PGX: A Multi-level GNN Explanation Framework Based on Separate Knowledge Distillation Processes

Tien-Cuong Bui  
cuongtb91@snu.ac.kr  
Dept. of ECE  
Seoul National University  
South Korea

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

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

ABSTRACT

Graph Neural Networks (GNNs) are widely adopted in advanced AI systems due to their capability of representation learning on graph data. Even though GNN explanation is crucial to increase user trust in the systems, it is challenging due to the complexity of GNN execution. Lately, many works have been proposed to address some of the issues in GNN explanation. However, they lack generalization capability or suffer from computational burden when the size of graphs is enormous. To address these challenges, we propose a multi-level GNN explanation framework based on an observation that GNN is a multimodal learning process of multiple components in graph data. The complexity of the original problem is relaxed by breaking into multiple sub-parts represented as a hierarchical structure. The top-level explanation aims at specifying the contribution of each component to the model execution and predictions, while fine-grained levels focus on feature attribution and graph structure attribution analysis based on knowledge distillation. Student models are trained in standalone modes and are responsible for capturing different teacher behaviors, later used for particular component interpretation. Besides, we also aim for personalized explanations as the framework can generate different results based on user preferences. Finally, extensive experiments demonstrate the effectiveness and fidelity of our proposed approach.

CCS CONCEPTS

- Computing methodologies → Machine learning; Artificial intelligence.

KEYWORDS

Explainable AI, Graph Neural Networks, Knowledge Distillation

1 INTRODUCTION

Graph Neural Networks (GNNs) [38] are essential components in numerous real-world applications nowadays, such as recommendation systems [34, 36, 40], social networks [7, 23], e-commerce websites [37, 45], and fraud detection [2, 5, 15]. As GNNs become more and more ubiquitous, we are interested in understanding their internal logic and influential factors in predictions. For instance, if a recommendation system suggests a user a movie, they want to know the reason for this suggestion. Besides, interpretation methods can also enhance trust and fairness in the system. It could be big trouble if suggestions are offensive due to gender discrimination or other social biases. Therefore, GNN explanation is mandatory to increase user trust in such systems.

GNNs suffer from the lack of transparency similar to other deep learning (DL) methods, thus making predictions unintelligible. Even worse, explaining GNN models is much harder than other DL models due to the complex message-passing patterns during execution. GNN predictions are impacted by node features, edge information, the graph structure, and observed node outcomes [27, 32]. While several GNN explanation methods have been proposed lately, they mainly focus on finding essential subgraphs via edge selection processes [19, 26, 41, 42] or Monte Carlo tree search [43]. However, we argue that they lack generalization and suffer from a computational burden, especially when graphs are massive. Besides, each user has different preferences and expectations for explanations matching their prior belief [20]. Therefore, explanations require multi-level and multi-granular outputs. Even though [35] proposes a multi-grained explanation method, it simply consists of two steps of edge selection and does not meet the requirements.

In this paper, we propose a multi-level GNN explanation framework to address the problems above, named PGX. From the data perspective, graph data consists of various data structures such as tabular data like node/edge features and the graph structure. Therefore, GNNs can be regarded as a multimodal learning process of multiple components in graph data. With that in mind, we break down the GNN explanation problem into simpler sub-problems categorized into a hierarchical structure presented in Fig. 3. Each level in our framework provides beneficial insights in a different way. Specifically, the highest level explanation measures the overall contribution of individual components by answering questions like “What are attributions of graph components to the model execution and predictions?” We adopt the non-additive interaction set concept proposed in [30] to answer this type of question. Furthermore, we relax the complexity of the problem by considering only node features and the graph structure in the interaction set, which
is equivalent to measuring each component’s marginal contribution. This relaxation also enables PGX to deliver explanations at fine-grained levels.

Next, we introduce a class of methods based on knowledge distillation [9] to measure particular feature attributes and the graph structure at fine-grained levels. Specifically, the second level of explanation is to obtain a general understanding of influential nodes in the graph and crucial attributes in node features, while the third level provides local explanations of predictions with specific neighboring impacts and feature attributions. Knowledge from the original GNN model is transferred to subsequent students, wherein each one is responsible for imitating a specific function of the teacher. For instance, a special graph-based student model tries to weight the interaction among nodes, while a non-graph-based neural network student is to capture the feature transformation process. Later, we implement the Personalized PageRank algorithm based on edge weights of the graph-based student for link analysis to find out globally important nodes and locally impactful neighbors. Furthermore, users can obtain interpretations of the graph structure via our interactive visualization. Next, the non-graph-based neural network is the input of feature attribution measurement methods [18, 25]. Each knowledge distillation process is a standalone procedure and executed only once, corresponding to each component. With the flexibility of student models, our approach is very promising and extensible. Furthermore, it also enables the explainer to provide customized explanations given different preferences towards personalization.

