algorithm 1. In training, Algorithm 1 allows the black-box teacher to guide the MLP student to achieve approximate predictive performance. DeepLIFT [17, 18] is then executed on top of the MLP model in the explanation phase to produce feature attributions for predictions. In reality, several methods, such as [33], can be integrated into SCALE since examining feature attributions is of great interest to the XAI research community. DeepLIFT is selected since it is an effective method for decomposing feature contributions in deep learning models and is very fast to compute.

E. Computational Complexity

Training. The computational cost grows linearly with the number of learners. Let N be the total learners, and the training time is approximately $N + 1$ times the time used for training a black-box GNN. Similarly, we need approximately $N + 1$ times GPU memory space for keeping models. We discuss acceleration methods for this limitation in Section VII. Next, the number of parameters in a self-explainable GNN equals the number of weights in the black-box GNN plus the number of parameters of an MLP used for computing edge weights. Similarly, the computational cost of the feature attribution module depends on the size of the student MLP model. Graphs are loaded only once into the GPU and shared among models to reduce memory consumption.

Inference & Explanation. We can use either the black-box model or explainable modules for inference since they achieve approximately the same accuracy. An explainable module provides both a prediction and an explanation together. Therefore, the computation cost is the cost of executing the predictive model plus the cost of making an explanation. For instance, the running time of Algorithm 2 is counted in constructing structural explanations for node-level predictions. Similarly, finding feature importances for a particular instance requires computation time for DeepLIFT execution.

V. Experimental Settings

A. Objectives

Our main focus was to present SCALE's correctness and execution efficiency. First, we compared SCALE with selected baselines to differentiate our framework from them via quantitative comparisons. We aimed to show that SCALE is better than post-hoc explanation methods in both aspects and is superior to intrinsically interpretable models on explanation correctness. Second, we compared our framework with two state-of-the-art post-hoc explanation methods [7, 8] in qualitative aspects to highlight the quality of explanations provided by SCALE. Third, we assessed the feature attribution module by confirming our observations on its provided results for the Amazon dataset with findings from a data mining-based method [19]. SCALE provided more detailed information in structural explanations and multi-level feature contributions compared to baselines. Finally, several ablation studies were conducted to verify our framework's efficiency from different aspects.

B. Datasets

We conducted experiments with five node classification and two graph classification datasets, as presented in Table I. Except for the Amazon dataset, others have been commonly used to perform functionally-grounded evaluations [34] of GNN explanation methods.

	#graphs	#nodes	#edges	#feat.	#labels
BA-Shapes	1	700	-	10	4
BA-Community	1	1400	-	10	8
Tree-Cycle	1	871	-	10	2
Tree-Grid	1	1231	-	10	2
Amazon	1	11.9K	351.2K	25	2
BA-2motifs	1K	25K	51.4K	10	2
Mutag	4.3K	131.5K	266.9K	14	2

TABLE I
Dataset Information. K means a thousand. The number of edges in the first four datasets varies in experiments.

Node Classification. Four synthetic graphs with ground-truth explanations, provided by [7], were used to evaluate the correctness and quality of structural explanations provided by methods. At each synthesis, the number of nodes was kept constant while the number of edges was varied. Specifically, BA-Shapes (BA-S) was constructed by attaching 80 five-node houses to a 300-node BA graph. BA-Community (BA-C) was created by joining two BA-Shapes graphs. Similarly, Tree-Cycle (Tree-C) and Tree-Grid (Tree-G) were generated by randomly attached cycle motifs and 3-by-3 grids to nodes in 8-level balanced binary trees, respectively. Since the node features of synthetic graphs had no meaning, the Amazon dataset [35, 36] was used to assess the feature attribution module. Specifically, it consisted of three graphs, wherein links were established based on mutual information between users. The goal was to detect fraudulent users based on given product reviews. Practically, we conducted experiments with all three graphs and selected a graph of users reviewing the same products, which produces the highest recall score, to measure node feature attributions.

Graph Classification. One synthetic graph and one real-world dataset were used. BA-2motifs (BA-2m) [8] consists of 1000 graphs with two classes constructed by adding specific motifs to BA graphs, where half contain 5-node house motifs and the other half include 5-node cycle motifs. Next, the Mutag dataset contains 4337 graphs classified into two classes based on their mutagenic effects. It also has ground-truth edge labels pointing out crucial subgraphs linking to mutagenic effects.

C. Baselines

We compared SCALE with five baseline methods on quantitative and qualitative results to prove the explanation correctness and execution performance. Specifically, baselines are categorized into intrinsically interpretable models and perturbation methods.

Intrinsically Interpretable Models use internal model weights to explain predictions directly. We selected four models to compare with SCALE as follows:

- **GCN-MLP:** GCN [25] is not an intrinsically interpretable model. Therefore, we replaced the normalized adjacency matrix with a trainable matrix similar to Equations (5) and (8) for graph and node classification, respectively. Learnable adjacency matrices were then used to provide structural explanations.
- **GAT [9]** can be considered a self-explainable GNN since attention heads capture interactions between nodes. We used three attention heads for each layer and averaged out all heads across layers to provide explanations.
- **SEGNN [10]** is a self-explainable GNN based on a similarity module that computes structure distances between an unlabelled node and K-nearest labeled neighbors. We set K such that the out-of-memory problems do not occur and the recall scores are maximized.
- **EGNN [31]** is a self-explainable model based on an offline KD paradigm [26], which filters out unimportant messages from 2-hop neighbors via two separate masking layers. To construct explanations, we aggregated neighbor scores and selected top K nodes that maximize recall scores.