The main contributions of this paper are three-fold:

- We propose a multi-level GNN explanation framework, which provides a complete view of GNN explanations. The framework can deliver explanations with appropriate detailed information depending on specific requirements.
- We also propose leveraging knowledge distillation to transform original GNN models into simpler student models. By doing that, we can exploit existing explainable AI tools and link analysis algorithms for GNN explanations.
- We present extensive experiments with convincing evidence to verify the fidelity and consistency of proposed methods.

The remainder of this paper is structured as follows. In Section 2, we introduce the related work. In Section 3, we describe background concepts and problem formulation. Then we first describe the framework overview in Section 4. Section 5 elaborates on attribution analysis processes based on knowledge distillation. In Section 6, experiments are presented. Section 7 presents discussions with learned lessons and interesting observations. Finally, we conclude our work in Section 8.

## 2 RELATED WORK

### 2.1 Perturbation-based Methods

Following GNNExplainer [41], multiple perturbation methods [19, 26, 43] have been proposed to explain GNNs by extracting essential subgraphs. These methods focus on designing mask learning processes that act as an edge filter. However, as they all followed experimental settings of [41], which were executed on trivial synthetic datasets using a transductive style, it is difficult for these methods to operate on real-world problems when unknown nodes appear. Besides, the datasets used for the node classification were not general, as graphs in many domains assume that nodes in a community should have the same label [46]. Therefore, we argue that explanations of these methods are only appropriate for the graph classification problem as masks act as a global filter rather than paying attention to local impacts. Furthermore, the number of learned parameters linearly increases, corresponding to the number of edges, thus making it hard to scale.

### 2.2 Game-theoretic Approach

Shapley value is a concept in game theory that distributes gains and costs to players working in a coalition, which was first applied to machine learning model explanations by [18]. In GNN prediction interpretation, SubgraphX [43] is the first work that utilizes the Shapley value. Specifically, it explains predictions by exploring subgraphs with a Monte Carlo tree search guided by the marginal contribution values. GraphSVX [6] applied [18] to measure impact factors to predictions, wherein it assumed that nodes and features constitute a set of independent players. This assumption is unrealistic as the GNN aggregates neighbors’ information multiple times before making predictions. Besides, as the number of players is enormous, the contribution is scattered, leading to unhelpful explanations. Therefore, we must carefully specify the set of players in a cooperative game to focus on crucial things only. This specification is beneficial to users as they have different concerns when looking at a prediction.

### 2.3 Model-level Explanations

Unlike instance-level methods, model-level methods provide high-level insights and understandings of model behaviors. Applying techniques like [21] to explain GNN behaviors is troublesome as message-passing patterns are complicated. [42] was the first work that tackled this problem via graph generation. Specifically, it trains a graph generator via policy gradient so that the generated graph patterns maximize a certain prediction of the model. However, the generation process also focuses only on finding globally important subgraphs making it less sensitive to the local structure of a certain prediction.

Lately, Wang et al., 2021 [35] proposed a multi-grained approach for global and local explanations. First, it specifies the class-wise knowledge across instances with the same prediction via a class-aware attribution module, similar to [19, 41]. As the coarse-grained edge selection fails to detect dependencies within the selected edges, an additional fine-tuning step optimizes the edge sampling process to generate an appropriate explanation for a certain prediction. However, it does not consider feature importances and impacts of neighboring labels.

### 2.4 Knowledge Distillation for GNNs

Knowledge distillation [8] can be used to explain the overall logic behind the black box models [1] by making the student model transparent. Lately, it has been applied to GNNs, wherein student models try to reach the accuracy of teachers while requiring fewer computations. Xiang et al., 2021 [3] proposed distilling knowledge from pre-trained GNNs to a student model using generated fake graphs. Similarly, Chaitanya et al., 2021 [12] investigated representation...
distillation by introducing a specialized loss function. Lately, Yuan et al., 2022 [16] applied [9] to construct an explainable GNN, which decomposes the interpretation process into multiple edge selection processes via attention mechanisms. Even though it provides a high-level understanding of model predictions, it encounters two fatal problems: Computation costs due to multi-hop decomposition; Trustability of the attention networks, which do not imitate message-passing patterns.

| Symbol | Definition |
|--------|------------|
| $G$: $V$: $E$: $X$ | Graph; Node set; Edge set; Node features |
| $f$: $g$ | A trained GNN model; A surrogate model |
| $N_i$ | A set of 1-hop neighbors of node $i$ |
| $m_{ij}$ | Message on the edge between node $i$ and $j$ |
| $h_i^l$, $h_j^l$ | Node $i$'s embedding at layer $l$ and the last layer $L$ |
| $I$ | An interaction set in a multi-modal approach |
| $L_{CE}$: $L_{DL}$: $L$ | Cross-entropy loss; Distillation loss; Overall loss |
| $\lambda$ | A hyper-parameter to adjust $L_{DL}$ |
| $\omega$ | A function operates on elements in $I$ |
| $\mathcal{R}_V$; $\pi$ | The rank vector of nodes; Preference vector |
| $z_t$: $z_s$: $\tau$ | Teacher outputs; Student outputs; Temperature term |
| $y$: $\hat{y}$: $p$ | Label values; Predicted values; Softmax function |

Table 1: Glossary of Notations.

3 PRELIMINARIES

3.1 Graph Neural Networks

GNNs adopt the message-passing paradigm to propagate and aggregate information along edges in the graph. Specifically, each GNN layer involves three crucial steps: propagation, aggregation, and updating. First, a message $m_{ij}^l = \text{Message}(h_i^{l-1}, h_j^{l-1})$, where $h_i^{l-1}$ is the representation of a node $i$ at layer $l - 1$, is propagated to adjacent nodes. Second, for each node $v_i$, GNN aggregates messages received from its neighbors $N_i$, wherein the aggregation operation is $m_i^l = \text{Aggregation}(m_{ij}^l, j \in N_i)$. Lastly, given $m_i^l$ and $h_i^l$, GNN computes a new representation vector for $v_i$ at layer $l$ as follows: $h_i^{l+1} = \text{Update}(m_i^l, h_i^l)$. The representation vectors of the last layer $h_i^L$ are then used for downstream tasks, such as node classification.

3.2 Knowledge Distillation

Knowledge Distillation, first proposed in [9], is originally used for model compression, wherein a small model (student) tries to mimic the large model (teacher)’s decisions. Gou et al., 2021 [8] presented a complete survey of knowledge distillation, including various techniques and approaches. Here, let us briefly summarize the original method [9].

$$p(z_t; \tau) = \frac{\exp(z_t / \tau)}{\sum_{z} \exp(z / \tau)} \quad (1)$$

In Fig. 1, given the same set of input features, both models first output so-called soft labels/predictions, which are results of softmax function with a temperature term $\tau$ shown in Eq. (1). Then, the student model computes the distillation loss, usually a soft cross-entropy loss function. After that, it also generates hard predictions to compare with the ground-truth labels via the cross-entropy loss function. Finally, we have the student loss function as follows:

$$L = L_{CE}(y, p(z_t, 1)) + \lambda L_{DL}(p(z_t, \tau), p(z_s, \tau)), \quad (2)$$

where $z_t$, $z_s$, and $y$ are teacher/student outputs and labels, respectively. $\lambda$ is a hyper-parameter that controls the magnitude of $L_{DL}$.

3.3 GNN Explanation: Problem Formulation

Given a graph $G = (V, E, X)$, a GNN model $f$ is trained to predict node/edge/graph level outcomes in classification problems. Explanations for GNN predictions aim at finding crucial factors contributing to the most to the model execution and prediction scores. Specifically, we construct a hierarchical structure of these factors, as presented in Fig. 3. This paper limits the scope to only node classification problems and discards edge features in all analyses for simplicity. We can easily extend the proposed solutions to other GNN problems but leave them for future work.

4 PGX: A MULTI-LEVEL EXPLANATION FRAMEWORK

Many factors contribute to GNN execution and predictions as GNN models usually adopt message-passing paradigms. Node information is passed to neighbors in complicated patterns making the model hard to disentangle and explain. As opposed to existing approaches which only focus on specific aspects of explanations, we propose a unified framework to explain GNNs on multiple levels.

4.1 Framework Overview

Definition 1. A GNN model operates on a graph $G$ is a multi-modal learning process that simultaneously handles different entities and relationships.

A GNN model can be considered a multimodal approach if we look at it from the data point of view, as Fig. 2 shows. Specifically, graph data is the constitution of multiple entities (nodes), relationships (edges), and attributes. GNN models are efficient methods to model these factors simultaneously. Due to multimodal properties, providing a single explanation for GNNs is incomplete. Besides, from a user perspective, each person prefers different explanations.
for the same input graph and trained model. To solve these problems, we introduce a novel multi-level GNN explanation framework represented as a hierarchical structure in Fig. 3.

Specifically, the framework provides explanations on multiple levels, from a global view to fine-grained attributions. At the top level, it measures the global attribution of each component in the graph data, such as the structure or features, by adopting the concept proposed in [30]. The second level focuses on essential factors within each particular component. In other words, it answers questions like “Which are the top influential nodes?” or “Which are the top influential features?” Next, the framework explains individual predictions by outputting neighbors’ influences and local feature attributions in the last level. Besides, the last-level explanation is flexible as we can view local influences in multiple hops via an interactive visualization tool. To handle the last two levels, we use knowledge distillation to transfer knowledge from the base model to simpler ones. Particularly, each student has a role in imitating a specific function of the teacher. For instance, a special graph-based model captures the interactions among nodes, while a non-graph-based neural network model is trained to imitate the whole feature transformation process. After that, each one is responsible for the interpretation of a particular component. Please note that training one student does not affect the learning process of other students. Therefore, it can be done sequentially or simultaneously, depending on particular situations.