Perturbation Methods integrate additional post-hoc training processes specialized for explanation purposes, which follow model-agnostic approaches. We selected two state-of-the-art methods similar to SCALE in problem formulation and explanation model initialization.

- **GNNExplainer [7]** was the first work that trained edge masks to determine crucial subgraphs based on an information theory approach. However, it has to retrain a mask for each target instance, thus making it less effective for inductive settings and large-scale graphs.
- **PGExplainer [8]** shared the same approach with GNNExplainer [7] but initialized masks using embedding vectors from the pre-trained model. Moreover, target instances share trainable weights.

Executing Algorithm 2 on GCN-MLP and GAT: The explanation querying algorithm can operate on various types of pre-trained GNNs as long as these models include normalized adjacency matrices representing interactions between nodes. We named the results of these integration processes SCALE-GCN-MLP and SCALE-GAT, respectively.

D. Evaluation Metrics for Structural Explanations

Following [7, 8], we also formulated explanations as binary classification problems, wherein edges in pre-defined ground-truth motifs were labeled as 1 (positive class), while the other edges belonged to the negative class. Moreover, previous methods used the AUC score as the evaluation metric since learned masks represent probabilities that edges are selected. Instead, we selected precision and recall scores for quantitative comparisons for the following reasons. First, Algorithm 2 cannot be evaluated by the AUC score. Second, we aimed to know the ratio of true positive and false positive edges in explanations varied in particular scenarios. Third, precision and recall scores gave us more information to judge explanation methods. In many cases, explanation methods can achieve high recall scores by including all ground-truth edges but still obtain low precision scores due to numerous false positive edges. Therefore, a good explanation method must provide explanation subgraphs that include all ground-truth edges and contain as few wrong edges as possible, resulting in high scores in both metrics.

$$\begin{aligned} \text{Precision} &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \\ \text{Recall} &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \end{aligned} \tag{10}$$

E. Configurations

We followed experimental configurations of [7, 8], which used 8:1:1 (train/validation/test) splitting strategy. For fair comparisons, we contacted the authors of GNNExplainer, PGExplainer, and SEGNN to request evaluation scripts for all datasets. Since we did not get responses, we did our

best to implement evaluation scripts based on their public source codes. We also followed their published papers and source codes to train baselines on datasets. Moreover, we used Youden's J Statistic for determining selection thresholds in baselines that output edge selection probabilities. According to [7, 8], explained instances were manually selected, regardless of which set they belonged to in training. Please refer to our GitHub code at https://bit.ly/SCALEGNN for more detail.

	MLP Layers	GCN Layers	Hidden Size	λ	Num. Epochs
Amazon	2	2	32	0.1	200
BA-Shapes	3	6	32	0.1	1000
BA-Community	3	6	64	0.1	1000
Tree-Cycle	3	6	64	0.1	1000
Tree-Grid	3	6	64	0.1	1000
BA-2motifs	3	4	64	4	200
Mutag	3	4	64	4	200

TABLE II
TRAINING HYPER-PARAMETERS

SCALE modules were trained with hyper-parameter settings presented in Table II. The hidden size represents the dimension of transformation matrices in GNN-based models and denotes the first layer's dimension in MLP models. The sizes of the last layers in MLP models depend on the model roles, which can be 1 in mask initialization or 2 in classification tasks. Furthermore, the hidden size was reduced by half after each layer. For instance, $[64, 32, 2]$ denotes an MLP with three layers, with the first layer containing 64 hidden units. Practically, we used the learning rate of 0.01 and set τ as 2 in all settings. The jumping probability d was set as 0.9 for Tree-Grid and 0.55 for others. In training, the magnitude of λ correlated with the amount of knowledge distilled from the teacher to the student. We studied the impacts of λ and d on explanation correctness in ablation studies.

We ran SCALE and baselines five times in each dataset using a machine with one NVIDIA Tesla V100 16GB GPU and reported the average results. We used PyTorch v1.10.2 and DGL v0.9.0 for building models in SCALE. Other baselines except PGExplainer were also executed using the same PyTorch version. We used Tensorflow v2.9.1 for experiments with PGExplainer. DeepLIFT was executed using a PyTorch API provided by [18].