4.2 Component-level Attribution Measurement

As the graph structure and node features are the main ingredients of GNN execution, ones often assume that they are equally important [6, 10]. We argue that interactions in GNNs are non-additive attributions due to complex patterns of message-passing schemes. Following [11, 29, 30], we define the component-level attributions as a non-additive interaction of nodes, edges, and features.

**Definition 2.** A function $f$ operating on $G = (V, E, X)$ consists of a statistical non-additive interaction of multiple factors indexed in the set $I \iff f$ cannot be decomposed into a sum of $|I|$ subfunctions $f_i$, which does not include $i^{th}$ interaction, such that: $f(x) \neq \sum_{i \in I} I f_i(x_{\setminus \{i\}})$.

Let us assume that $f$ operating on $G$ can be represented as a generalized additive function [29, 30] as follows:

$$f(G) = \sum_{i \in I} \omega_i(x_{I_i}) + b, \quad (3)$$

where $x_{I_i}$ corresponds to the element $i^{th}$ in $I$, $\omega_i$ is a function of $x_{I_i}$, and $b$ is regarded as a bias. Following [30], we propose using reference values to calculate the attribution of each element in $I$. Let us denote the reference value of $x_{I_i}$ as $x'_{I_i}$. The attribution of $I_i$ is calculated as follows:

$$\phi(I_i) \equiv \omega_i(x_{I_i}) - \omega_i(x'_{I_i}). \quad (4)$$

**Proof.** Based on Eq. (3), we have:

$$\phi(I_i) = f(G) - f(G'_{\setminus \{i\}})$$

$$= \left(\omega_i(x_{I_i}) + \sum_{j \in I, j \neq i} \omega_j(x_{I_j}) + b\right) - \left(\omega_i(x'_{I_i}) + \sum_{j \in I, j \neq i} \omega_j(x_{I_j}) + b\right)$$

$$= \omega_i(x_{I_i}) - \omega_i(x'_{I_i}). \quad (5)$$

To relax the problem complexity, we limit the set of interactive factors to $I = (E, X)$. In other words, we consider the graph structure and the set of node features as two interactions in $I$. Therefore, we compute the marginal contribution (MC) for $I_i$, as follows:

$$\phi(X) = f(G) - f(G'_{\setminus \{X\}}),$$

$$\phi(E) = f(G) - f(G'_{\setminus \{E\}}). \quad (6)$$

Here, $f$ outputs a predicted probability score for a given graph. In practice, reference values for node features are either fixed-value
vectors or average-value vectors. However, generating a reference graph is difficult due to the network structure complexity. Inspired by perturbation techniques applied to images [18, 25], we create a graph that contains only self-loop edges for the structure reference.

5 FINE-GRAINED ATTRIBUTION ANALYSIS

Fine-grained attribution analysis corresponds to the bottom-two levels in Fig. 3, which is responsible for evaluating specific contributions of elements in the graph structure and node features. In Section 4.2, we assume that interactions only occur within an individual component, such as edges in the graph structure and attributes in node features. Based on this assumption, we evaluate fine-grained attribution for components separately using two different knowledge distillation procedures.

DEFINITION 3. Given a GNN model \( f \), fine-grained attributions can be obtained via separate knowledge distillation processes.

5.1 Node Feature Attribution

Feature attribution analysis is an active area in interpretable machine learning with numerous methods [18, 25, 30]. However, existing methods are inappropriate for GNNs due to complex message-passing patterns. We solve this problem indirectly using the knowledge distillation method, presented in Fig. 1. Specifically, a non-graph-based model is trained to capture the feature transformation process of the original GNN. As depicted in Fig. 4, any neural network that takes a set of samples as inputs can be a candidate for the student model. Practically, the student model is exposed to more labeled data, thus making it capable of imitating the GNN teacher behaviors. After that, the student model is plugged into feature attribution methods to obtain a global summary of feature importances or a local explanation of an instance. As feature attribution methods have been continuously proposed, our approach enables one to integrate these methods into GNNs more easily.

Feature Attributions

![Diagram](image-url)

**Figure 4:** Execution Pipeline for Measuring Feature Attributions based on Knowledge Distillation. Pre-trained GNN models are distilled to non-graph-based neural networks used for feature attribution analysis.

5.2 Graph Structure Attribution

Unlike existing methods [19, 41] that extract essential subgraphs via mask learning processes, we approach the graph structure attribution analysis as a link analysis problem, which is well-studied in graph research. As presented in Algorithm 1, the attribution analysis procedure consists of only four steps. First, knowledge from model \( f \) is distilled to a graph-based surrogate model \( g \) via the paradigm presented in Fig. 1. After training \( g \), we extract a learned matrix representing interactions among nodes. As this matrix is row-wise normalized, we implement a customized version of the PageRank [22] algorithm with an initialized preference vector \( \pi \) to measure node importances. After that, we visualize the output ranks \( \mathcal{R}_V \) and highlight the top-\( k \) nodes, where \( k \) is an arbitrary parameter. If \( \pi \) is uniformly distributed to all nodes in the graph, the top-\( k \) nodes can be considered anchor points or influential seeds of the graph. Besides, we can personalize an instance explanation by manipulating the preference vector \( \pi \), similar to the Personalized PageRank algorithm.

**Algorithm 1** Graph Structure Attribution Analysis Procedure

**Input:** \( f, \mathcal{G}, \pi \)

**Output:** \( g, \mathcal{R}_V \)

1. \( g = \text{distill}(f) \) [Distill knowledge from \( f \) to \( g \)]
2. \( \mathcal{A}^* = \text{extractAdjacency}(g) \) [Extract learned interaction matrix]
3. \( \mathcal{R}_V = \text{PageRank}(\mathcal{A}^{*T}, \pi) \) [Execute PageRank on \( \mathcal{A}^* \) and \( \pi \)]
4. visualize \( (g, \mathcal{R}_V, \mathcal{A}^*) \)

To make Algorithm 1 work, we have to carefully design the student model, in which \( \mathcal{A}^* \) requires two conditions: interaction representation and shared across layers. We propose two possible candidates that satisfy these conditions: GCN-LPA [32] and SGAT, a customized version of GAT [31]. First, Wang et al., 2021 [32] introduced a GCN-based model that combines GCN [13] with the label propagation algorithm (LPA). This model trains the adjacency matrix in the original GCN formula to capture feature and label influences among nodes. Second, we introduce SGAT, a modified version of GAT [31], wherein only the first layer’s attention with 1-head is computed and shared across layers, inspired by [39]. One may think that why don’t we measure node interaction using attention heads in GAT directly. For instance, Ying et al., 2019 [41] averaged all attention heads across layers. However, we argue that this process can change the edge importance distribution.

6 EXPERIMENTS

6.1 Experimental Setups

| Dataset   | #Nodes | #Edges | #Node Features | #Classes |
|-----------|--------|--------|----------------|----------|
| Cora      | 2,708  | 10,556 | 1,433          | 7        |
| Citeseer  | 3,312  | 9,228  | 3,703          | 6        |
| Pubmed    | 19,717 | 44,338 | 500            | 3        |
| YelpChi(R-U-R) | 45,954 | 98,630 | 32             | 2        |
| Amazon(U-P-U) | 11,944 | 351,216 | 25            | 2        |

**Table 2:** Dataset Information.

Datasets. We selected two types of datasets: citation graphs and fake review detection [33]. We observed that node features in citation graphs were constructed from the bag-of-words model representing the most common words in documents. Therefore, finding important features for individual predictions or visualizing a global view of feature attributions is less intuitive. Instead, we focused on analyzing graph structures to provide appropriate explanations. Inversely, fake review graphs contained handcrafted features and
were constructed based on mutual interactions, such as reviews from the same user having connections to each other (R-U-R) or users commenting on the same product link to each other (U-P-U). Therefore, we use these datasets to evaluate the feature attribution procedure. Detailed information on all datasets is listed in Table 2. Evaluation Metrics. We used seven metrics to evaluate proposed methods in different scenarios, including those listed in Eq. (6), Eq. (7), ROC-AUC (AUC), and Recall (R). Besides, accuracy (ACC), agreement (ARG), and $\Delta_{\text{acc}}$ were scaled to percentage. Experiments used different metrics depending on particular configurations. Experiments used different metrics depending on particular configurations.

\[
\text{Accuracy} = \frac{1}{N} \sum_{j=1}^{N} \mathbb{1}(\hat{y}^j = y^j)
\]

\[
\Delta_{\text{acc}}(J_i) = \frac{1}{N} \sum_{j=1}^{N} (\mathbb{1}(\hat{y}^j = y^j) - \mathbb{1}(\hat{y}^j \setminus \{i\} = y^j))
\]

\[
\text{Agreement} = \frac{1}{N} \sum_{j=1}^{N} \mathbb{1}(\hat{y}^j = \hat{y}^j_i)
\]

\[
\text{Predictive KL} = \frac{1}{N} \sum_{j=1}^{N} KL(\hat{p}_i^j || \hat{p}_j^i)
\]

Overall Configurations. We implemented GNN models using DGL [33] v0.8 and followed its example code for parameter settings. All experiments were run on an NVIDIA DGX A100 server with 4 Tesla V100 GPUs. GNN models were trained for 200 epochs most of the time without stopping using the Adam optimizer with a learning rate of 0.01 and weight decay of 5e-4. We report exceptional configurations in particular experiments.