VI. EXPERIMENTAL RESULTS

A. Quantitative Comparison with Baselines

First, we compared methods on the correctness of structural explanations. From Table III, we have the following observations. SCALE is superior to all baselines in explaining both node and graph predictions. Specifically, it achieves outstanding precision and recall scores in node classification datasets and outperforms state-of-the-art methods GNNExplainer and PGExplainer. GCN-MLP, GAT, and EGNN achieve high recall scores on BA-based datasets since ground-truth motifs only need at most 2-hop traversals. However, SEGNN performs poorly on these datasets since the number of sampling hops cannot be larger than one due to out-of-memory errors. In tree-based datasets, self-explainable models are inferior to SCALE due to the ineffectiveness of their explanation procedures. Perturbation-based methods follow sampling-then-choosing approaches, which can cause multiple false edges to be included in explanations. Conversely, SCALE achieves high precision scores since it expands explanation motifs from target nodes until meeting vertex thresholds corresponding to ground-truth motifs based on Algorithm 2. SCALE outperforms baselines on Mutag, wherein the precision score gains are 15.54% compared to PGExplainer and 51.52% in comparison with GNNExplainer. Its performance is also comparable to PGExplainer on the BA-2motifs dataset.

Secondly, we aim to demonstrate the superiority of SCALE in running performance. Table IV shows that SCALE is significantly faster than post-hoc explanation methods in all experiments. The performance gains are up to 94x compared to GNNExplainer and 120x in comparison with PGExplainer. Even though SCALE is slightly slower than self-explainable baselines in some scenarios, the gaps are insignificant. Moreover, this drawback is acceptable, considering SCALE's outstanding explanation scores compared with these methods.

B. Qualitative Comparison with Baselines

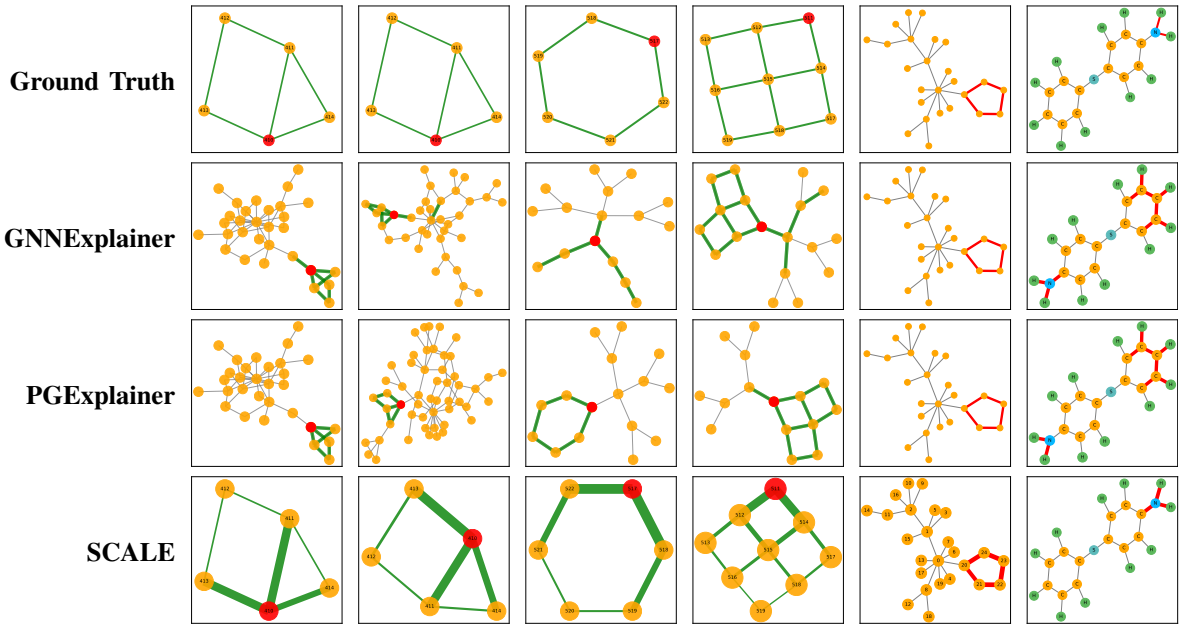
We chose one instance from each dataset and visualized explanations provided by SCALE, GNNExplainer, and PGExplainer in Figure 3. SCALE can highlight crucial edges in graph classification explanations similar to two baselines. Since SCALE achieves higher precision scores, its explanations include fewer false positive edges. Even though GNNExplainer and PGExplainer can highlight impactful edges of target nodes in node classification explanations, they cannot differentiate the contributions of these edges since edge weights only represent selection probabilities. Conversely, SCALE visualizes edges with different widths using the probability that a random walker jumps through these edges. This feature makes explanations highly intuitive, wherein the thicker the edges, the more important they are to target nodes. Moreover, we observed that edges started from direct neighbors within the same community received higher walk probabilities.

Multi-level expansion of explanations is useful in many scenarios, such as recommender systems. Even though GNNExplainer and PGExplainer can also present explanations on multiple levels by adjusting the visibility threshold, setting this value too low may result in outputs with multiple disconnected components due to the independence of edge selections. Edges started from direct (1-hop) neighbors may have lower selection probabilities than those from 2-hop or 3-hop neighbors. Moreover, the maximum number of hops is predefined at the sampling step before training explainable models. Therefore, we cannot expand explanations beyond this limit. Conversely, SCALE's explanations are more intuitive, as presented in Figure 4. As can be seen, close neighbors receive higher scores than distant ones. Moreover, one can expand explanations by adjusting the visibility threshold or displaying influential neighbors layer by layer without limitations.