### 6.2 Component-level Attribution Analysis

**Settings.** We selected APPNP, GraphSage, and GAT for teacher models and selected GCN-LPA, SGAT, and MLP as students. GraphSage and GAT followed the overall configurations, while APPNP was similar to the previous experiment. We followed [32] to select parameter settings for GCN-LPA while configuring SGAT as similar to GAT and MLP as the same as APPNP’s MLP. [28] shows that knowledge distillation sometimes causes a large divergence between the predictive distribution of the student and the teacher, wherein a good student does not imply high fidelity. In such cases, student models cannot be used as surrogate models for teacher interpretation. Therefore, we aim to answer two questions:

1. Are student models as accurate as their teachers on graph data?
2. Are student models trustworthy to use for explanations?

We evaluated models based on three metrics accuracy, agreement, and KL. According to [28], agreement and predictive KL scores correspond to student fidelity. Besides, the fidelity is inversely proportional to student generalization, which means if students outperform teachers, they are less trustworthy for explanations. In other words, if students can do some things that teachers cannot, explanations for students are not equivalent to those of teachers. A good student model must be as accurate as its teacher (high accuracy) and perfectly match the predictive distribution resulting in high agreement and low KL divergences. Table 4 presents model results. As can be seen, all student models outperform their baselines and are comparable to teachers in the accuracy metric. When comparing them in agreement and KL metrics, students of APPNP and GraphSage have the best results. Surprisingly, GAT’s students are inferior compared to these other ones. It’s hard to tell what causes these results, especially with the complexity of graph data. [28] naively suggests that increasing the student training set allows it to obtain more knowledge from the teacher. We guessed that the GAT teacher generates noise in its predictions, thus making it hard for students to mimic its behaviors correctly. Besides, out of two graph-based students, GCN-LPA is superior to SGAT as the label propagation process takes advantage of label similarity among nodes in citation graphs. However, SGAT is more applicable to scenarios requiring inductive training and testing as the model computes edge weights directly from node embeddings.

### 6.3 Fidelity of Surrogate Models

**Settings.** We chose APPNP, GraphSage, and GAT for teacher models and selected GCN-LPA, SGAT, and MLP as students. GraphSage and GAT followed the overall configurations, while APPNP was similar to the previous experiment. We followed [32] to select parameter settings for GCN-LPA while configuring SGAT as similar to GAT and MLP as the same as APPNP’s MLP. [28] shows that knowledge distillation sometimes causes a large divergence between the predictive distribution of the student and the teacher, wherein a good student does not imply high fidelity. In such cases, student models cannot be used as surrogate models for teacher interpretation. Therefore, we aim to answer two questions:

1. Are student models as accurate as their teachers on graph data?
2. Are student models trustworthy to use for explanations?

We evaluated models based on three metrics accuracy, agreement, and KL. According to [28], agreement and predictive KL scores correspond to student fidelity. Besides, the fidelity is inversely proportional to student generalization, which means if students outperform teachers, they are less trustworthy for explanations. In other words, if students can do some things that teachers cannot, explanations for students are not equivalent to those of teachers. A good student model must be as accurate as its teacher (high accuracy) and perfectly match the predictive distribution resulting in high agreement and low KL divergences. Table 4 presents model results. As can be seen, all student models outperform their baselines and are comparable to teachers in the accuracy metric. When comparing them in agreement and KL metrics, students of APPNP and GraphSage have the best results. Surprisingly, GAT’s students are inferior compared to these other ones. It’s hard to tell what causes these results, especially with the complexity of graph data. [28] naively suggests that increasing the student training set allows it to obtain more knowledge from the teacher. We guessed that the GAT teacher generates noise in its predictions, thus making it hard for students to mimic its behaviors correctly. Besides, out of two graph-based students, GCN-LPA is superior to SGAT as the label propagation process takes advantage of label similarity among nodes in citation graphs. However, SGAT is more applicable to scenarios requiring inductive training and testing as the model computes edge weights directly from node embeddings.

| Baseline | Feature | Structure | $\Delta_{\text{acc}}$ | MC | $\Delta_{\text{acc}}$ | MC |
|----------|---------|-----------|----------------------|----|----------------------|----|
| Cora     | 83.1    | 68.7      | 0.48                 | 11.6| 0.20                 |
| Citeseer | 71.9    | 64.2      | 0.48                 | 6.2 | 0.13                 |
| Pubmed   | 79.3    | 61.3      | 0.54                 | 6.1 | 0.22                 |

Table 3: The Contribution of Components in Prediction Accuracy on Citation Graphs.
### 6.4 Measuring Node Feature Attributions

**Settings.** We chose two fake review detection datasets, taken from [5, 24, 44], and described their top-10 features in Table 5. Please refer to [24, 44] for a more detailed description. In training, we applied the class-weighted technique to the GCN teacher and the oversampling method to students as datasets were highly imbalanced. Besides, we trained MLP students for 1000 epochs. For evaluation, we used three metrics: ROC-AUC (AUC), Recall (R), and agreement.

#### Dataset |

| Column | Description |
|--------|-------------|
| MNUV   | Min. of unhelpful votes |
| DGP    | #days between user first and last rating |
| LFS    | Length of feedback summary |
| MDR    | Median of rates given by the user |
| MXR    | Max. of rates given by the user |
| MNR    | Min. of rates given by the user |
| #SS    | #5-star given by the user |
| LU     | Length of username |
| #1s    | #1-star given by the user |
| STM    | Text sentiment of reviews |

| Dataset | Column | Description |
|---------|--------|-------------|
| ISR     | If review is user’s sole review |
| L       | Review length in words |
| Rank    | Rank order among all the reviews of product |
| UAvgRD  | User’s Avg. rating deviation |
| PMNR    | Product’s Max. #reviews written in a day |
| UMNR    | User’s Max. #reviews written in a day |
| EXT     | Extremity of rating, high vote vs low vote |
| UBST    | Burstiness, spammers are short-term members |
| UWHRD   | Weighted rating deviation by recency of user |
| PETG    | Entropy of temporal gaps of a product’s reviews |

#### Table 5: Top-10 features in Yelp and Amazon datasets. Please refer to [5, 24, 44] for more information.

Here, we aim to verify whether feature attributions can be accurately measured by the procedure presented in Fig. 4. After distilling knowledge from the GCN teacher to a simple MLP model, we executed SHAP on the student model to obtain a global understanding of feature attributions in two datasets, as presented in Fig. 5. In the Yelp dataset, ISR (user’s sole review), L (review length), and Rank (among a product’s reviews) are top-3 contributed attributes. Similarly, for the Amazon dataset, the minimum number of unhelpful votes of a user and the day gap are the two most influential factors. Without domain knowledge and a clear understanding of the problem, it’s hard to judge whether attribution plots make sense. Therefore, we re-train the student model with only top-5 features for both datasets. The results, listed in Table 6, confirm the correctness of the attribution process as the top-5 re-trained student is almost as accurate as the original student model.

#### Table 6: Results of the GCN Teacher and MLP Students on Yelp and Amazon Datasets. The MLP student trained with top-5 features is comparable to one trained with all features.

| Dataset | Column | Description |
|---------|--------|-------------|
| ISR     | If review is user’s sole review |
| L       | Review length in words |
| Rank    | Rank order among all the reviews of product |
| UAvgRD  | User’s Avg. rating deviation |
| PMNR    | Product’s Max. #reviews written in a day |
| UMNR    | User’s Max. #reviews written in a day |
| EXT     | Extremity of rating, high vote vs low vote |
| UBST    | Burstiness, spammers are short-term members |
| UWHRD   | Weighted rating deviation by recency of user |
| PETG    | Entropy of temporal gaps of a product’s reviews |

### 6.5 Examining Graph Structure Attributions

To increase trust in the explainer, we implemented a visualization tool that supports interactive explanations of the graph structure. The software provides a global view of graph embedding and multi-level visualization for local explanations. Each view supports edge filtering and highlighting features for nodes and edges to provide a clear understanding of internal model logic. This section presents Algorithm 1’s results on the Cora graph with APPNP teacher and SGAT student using this software.
Figure 5: Feature Attributions in the Fake Review Detection Task. We only display top-10 important features attributing to model predictions for two corresponding datasets.

As presented in Fig. 6, nodes in the Cora graph are displayed in clusters corresponding to its embedding vectors, projected into the 2-D dimension using T-SNE. From this global view, we can detect cluster boundaries and specify influential factors on a high level. By selecting a node, we can see how it impacts on or is affected by its neighbors. Furthermore, the global view provides an overall understanding of how neighbors impact nodes located at the boundary lines in different clusters. We also observed that important nodes usually have a high degree connected to numerous nodes within their clusters and are located at extreme corners. However, an influential node in the global view only means that it connects to numerous neighbors and sometimes does not have a decisive impact on them. For instance, a top-cited paper has many links to papers across domains, but downstream nodes’ classes depend on their locally connected neighbors.