	BA-Shapes		BA-Community		Tree-Cycle		Tree-Grid		BA-2motifs		Mutag	
	P	R	P	R	P	R	P	R	P	R	P	R
GCN-MLP	94.03	52.50	65.51	92.08	68.75	85.56	48.73	69.13	21.66	**100**	13.71	66.67
GAT	89.35	99.03	72.19	97.36	68.16	75.83	54.71	30.14	-	-	-	-
SEGNN	98.55	49.38	97.39	46.46	69.79	82.78	76.35	73.33	-	-	-	-
EGNN	56.55	99.44	42.73	93.89	62.17	66.67	77.85	63.03	-	-	-	-
GNNExplainer	80.44	**100**	59.27	**100**	73.92	87.34	75.18	53.79	26.46	94.18	14.66	71.49
PGExplainer	96.85	**100**	54.97	97.07	99.25	99.57	91.05	87.81	95.58	**100**	50.64	99.37
SCALE	**98.90**	**100**	**99.17**	**100**	**99.45**	**100**	**97.11**	**91.00**	**96.25**	**100**	**66.18**	**99.72**
SCALE-GCN-MLP	98.87	**100**	85.75	98.61	99.36	**100**	94.90	84.31	-	-	-	-
SCALE-GAT	98.63	**100**	91.89	94.44	83.59	79.02	84.11	64.84	-	-	-	-

TABLE III

QUANTITATIVE COMPARISON OF SCALE AND BASELINES ON STRUCTURAL EXPLANATION CORRECTNESS. SCALE IS SUPERIOR TO BASELINES, ESPECIALLY INTRINSICALLY INTERPRETABLE MODELS. EXPLANATION RESULTS ARE SIGNIFICANTLY IMPROVED WHEN EXECUTING ALGORITHM 2 ON GCN-MLP AND GAT. HERE, P IS SHORT FOR THE PRECISION SCORE, AND R DENOTES THE RECALL SCORE.



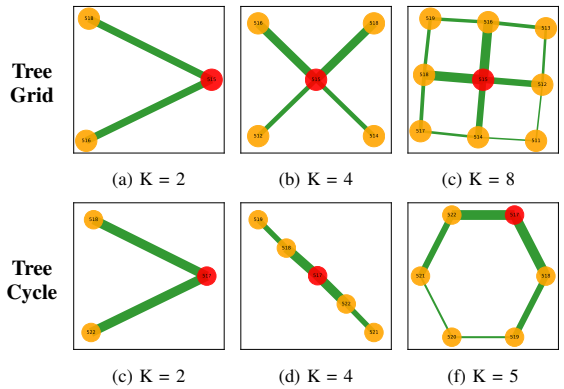
(a) BA-Shapes (b) BA-Community (c) Tree Cycle (d) Tree Grid (e) BA-2motifs (f) Mutag

Fig. 3. Qualitative Comparison between SCALE, PGExplainer, and GNNExplainer. Green/Red edges are the ones selected for explanations. The thicker the edges, the more important they are to predictions. In node classification datasets, only SCALE can highlight the importance of edges.

	BA-S	BA-C	Tree-C	Tree-G	BA-2m	Mutag
GCN-MLP	0.16	0.22	0.20	0.89	0.42	2.49
GAT	0.16	0.20	0.18	1.11	-	-
SEGNN	0.24	0.26	0.33	1.59	-	-
EGNN	13.52	19.60	15.43	23.05	-	-
GNNExpl.	40.79	40.77	34.11	155.35	107.42	630.42
PGExpl.	29.33	167.89	55.61	515.16	183.4	153.2
SCALE	**1.58**	**1.62**	**2.17**	**5.81**	**1.53**	**6.70**

TABLE IV

COMPARING EXECUTION TIME ON EXPLANATION SET BETWEEN SCALE AND BASELINES. RESULTS ARE MEASURED IN SECONDS.



(a) K = 2 (b) K = 4 (c) K = 8

(c) K = 2 (d) K = 4 (f) K = 5

Fig. 4. Multi-level Expansion of Structural Explanations for Node-level Predictions. K denotes the number of top influential nodes sorted by importance scores.

C. Efficiency of Explanation Querying Procedure for Node Classification

Here, we further studied the effectiveness of Algorithm 2 by executing it on adjacency matrices provided by GCN-MLP and GAT. Naively selecting important edges for node classification of target nodes was an inefficient approach, especially in complicated graphs. Therefore, the explanation results given by this approach on the original GCN-MLP and GAT were inaccurate as ground-truth motifs get complicated in Tree-

Cycle and Tree-Grid datasets. When applying Algorithm 2 to these models, explanation correctness increased significantly, as presented in the last two rows of Table III. However, the improved results were still lower than SCALE. Since edge weights represented the influences of neighbors, their correctness highly impacted the RWR procedure and the quality of explanations. Our proposed online KG paradigm enables a self-explainable GNN to capture interactions between nodes efficiently, thus resulting in more accurate edge weights.

D. Evaluating Feature Attribution Module

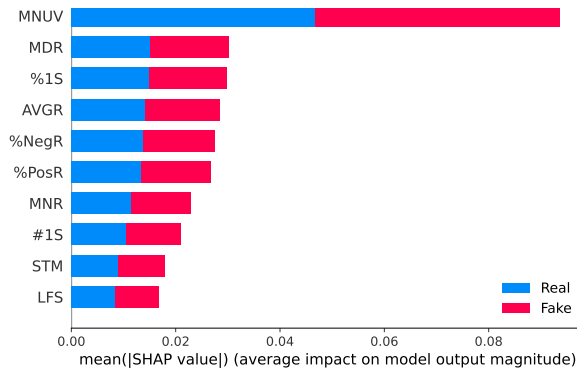
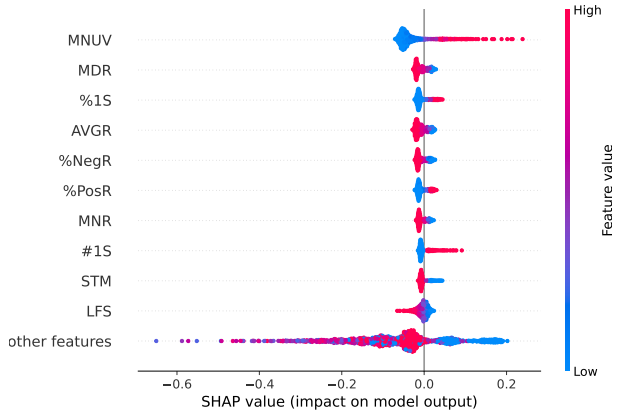


Fig. 5. An Overall Summary of Feature Attributions for Node Classification on Amazon Dataset. It summarizes the average impact of features on predictive probabilities. The longer the bar, the more influential a feature is.

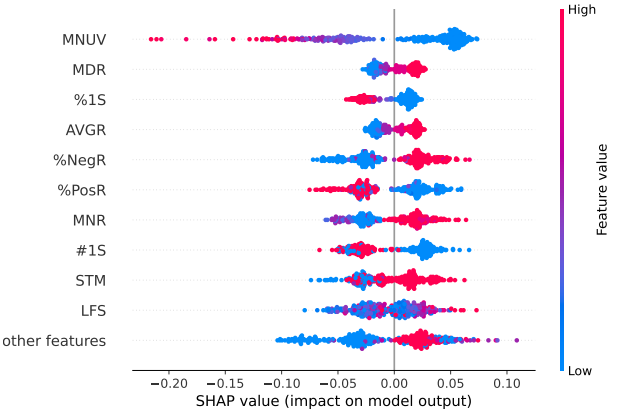
We aimed to evaluate the efficiency of the feature attribution module. The Amazon dataset was used since it contains intelligible node features. Each node in the Amazon graph has 25 statistical properties representing reviewing behaviors of users for products on the Amazon platform. Fraudsters or attackers are users who try to cheat the recommender system to promote particular products. Moreover, they also cover themselves by trying to be like benign users as much as possible. Since ground-truth explanations do not exist, we compared our observations on feature attributions provided by SCALE with insights discovered by Zhang et al. [19].

Here, overviews of feature contributions and examples of instance-level explanations are reported. Specifically, Figure 5 presents an overall summary of feature importances, while Figure 6 elaborates on relationships between feature values and their impacts on predictive probabilities. Then, two examples of instance-level explanations for each class are shown in Figure 7. SHAP values in the figures mean the magnitude of marginal probability contributions. Here are descriptions of some features presented in these figures: **MNUV** - Minimum number of unhelpful votes; **MDR** - Median of ratings; **%1S** - Ratio of 1-star votes; **AVGR** - Average of ratings; **%NegR** - Ratio of negative ratings; **STM** - Sentiment of feedback; **LFS** - Length of feedback. Please refer to [35, 36] for the complete description.

Based on the presented reports, we obtained the following observations. The minimum number of unhelpful votes plays a crucial role in model predictions, meaning clear differences exist between the voting patterns of two user classes. As

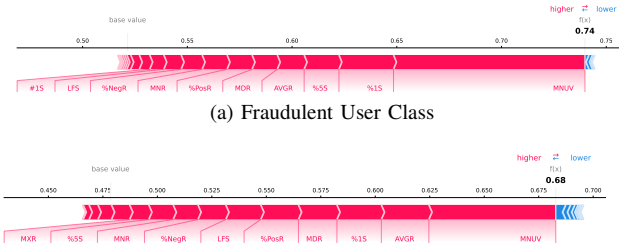


(a) Fraudulent User Class



(b) Benign User Class

Fig. 6. A Summary of Value-impact Relationships for Each Class in Amazon Dataset. Y-axes are feature names, and X-axes are feature impact values. The color bar in the bottom figures represents the magnitude of values (redder - bigger, bluer - smaller). All figures have the same order of features.



(a) Fraudulent User Class

(b) Benign User Class

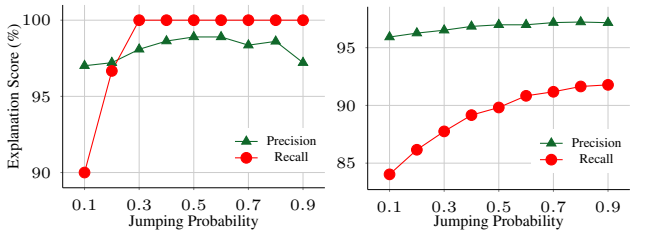
Fig. 7. Examples of Feature Attributions of Predictions for Each User Class. Red color bars present how much each feature increases the predictive probability, while blue color bars demonstrate the impact of features in the opposite direction.

Figure 6 shows, fraudulent users receive numerous negative votes from others since high **MNVU** values correspond to large probability contributions. We can also see that a large number/ratio of low-star ratings and feedback with negative sentiment increases the possibility that a user is fraudulent. Conversely, regular users give fair ratings and reviews with neutral or positive sentiments. Moreover, fake reviews' feed-

back summaries are usually shorter than real reviews. These observations match findings from Zhang et al. [19] that attackers usually give a high rating to a target item (promoted one) and low ratings to other regular items. Therefore, we can say that our proposed method for examining feature contributions is effective and accurate.

E. Ablation Studies

We conducted a series of ablation studies to study different aspects of SCALE. First, We aimed to select appropriate jumping probability values for particular scenarios. Then, we studied how explanation correctness responds to the changes in model accuracy. Next, different KD settings were evaluated to elaborate on the efficiency of the online KD paradigm. Finally, we studied how distilled knowledge impacts explanations in different classification tasks.



(a) Tree Cycle (b) Tree Grid

Fig. 8. Relationships Between Jumping Probability and Explanation Scores. A reasonable probability should be between 0.5 and 0.9.

Relationships Between Jumping Probability and Explanation Correctness. We have the following observations based on experiments conducted on Tree-Cycle and Tree-Grid datasets. A random walker tends to restart more often with a small probability, whereas it explores new states with a large value. As illustrated in Figure 8, small jumping probabilities cause low precision and recall scores, especially when many hops are required to complete the motifs. In the Tree-Cycle dataset, the precision score gradually improves as the jumping probability increases to 0.6 but decreases when the probability exceeds this value. In the Tree-Grid dataset, precision and recall scores correspond to the magnitude of the jumping probability since the complexity of grid motifs requires long walks to traverse all nodes in ground-truth explanations. Therefore, the large probability is appropriate for the Tree-Grid case, while the probability between 0.5 and 0.6 is better for the Tree-Cycle dataset. In practice, a reasonable value can range between 0.5 and 0.9 depending on the characteristics of graphs in particular scenarios.

Relationships Between Model Accuracy and Explanation Correctness. Most existing GNN explanation methods assume that pre-trained models are extremely accurate. Here, we studied how the explanation correctness is impacted when the model accuracy decreases in the Mutag dataset. As Figure 9 depicts, the precision score increases significantly as the model accuracy improves from 70% to 80%, meaning fewer false positive edges are included in explanations. When the

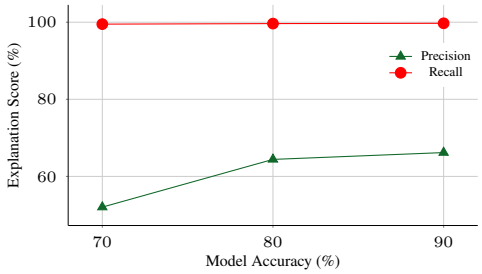
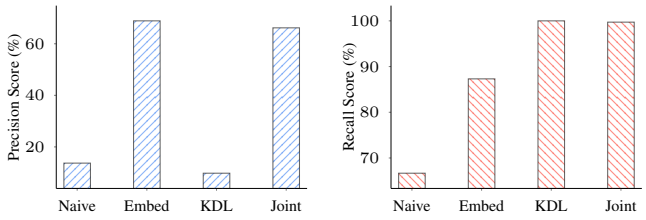


Fig. 9. Model Accuracy and Explanation Correctness Relationships on Mutag Dataset. The explanation correctness is proportional to the model accuracy.

precision accuracy increases to 90%, the precision score only improves slightly, suggesting that 80% is an acceptable value for extracting influential subgraphs in this dataset. We can conclude that explanations are more relevant and accurate as the model accuracy increases.

How effective is Online KD? This experiment specifies the effect of distilled knowledge, including embedding vectors and predictive distributions taken from the black-box GNN, on the correctness of structural explanations. We used the Mutag dataset for this experiment. We compared the experimental results of four settings on the self-explainable GNN as follows:

- **Naive**: The learner uses neither embedding vectors nor predictive distributions in training.
- **Embed**: The learner uses only embedding vectors to initialize learnable masks and sets $\lambda = 0$ in $\mathcal{L}^s$.
- **KDL**: The learner does not use embedding vectors to initialize learnable masks.
- **Joint**: The learner uses both components in training.



(a) Precision (b) Recall

Fig. 10. Explanation Correctness in Different Settings of the Online Knowledge Distillation Paradigm. SCALE provides the most accurate explanations when initializing a learnable mask with a black-box model's embedding vectors and using the KD loss for training learners.

Each setting was executed five times and reported the average result. As presented in Figure 10, precision and recall scores are high in the **Joint** case but significantly low in the **Naive** case. Furthermore, the explanation model achieved good precision in the **Embed** scenario, which was even better than the **Joint** case. In this case, many true positive edges were not selected in explanations causing high precision and low recall. Conversely, the learner could not learn to eliminate unimportant edges in the **KDL** setting, thus resulting in high recall and low precision scores. These results demonstrated that embedding vectors from the black-box GNN are essential

for mask initialization. Moreover, distilled information from the predictive distribution of the black-box GNN enhanced the learning process of the self-explainable GNN.

Studying the Balancing Factor. Here, we aimed to assess the effect of the balancing factor λ in the joint objective function of student models on explanation correctness. We conducted experiments with all datasets and selected results from the Tree-Cycle and BA-2motifs datasets to present in Figure 11 as variations in precision/recall scores were evident for them.

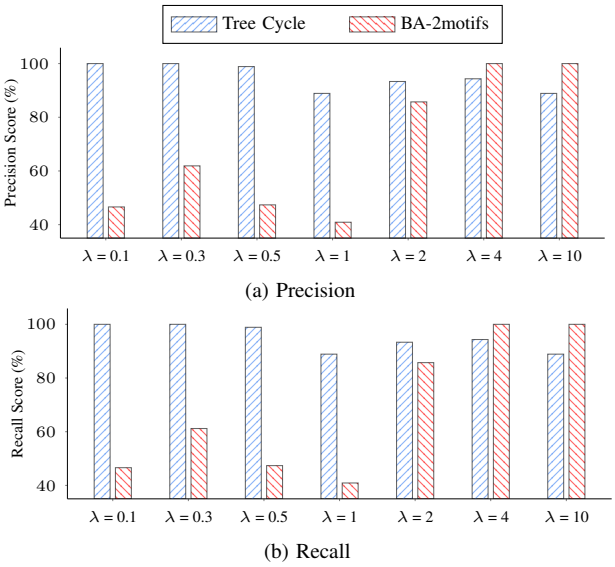


Fig. 11. Impact of Balancing Factor λ on Explanation Correctness. SCALE obtains accurate explanations with $\lambda \geq 2$ in graph classification problems and $\lambda \leq 1$ in node classification problems.

For graph classification datasets, the distilled knowledge from a black-box GNN plays a crucial role in training a self-explainable GNN model. Specifically, SCALE got better results with $\lambda \geq 2$. Even though the self-explainable model obtained high prediction accuracy, it could not provide accurate explanations when $\lambda \leq 1$. Furthermore, explanation correctness also slightly decreased when $\lambda \geq 4$, which means that too much information from the black box model is also not helpful for the explainable model.

Unlike graph classification, λ values smaller than one allowed SCALE to provide accurate explanations for node classification problems. The explanation correctness gradually increased as the balancing factor decreased when $\lambda \leq 1$. These results suggested that student models cannot deal with too much knowledge distilled from the teacher model.

VII. Discussion

A. Limitations and Future Improvements

Even though SCALE has several advantages over existing methods, it also contains some limitations. First, it does not consider the explanation problem in link prediction. This problem can be solved using the same approach as explaining node-level predictions. Second, learners and the black-box GNN model are trained in a single thread. In practice, we can accelerate training processes further via distributed settings. Third, the RWR algorithm is executed in a naive way in experiments that will suffer from long execution time for enormous real-world graphs. Since RWR is widely adopted in many large-scale systems, several techniques, such as [37], can be applied to accelerate this algorithm. Fourth, structural explanations for graph classification only show selected influential edges without detailed importance scores. We can further improve the quality of explanations by implementing game theoretic methods like [4] to compute exact edge contributions since we only need to consider combinations of crucial edges. Next, this paper does not study interactions among graph structures and node features due to their complexity. Methods such as [33, 38] can be integrated into SCALE to increase its explanation capability. Finally, as the number of learners increases with different explanation aspects, combining their outputs is a promising approach that allows the explanation framework to match users' preferences.

B. Applications

SCALE can be applied to numerous applications and systems since it can instantly provide accurate explanations. For instance, recommender systems [39, 40] can easily integrate SCALE into their engines to provide explanation functions to increase the transparency of their systems. SCALE can also provide valuable insights and accelerate research in several domains, such as human action recognition [41], bioinformatics [42], and to name a few. Moreover, feature attributions are beneficial in scenarios when graphs contain intelligible node/edge features. Therefore, SCALE is useful for easing the difficulties in finding insights from input features in graph analytics problems.

VIII. Conclusion

This paper presented SCALE, the first explanation framework that trained multiple specialty learners to explain GNNs since it was complex to construct one single explainer to examine attributions of factors in an input graph. We aimed to formulate explanation problems as general as post-hoc GNN explanation methods and achieve the explanation speed of self-explainable models. SCALE concentrated on identifying influential factors affecting model predictions from graph structures and node features. It provided explanations with more detailed information compared to existing methods. To achieve these goals, it trained specialty learners simultaneously with a black-box GNN model based on online knowledge distillation. After training, predictions and explanations were provided instantly, wherein several explainers examined different contributions of factors. Specifically, edge masking and random walk with restart procedures were implemented to provide structural explanations for graph-level and node-level predictions. Node feature attributions at different levels were provided by executing a fast feature attribution method on top of a trained MLP learner. Extensive experiments and ablation studies demonstrated SCALE's capabilities and performance superiority.

REFERENCES

[1] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[2] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2016, p. 1135–1144.

[4] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[5] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.

[6] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," *arXiv preprint arXiv:2012.15445*, 2020.

[7] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[8] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang, "Parameterized explainer for graph neural network," *Advances in neural information processing systems*, vol. 33, pp. 19 620–19 631, 2020.

[9] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *stat*, vol. 1050, p. 20, 2017.

[10] E. Dai and S. Wang, "Towards self-explainable graph neural network," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 302–311.

[11] Z. Zhang, Q. Liu, H. Wang, C. Lu, and C. Lee, "Protgnn: Towards self-explaining graph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 9127–9135.

[12] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma, Z. Huang, Q. Guo, H. Zhang, H. Lin, J. Zhao, J. Li, A. J. Smola, and Z. Zhang, "Deep graph library: Towards efficient and scalable deep learning on graphs." 2019.

[13] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[14] M.-F. Chiang, J.-J. Liou, J.-L. Wang, W.-C. Peng, and M.-K. Shan, "Exploring heterogeneous information networks and random walk with restart for academic search," *Knowledge and information systems*, vol. 36, no. 1, pp. 59–82, 2013.

[15] H. Park, J. Jung, and U. Kang, "A comparative study of matrix factorization and random walk with restart in recommender systems," in *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017, pp. 756–765.

[16] H. Wang, Z. Wei, J. Gan, S. Wang, and Z. Huang, "Personalized pagerank to a target node, revisited," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 657–667.

[17] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *International conference on machine learning*. PMLR, 2017, pp. 3145–3153.

[18] "Shap deepexplainer documentation," https://shap-lrjball.readthedocs.io/en/latest/generated/shap.DeepExplainer.html, (Accessed on 10/08/2022).

[19] S. Zhang, H. Yin, T. Chen, Q. V. N. Hung, Z. Huang, and L. Cui, "Gcn-based user representation learning for unifying robust recommendation and fraudster detection," in *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 689–698.

[20] F. Baldassarre and H. Azizpour, "Explainability techniques for graph convolutional networks," *arXiv preprint arXiv:1905.13686*, 2019.

[21] T. Schnake, O. Eberle, J. Lederer, S. Nakajima, K. T. Schütt, K.-R. Müller, and G. Montavon, "Higher-order explanations of graph neural networks via relevant walks," *arXiv preprint arXiv:2006.03589*, 2020.

[22] M. Sejr Schlichtkrull, N. De Cao, and I. Titov, "Interpreting graph neural networks for nlp with differentiable edge masking," pp. arXiv–2010, 2020.

[23] H. Yuan, H. Yu, J. Wang, K. Li, and S. Ji, "On explainability of graph neural networks via subgraph explorations," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 241–12 252.

[24] H. Wang and J. Leskovec, "Unifying graph convolutional neural networks and label propagation," *arXiv preprint arXiv:2002.06755*, 2020.

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

[26] G. Hinton, O. Vinyals, J. Dean *et al.*, "Distilling the knowledge in a neural network," 2015.

[27] R. Alharbi, M. N. Vu, and M. T. Thai, "Learning interpretation with explainable knowledge distillation," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 705–714.

[28] X. Deng and Z. Zhang, "Graph-free knowledge distillation for graph neural networks," 2021.

[29] C. K. Joshi, F. Liu, X. Xun, J. Lin, and C.-S. Foo, "On representation knowledge distillation for graph neural networks," 2021.

[30] C. Yang, J. Liu, and C. Shi, "Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework," in *Proceedings of the Web Conference 2021*, 2021, pp. 1227–1237.

[31] Y. Li, L. Liu, G. Wang, Y. Du, and P. Chen, "Egnn: Constructing explainable graph neural networks via knowledge distillation," *Knowledge-Based Systems*, vol. 241, p. 108345, 2022.

[32] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[33] M. Tsang, S. Rambhatla, and Y. Liu, "How does this interaction affect me? interpretable attribution for feature interactions," *Advances in neural information processing systems*, vol. 33, pp. 6147–6159, 2020.

[34] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[35] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 985–994.

[36] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 315–324.

[37] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *Sixth international conference on data mining (ICDM'06)*. IEEE, 2006, pp. 613–622.

[38] H. Wang and J. Leskovec, "Combining graph convolutional neural networks and label propagation," *ACM Transactions on Information Systems (TOIS)*, vol. 40, no. 4, pp. 1–27, 2021.

[39] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: a survey," *ACM Computing Surveys (CSUR)*, 2020.

[40] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The world wide web conference*, 2019, pp. 417–426.

[41] Z. Sun, Q. Ke, H. Rahmani, M. Bennamoun, G. Wang, and J. Liu, "Human action recognition from various data modalities: A review," *IEEE transactions on pattern analysis and machine intelligence*, 2022.

[42] X.-M. Zhang, L. Liang, L. Liu, and M.-J. Tang, "Graph neural networks and their current applications in bioinformatics," *Frontiers in genetics*, vol. 12, 2021.