Similarly, Fig. 7 depicts multiple levels of local explanations for an example node. In local explanation, our software provides additional features such as graph layout selection, k-hop highlighting, edge filtering, feature similarity, and label similarity [17] to support various requirements. First, we can adjust the number of visible
neighbors in one view ordered by the Personalized PageRank algorithm rooted as the target node. With this feature, we present influential flows from neighbors to the target node at different levels. For simplicity, we only show two levels in Fig. 7. Secondly, feature and label similarities, shown in the top left corner of this figure, present the purity of the community rooted at a target node, which provides another perspective of how the prediction was made. Finally, multi-layout graph visualization is crucial as users may prefer different views of the same output, as shown in Fig. 8.

![Figure 8: Different Graph Layouts for a Local Explanation of Node 5 (Red Node) in Cora Graph.](image)

6.6 Examining Online Knowledge Distillation

**Settings.** We adopted the training paradigm from [14]. For simplicity, we selected only the best teacher APPNP with two students, GCN-LPA and MLP. All models were trained on citation graphs using transductive settings.

|          | Teacher |          | Student |          | Student |
|----------|---------|----------|---------|----------|---------|
|          | APPNP   | GCN-LPA  | MLP     | ACC      | ACC     | ARG     | KL      | ACC      | ARG     | KL      |
| Cora     | 83.5    | 81.7     | 88.8    | 0.159    | 75.0    | 76.1    | 1.568   |
| Citeseer | 72.4    | 69.1     | 83.5    | 0.113    | 69.9    | 73.8    | 0.471   |
| Pubmed   | 78.9    | 76.4     | 91.5    | 0.086    | 76.2    | 84.1    | 0.386   |

Table 7: Performance of Models in the Online Knowledge Distillation Task. Here, agreement scores are lower than the scores obtained by the offline mode.

As shown in Table 7, all models obtain low agreement scores compared with their teachers. This observation suggested that even though online knowledge distillation can help student models achieve teacher accuracy, its fidelity level is significantly lower than the offline mode. Therefore, this trade-off should be carefully considered in real-world scenarios to balance system performance and accuracy benefits.

7 DISCUSSION

Recently, explainable AI (XAI) has been an active research area as explanation is now a crucial part of advanced AI systems (software 2.0 [4]). We argue that XAI for GNN is only at the early stage, and there is plenty of room for improvement. When starting this work, we observed that most current methods linked to [41] proposed to train an explainer using masking methods on synthesized datasets with handcrafted ground-truth information. However, explanation ground truths are difficult to obtain in real-world scenarios. Furthermore, explanations are selected and don’t refer to statistical relationships due to biases [20]. Hence one explanation does not fit all people. Besides, an interactive visualization tool, which provides multiple meaningful information, is needed to enhance consensus between users and the explanation system.

Many feature attribution methods have been proposed lately, such as [18, 30]. We thought about how to use them for GNN explanations. However, it is not easy to apply these methods to GNN models directly due to computations in the graph structure. We found that knowledge distillation works perfectly with GNNs, wherein we have various options for student models. Therefore, the idea is to transform the original GNN models into interpretable versions. A similar approach is applied to the graph structure attribution analysis with numerous efficient methods such as PageRank [22].

Our work has a few limitations. First, we haven’t elaborated on fine-grained interactions among features and the graph structure. Even though we found that node features are crucial to the prediction performance, how the model performance got enhanced when including the graph structure was not analyzed. Besides, we only tackled the node classification problem while untouched the other problems such as graph classification or link prediction. These challenges will be potential research topics.

8 CONCLUSION

In this paper, we proposed a multi-level GNN explanation framework. We observed that graph data is a composition of entities regarded as nodes and relationships between them, which form the graph structure. Therefore, we considered GNN as a multimodal learning process of these components. Next, we decomposed the explanation problem into multiple sub-problems, formulating a hierarchical structure. At the top level, the framework specified the contribution of each component to GNN execution and predictions. In contrast, fine-grained levels focused on feature attribution and graph structure attribution analysis based on knowledge distillation. Furthermore, we also aimed to personalize explanations by providing customized outputs for different preferences. To confirm our hypotheses, we conducted extensive experiments and showed that the proposed approach is promising and extendable.
[42] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. 2020. XGNN: Towards Model-Level Explanations of Graph Neural Networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (Virtual Event, CA, USA) (KDD ’20). Association for Computing Machinery, New York, NY, USA, 430–438. https://doi.org/10.1145/3394486.3403985

[43] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. 2021. On Explainability of Graph Neural Networks via Subgraph Explorations. In Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139), Marina Meila and Tong Zhang (Eds.). PMLR, Virtual Event, 12241–12252.

[44] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Nguyen Hung, Zi Huang, and Lichen Cui. 2020. 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. Association for Computing Machinery, New York, NY, USA, 689–698.

[45] Huachi Zhou, Quaoyu Tan, Xiao Huang, Kaisong Zhou, and Xiaoling Wang. 2021. Temporal Augmented Graph Neural Networks for Session-Based Recommendations. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. Association for Computing Machinery, New York, NY, USA, 1798–1802.

[46] Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation.