ENGAGE: Explanation Guided Data Augmentation for Graph Representation Learning

Yucheng Shi[1], Kaixiong Zhou[2], and Ninghao Liu[1]✉

[1] University of Georgia, Athens, GA 30602, USA
{yucheng.shi, ninghao.liu}@uga.edu
[2] Rice University, Houston, TX 77005, USA
kaixiong.zhou@rice.edu

Abstract. The recent contrastive learning methods, due to their effectiveness in representation learning, have been widely applied to modeling graph data. Random perturbation is widely used to build contrastive views for graph data, which however, could accidentally break graph structures and lead to suboptimal performance. In addition, graph data is usually highly abstract, so it is hard to extract intuitive meanings and design more informed augmentation schemes. Effective representations should preserve key characteristics in data and abandon superfluous information. In this paper, we propose ENGAGE (ExplaNation Guided data AuGmEntation), where explanation guides the contrastive augmentation process to preserve the key parts in graphs and explore removing superfluous information. Specifically, we design an efficient unsupervised explanation method called smoothed activation map as the indicator of node importance in representation learning. Then, we design two data augmentation schemes on graphs for perturbing structural and feature information, respectively. We also provide justification for the proposed method in the framework of information theories. Experiments of both graph-level and node-level tasks, on various model architectures and on different real-world graphs, are conducted to demonstrate the effectiveness and flexibility of ENGAGE. The code of ENGAGE can be found here [1].

Keywords: Graph learning · Contrastive learning · Explainability.

1 Introduction

Graph representation learning has been shown to be powerful in many graph analysis tasks [10,57,51,52,30]. In particular, unsupervised graph representation learning methods [16,23,21] have attracted substantial attention, as they are adaptive to various applications especially when labels are scarce. Among them, recently contrastive learning [35,57] has achieved superior performance by learning representations from contrastive views to reduce superfluous information. However, most existing works [2,3,57,52] simply apply random sampling for data augmentation, which could accidentally break the graph structures, thus reducing

[1] https://github.com/sycny/ENGAGE.

the effectiveness of representations. In other domains such as computer vision and natural language processing, more intelligent data augmentation methods have been proposed by leveraging the semantic information [22,35] or domain knowledge [4]. However, it is obscure to define and extract meaningful elements from abstract data types such as graphs. Also, the non-Euclidean property of graph increases the augmentation difficulty.

Some preliminary efforts have been made to tackle the problem. For example, Fang et al. [5] propose using knowledge graphs to augment the original input graph and help construct contrastive pairs, which is applicable to computational chemistry. However, a more common and intriguing problem is how to design intelligent augmentation without external information sources. In addition, Zhu et al. [58] apply centrality as the node importance indicator and encourage perturbing nodes of lower centrality. However, node centrality is a static metric and is not necessarily relevant to downstream tasks in all scenarios. Some work [11,51] design end-to-end frameworks to automatically formulate data augmentation policies, but they could induce high computational costs. Also, their black-box nature would fail to explicitly locate task-relevant information in graph data.

To solve the problem, this work proposes ENGAGE (ExplaNation Guided data AuGmEntation), where we use explanation to guide the generation of contrastive views for learning effective unsupervised representations on graph data. Specifically, we first propose a new explanation method called Smoothed Activation Map (SAM). Different from existing explanation methods that focus on understanding model predictions, SAM measures node importance based on the distribution of representations in the latent space. Then, with the node importance scores, we design an explanation guided graph augmentation method by perturbing edges and node features, which is used to construct paired graph views for contrastive learning. By leveraging explanations, significant graph structures are less likely to be damaged by accidents. Meanwhile, our SAM method is efficient by using the quantization technique [14], so that it can be applied into the training process. Finally, we conduct experiments on multiple datasets and tasks to demonstrate the effectiveness and flexibility of our proposed method. The contributions can be summarized as below:

- We propose the ENGAGE framework which leverages explanation to inform graph augmentation, and uses contrastive learning for training representations to preserve the key parts in graphs while removing uninformative artifacts.
- We propose a new efficient explanation method called SAM for unsupervised representation learning on graph data.
- We conduct comprehensive experiments on both node-level and graph-level classification tasks, with two representative contrastive learning models and different encoders, demonstrating the effectiveness and flexibility of ENGAGE.

2 Related work

2.1 Representation Learning for Graph Data

Learning effective node or graph representations benefits various graph mining tasks. There are several major types of representation learning methods for graph data, including random-walk based methods [8,23,25,33], graph neural networks (GNNs) [7,9,37,38,43,44], and graph contrastive learning [2,57,46,32]. As the pioneering methodology of representation learning on graphs, random-walk based methods optimize node representations so that nodes within similar contexts tend to have similar representations. Then, graph neural networks become the new state-of-the-art architecture to process graph data. Finally, graph contrastive learning could be used to further refine the representation learning process with GNNs as the encoder.

2.2 Graph Contrastive Learning

Contrastive learning trains representations by maximizing the mutual information between different augmented views [2,12,3,34,38]. For example, GRACE [57] constructs the negative node examples by dropping edges and masking features, and employs a contrastive model similar to SimCLR [2]. GraphCL [52] perturbs graphs with four kinds of data augmentations on the graph level, and employs NT-Xent loss [2] for training. SimGRACE [46] gets rid of input data augmentation, and chooses to perturb model parameters. Data augmentation is vital for contrastive learning to achieve good performance on downstream tasks [22,35]. However, existing methods mainly rely on random perturbation, which could hurt graph information in the augmented views. To tackle the issue, Zhu et al. [58] propose to use node centrality as an importance indicator and perturb nodes with lower node centrality. Xu et al. [47] propose an information-aware representation learning model [36] to keep task-relevant information both in local and global views. Li et al. [17] propose a graph rationale guided data augmentation to better preserve semantic information. You et al. [51] design an end-to-end framework to automatically optimize data augmentation. However, it remains a challenge how to design data augmentation that is adaptive to different graph/node samples while keeping relatively low computing costs.

2.3 Explanation for Graph Neural Networks

Existing approaches for explaining GNN models can be divided into several categories [54], such as substitute-based, relevance-based, perturbation-based, generation-based, and rationale-based methods. Substitute-based methods approximate the original model with simplified but explainable substitutes, which either leverage GNN mechanisms [1,24,42] or bypass the model information [13,39,55]. Relevance-based methods compute input contributions by redistributing activations between neurons from the output layer to the input layer [24,27,28]. Perturbation-based methods [19,20,26,41,50] estimate input scores with the assumption that removing important features will have a large impact on the

prediction. Generation-based methods produce synthetic graphs that maximally activate the target neuron [53]. Rationale-based methods find a small subgraph which best guides the model prediction as explanation [45,17,18]. Existing methods mainly focus on supervised graph learning models, while we attempt to obtain explanation for unsupervised graph learning. Meanwhile, we propose to leverage explanation for improving models, which further requires the explanation algorithm to be efficient.

3 Preliminaries

3.1 Notations

Let $G = \{\mathcal{V}, \mathcal{E}, \boldsymbol{A}, \boldsymbol{X}\}$ be a graph, where $\mathcal{V}$ and $\mathcal{E}$ denote the set of nodes and edges, respectively. $\boldsymbol{A} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the adjacency matrix. If node v_i and node v_j are connected, then $A_{ij} = 1$, where we use $e_{i,j}$ to denote that edge. $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}|\times D}$ is the feature matrix and $\boldsymbol{x}_i = \boldsymbol{X}_{i,:}$ is the feature vector of node v_i. The GNN encoder is denoted as $f(\cdot)$, which transforms the nodes to representations $\boldsymbol{Z} = f(\boldsymbol{X}, \boldsymbol{A}), \boldsymbol{Z} \in \mathbb{R}^{|\mathcal{V}|\times K}$, where K is the latent dimension and $\boldsymbol{z}_i = \boldsymbol{Z}_{i,:}$ denotes the representation of node v_i. Meanwhile, we also consider graph-level tasks in this work. Let $\mathcal{G} = \{G_1, G_2, ..., G_N\}$ denote a set of graphs. In this case, the representation of a graph G_n is obtained as $\boldsymbol{z}_n = f(G_n)$, where $\boldsymbol{z}_n \in \mathbb{R}^K$. Our proposed method is applicable to both types of tasks. Unless otherwise stated, the methodology part in this paper assumes using graph-level tasks for illustration.

3.2 Contrastive Learning Frameworks

In this work, we adopt contrastive learning for learning node/graph representations. We consider two types of contrastive learning frameworks. The first type includes GraphCL [52] and GRACE [57] that are motivated by SimCLR [2] and learn to maximize the consistency between positive views compared with negative views. The second type is based on Simsiam [3] and could get rid of negative samples by using the stop gradient to prevent model collapse. We modify SimCLR and Simsiam frameworks to adapt to graph representation learning, and apply ENGAGE on both frameworks.

Learning Objectives. In contrastive learning, each instance is augmented into two *positive views* whose representations are $\boldsymbol{z}^1$ and $\boldsymbol{z}^2$, respectively. In SimCLR [2], the loss for learning the representation of i-th instance is:

$$\ell_i = -\log \frac{\exp\left(\text{sim}\left(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2\right)/\tau\right)}{\sum_j \mathbb{1}_{[j\neq i]} \exp\left(\text{sim}\left(\boldsymbol{z}_i^+, \boldsymbol{z}_j\right)/\tau\right)}, \tag{1}$$

where $\boldsymbol{z}_i^+$ refers to $\boldsymbol{z}_i^1$ or $\boldsymbol{z}_i^2$, and $\boldsymbol{z}_j$ denotes the embeddings of other instances as negative views. Positive views are expected to preserve the core information

of the original instance, while negative views are expected to be irrelevant to the original instance. $\text{sim}\left(\boldsymbol{z}^1, \boldsymbol{z}^2\right) = {\boldsymbol{z}^1}^T \boldsymbol{z}^2 / \left\|\boldsymbol{z}^1\right\| \left\|\boldsymbol{z}^2\right\|$. The Simsiam [3] model gets rid of the negative views, and its loss function is defined as below:

$$\ell_i = -(\boldsymbol{z}_i^1 / \left\|\boldsymbol{z}_i^1\right\|_2) \cdot (\boldsymbol{z}_i^2 / \left\|\boldsymbol{z}_i^2\right\|_2). \tag{2}$$

Encoder and MLP Heads. We use graph neural networks as the encoder $f(\cdot)$ to learn graph/node representations $\boldsymbol{z}$. To comprehensively test our proposed framework, we apply several architectures, including graph convolutional networks (GCN) [15] and graph attention networks (GAT) [37] for node-level tasks. Also, we employ graph isomorphism networks (GIN) [48] for graph-level tasks. Then, the representations are fed into different MLP heads depending on the CL framework. In SimCLR, there is only one MLP head called $p_o(\cdot)$. In Simsiam, there are two MLP heads, namely $p_o(\cdot)$ and $p_e(\cdot)$.

4 The ENGAGE Framework

4.1 Mitigating Superfluous Information in Representations

We begin by introducing the guideline of minimizing superfluous information, which we will follow throughout this work, for learning effective and robust representations. Without loss of generality, we denote the original input and downstream task label as X and $\boldsymbol{y}$, respectively.

Definition 1. *(Sufficiency) A representation $\boldsymbol{z}$ is defined as sufficient for label $\boldsymbol{y}$ iff $I(X; \boldsymbol{y}) = I(\boldsymbol{z}; \boldsymbol{y})$.*

An illustration of sufficient representation is shown in Figure 3(a). To achieve good downstream task performance, it is encouraged to learn $\boldsymbol{z}$ from X without losing task-relevant information. Meanwhile, according to the information bottleneck principle [36], by reducing superfluous information from X, the sufficient representation $\boldsymbol{z}$ becomes more robust and gives better performance. This can be better understood by splitting the mutual information between X and $\boldsymbol{z}$ into two parts as below:

$$I(X; \boldsymbol{z}) = \underbrace{I(X; \boldsymbol{z} \mid \boldsymbol{y})}_{\text{superfluous information}} + \underbrace{I(\boldsymbol{z}; \boldsymbol{y})}_{\text{task-relevant information}}. \tag{3}$$

The first term is the information in $\boldsymbol{z}$ that is not useful for predicting $\boldsymbol{y}$, which should be minimized. The second term denotes the predictive information, which is not affected by the representation as long as $\boldsymbol{z}$ is sufficient for $\boldsymbol{y}$.

Recent research shows contrastive learning (CL) can control the amount of information preserved in representations [35,40]. However, without label information, it is difficult to specify which part of input information to be preserved in CL. In this work, we utilize explanation to identify non-trivial structural information in graphs. Then, the information recommended by explanation is given higher priority to be preserved in representations, while other information is more likely to be discarded. This is done by using explanation, instead of random perturbation, to guide data augmentation in contrastive learning.

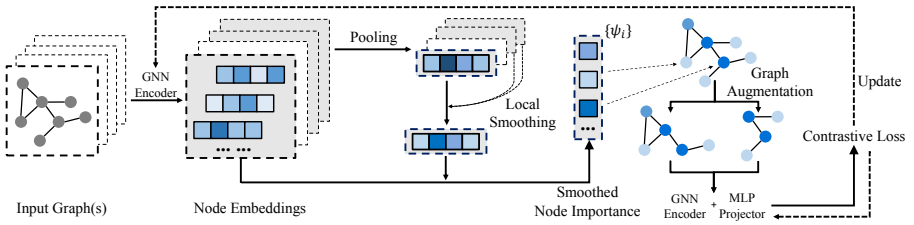


Fig. 1: Graph augmentation guided by Smoothed Activation Map (SAM) for contrastive learning.

4.2 Efficient Explanations for Unsupervised Representations

Class Activation Map. The goal of explanation is to identify the key components in input. We use GCN to illustrate the idea of explaining graph neural networks. A graph convolutional layer is defined as $F^l = \sigma(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}F^{(l-1)}\boldsymbol{W}^l)$, where F^l is output of l-th layer, $F^0 = \boldsymbol{X}$, $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_N$ is the adjacency matrix with self-connections, and $\tilde{\boldsymbol{D}}$ is the degree matrix of $\tilde{\boldsymbol{A}}$. $\boldsymbol{W}^l$ denotes trainable weights, and $\sigma(\cdot)$ is the activation function. In supervised learning, the Class Activation Map (CAM) [24,56] can be used to compute the importance of node v_i as: $\psi_i^c = \text{ReLU}(\sum_k w_k^c F_{k,i}^L)$, where L is the final convolutional layer, and $F_{k,i}^l$ is the k-th latent dimension (i.e., channel) of node v_i at the l-th layer. w_k^c is the weight of channel k at the output layer towards predicting class c.

However, there are two challenges that impede us from directly applying CAM to our problem. First, CAM is designed for specific model architectures with a GCN and a fully-connected output layer (consisting of the w_k^c weights), where its applicability is limited. Second, CAM works for supervised models, while we focus on unsupervised learning.

Smoothed Activation Map. To tackle the challenges, we propose Smoothed Activation Map (SAM) to explain representations without class labels. The key idea is to leverage the distribution of graph/node representations to identify the locally important and reliable graph components, as shown in Figure 2. Unlike CAM, our method works for different types of GNN models and does not require supervised signals. Without loss of generality, we write the feedforward process of GNNs as follows:

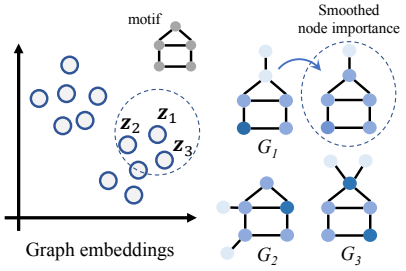


Fig. 2: Extracting reliable explanations via local smoothing.

$$\mathbf{a}_i^l = \text{AGGREGATION}^l(\{F_{i'}^{(l-1)} : n' \in \mathcal{N}_i\}),\ F_i^l = \text{COMBINE}^l(F_i^{(l-1)}, \mathbf{a}_i^l), \tag{4}$$

Algorithm 1 SAM-based node importance estimation for graph-level tasks.

Input: Encoder $f(\cdot)$, the target graph $G_n = \{\mathcal{V}, \mathcal{E}, \boldsymbol{A}, \boldsymbol{X}\} \in \mathcal{G}$.

$\boldsymbol{Z} \leftarrow f(\boldsymbol{X}, \boldsymbol{A})$; $\boldsymbol{z} \leftarrow \text{Pool}(\boldsymbol{Z})$; // $z \in \mathbb{R}^K$, the graph-level embedding of G_n

For G_n, find its m nearest-neighbor graphs $\tilde{\mathcal{N}}_n$ in the latent space;

$\tilde{w}_k^{graph} \leftarrow \text{Norm}(\boldsymbol{z} + \sum_{n' \in \tilde{\mathcal{N}}_n}^{m} \boldsymbol{z}_{n'})[k]$; // smoothed importance score

for $v_i \in G_n$ **do**

  $F_{k,i}^L \leftarrow \boldsymbol{Z}[i,k]$;

  $\psi_i \leftarrow \text{ReLU}(\sum_k \tilde{w}_k^{graph(n)} F_{k,i}^L)$; // node importance score

end for

Output: Node importance scores $\{\psi_i\}$.

where F_i^l is the embedding of v_i at l-th layer. $\mathcal{N}_i$ denotes the neighbors of v_i. Then, the SAM heat-map is calculated as:

$$\psi_i = \text{ReLU}(\textstyle\sum_k \tilde{w}_k F_{k,i}^L), \tag{5}$$

where ψ_i explains the importance score of v_i, $\tilde{w}_k$ is the smoothed importance of channel k, and L is the final layer. For graph-level tasks, the channel importance of graph G_n is estimated based on its local context, so $\tilde{w}_k^{graph(n)} = \text{Norm}(\sum_{n' \in \tilde{\mathcal{N}}_n} \text{Pool}\{F_{i'}^L : i' \in G_{n'}\})[k]$, where $\tilde{\mathcal{N}}_n$ denotes the set of neighbors graphs around G_n in the embedding space. The Pool() operation produces the heat-map of $G_{n'}$ from its nodes embeddings, where we use Average Pooling in experiments. The Norm() operation means performing L_2 normalization for heat-maps. The set $\tilde{\mathcal{N}}_n$ of neighbors can be retrieved efficiently by using quantization techniques [14]. For node-level tasks, the channel importance is estimated by averaging the heat-maps of nearby node embeddings, i.e., $\tilde{w}_k^{node(i)} = \text{Norm}(\sum_{i' \in \tilde{\mathcal{N}}_i} F_{i'}^L)[k]$, where $\tilde{\mathcal{N}}_i$ is the set of neighbors that are close to v_i in the embedding space. The explanation process is shown in Figure 1.

Both $\tilde{w}_k^N$ and $\tilde{w}_k^G$ are estimated in unsupervised learning, and they play a similar role as w_k^c in CAM. By considering nearby nodes and graphs, explanation information of the target node/graph is smoothed. Another perspective to understand SAM is that it provides explanation not only for a single node or graph, but for a group of nodes or graphs in the local manifold region.

Finally, many explanation methods have been proposed for graph neural networks (e.g., perturbation-based, substitute-based, and generation-base methods), so we want to justify our selection of CAM-style explanation in this work. The main consideration is computational efficiency. The heat-maps in CAM-style methods are directly obtainable in the feedforward process of GNNs, requiring minimal additional computation. In contrast, both perturbation-based and substitute-based methods require sending a number of perturbed inputs to estimate the effect of different graph features, which leads to significantly greater time costs. Generation-based methods face the similar issue, as they require training a global explainer on a large number of graph samples.

4.3 Explanation-Guided Contrastive Views Generation

We design two data augmentation operations utilizing explanation results, where graph components highlighted by explanation are considered as important [24,22]. The general idea is to keep important information intact, while maximally perturbing unimportant information.

Edge Perturbation. We define two masks $\boldsymbol{M}^{\text{edge},1}, \boldsymbol{M}^{\text{edge},2} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ to perturb edges and obtain different data views from the original graph. The adjacency matrix of the two new views are

$$\boldsymbol{A}_1 = \boldsymbol{A} \odot \boldsymbol{M}^{\text{edge},1},\ \boldsymbol{A}_2 = \boldsymbol{A} \odot \boldsymbol{M}^{\text{edge},2}, \tag{6}$$

where $\odot$ denotes the Hadamard product. The masks are obtained based on the explanation results:

$$M_{i,j}^{\text{edge},1} = \begin{cases} 1, & \text{if } \phi_{i,j} > \theta_e \\ \text{Bernoulli}(\phi_{i,j}) & \text{if } \phi_{i,j} \le \theta_e \end{cases},\ M_{i,j}^{\text{edge},2} = \begin{cases} 1, & \text{if } \phi_{i,j} > \theta_e \\ 1 - M_{i,j}^{\text{edge},1} & \text{if } \phi_{i,j} \le \theta_e \end{cases}, \tag{7}$$

where θ_e is a threshold, and $\phi_{i,j} = (\psi_i + \psi_j)/2$ is the edge importance between v_i and v_j. The edge importance averages the explanation scores of end nodes. The intuition of edge masking is that, with explanations, we want to maintain the important connections intact while perturbing the relatively unimportant part as much as possible. Based on the above definition, the two views are set to keep minimal mutual information while both contain task-relevant information. Here θ_e is a threshold that controls the degree of distinction between the two graph views. Empirically, we set $\theta_e = \mu_\phi + \lambda_e * \sigma_\phi$, where μ_ϕ and σ_ϕ are the mean value and standard deviation of all the edge interpretability from a graph or a batch of graphs. λ_e is the hyperparameter used to control the size of important part, because the range of importance scores can be different for different datasets.

Feature Perturbation. We also define two masks $\boldsymbol{M}^{\text{feat},1}, \boldsymbol{M}^{\text{feat},2} \in \{0,1\}^{|\mathcal{V}|\times D}$ to perturb node features. The feature matrix of the two views are

$$\boldsymbol{X}_1 = \boldsymbol{X} \odot \boldsymbol{M}^{\text{feat},1},\ \boldsymbol{X}_2 = \boldsymbol{X} \odot \boldsymbol{M}^{\text{feat},2}. \tag{8}$$

The masks are obtained based on explanation results:

$$M_{i,d}^{\text{feat},1} = \begin{cases} 1, & \text{if } \psi_i > \theta_f \\ \text{Bernoulli}(\psi_i) & \text{if } \psi_i \le \theta_f, \end{cases},\ M_{i,d}^{\text{feat},2} = \begin{cases} 1, & \text{if } \psi_i > \theta_f \\ 1 - M_i^{\text{feat},1} & \text{if } \psi_i \le \theta_f \end{cases}, \tag{9}$$

where θ_f is a threshold, and ψ_i is the explanation of node v_i. Similar to edge perturbation, we set the threshold as $\theta_f = \mu_\psi + \lambda_f * \sigma_\psi$, where μ_ψ and σ_ψ are the mean and standard deviation of node explanations from a graph or a batch of graphs. By this design, we can keep important information intact, while the features of unimportant nodes are more likely to be perturbed. Also, the two views are constructed in a way to reduce their mutual information $I(\boldsymbol{X}_1, \boldsymbol{X}_2)$.

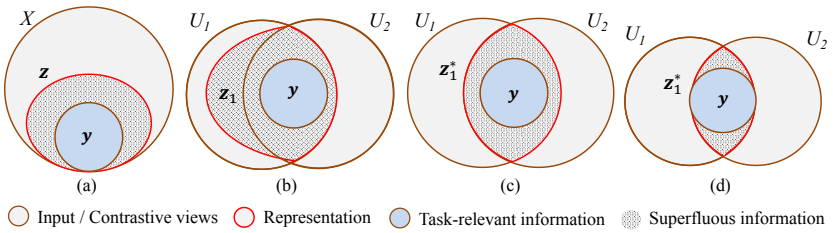


Fig. 3: Illustration of theoretical justification (best viewed in color): (a) $\boldsymbol{z}$ contains all task-relevant information and is a sufficient representation of X; (b) $\boldsymbol{z}_1$ is a sufficient representation of U_1 containing task-relevant and non-shared information; (c)(d) $\boldsymbol{z}_1^*$ is the approximate minimal sufficient representation of U_1 that only includes shared information between U_1 and U_2.

4.4 Theoretical Justification

In this part, we theoretically justify the design of our explanation guided method for building the contrastive views. The original input is denoted as X, the positive pair containing two data views obtained through augmentation is $\{U_1, U_2\}$, and $\boldsymbol{y}$ is the downstream task label.

Definition 2. *(Redundancy) The view U_1 is redundant with respect to view U_2 for $\boldsymbol{y}$ iff $I(U_1; \boldsymbol{y}|U_2) = 0$.*

The intuition behind is that a view U_1 is redundant for the task if the information in $\boldsymbol{y}$ is already observed in U_2. Let $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ denote the representation of U_1 and U_2, respectively. When U_1 and U_2 are mutually redundant, if $\boldsymbol{z}_1$ is sufficient for U_2 (refer to Definition 1 for "sufficiency"), then $\boldsymbol{z}_1$ retains task-relevant information about $\boldsymbol{y}$, as shown in Figure 3(b). The proofs for the above statements are provided in Appendix A[2].

The statements above suggest that, in contrastive learning, the task-relevance of the representations $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ depends on the quality of the constructed contrastive views U_1 and U_2, where it is crucial to keep important task-relevant information intact when building U_1 and U_2. However, in unsupervised learning, the task label is not accessible. Thus, our assumption in this work is that, the graph information important for $\boldsymbol{z}$ in unsupervised representation distribution could also be important for downstream tasks, and should be preserved in U_1 and U_2 with higher priority. This motivates the design of explanation guided contrastive views construction from Equation 6$\sim$ 9, since graph components with high importance scores could be regarded as approximately preserving the key information of graphs. In addition, to mitigate noises in a single node/graph instance, our proposed SAM method gathers the explanatory information from the local context.

Definition 3. *(Sufficient Contrastive Representation) A representation $\boldsymbol{z}_1$ of U_1 is sufficient for U_2 iff $I(\boldsymbol{z}_1; U_2) = I(U_1; U_2)$.*

[2] The appendix file is provided here: `https://github.com/sycny/ENGAGE`.

Definition 4. *(Minimal Sufficient Contrastive Representation) The sufficient contrastive representation $\boldsymbol{z}_1$ of U_1 is defined as minimal iff $I(\boldsymbol{z}_1; U_1) \leq I(\boldsymbol{z}_1'; U_1)$, $\forall \boldsymbol{z}_1'$ that is a sufficient contrastive representation.*

Let $\boldsymbol{z}_1^*$ denote the minimal sufficient contrastive representation learned from U_1. We can have $I(\boldsymbol{z}_1^*; U_2) = I(U_1; U_2)$, which implies that the shared information $I(U_1; U_2)$ is retained in the contrastive representation while the non-shared information is ignored [6]. To obtain such representations, the encoder in contrastive learning is trained so that $\boldsymbol{z}_1 \approx \boldsymbol{z}_1^*$ and $\boldsymbol{z}_2 \approx \boldsymbol{z}_2^*$ (it also implies $I(\boldsymbol{z}_1, \boldsymbol{z}_2) \approx I(U_1, U_2)$) [40]. After encoding, the minimal sufficient representation $\boldsymbol{z}_1$ (or $\boldsymbol{z}_2$) is obtained from U_1 (or U_2), and is used for prediction in downstream tasks. An illustration of $\boldsymbol{z}_1^*$ is presented in Figure 3(c). However, the representation may still contain a significant amount of superfluous information, which could degrade the downstream task performance.

Our design in Equation 6$\sim$ 9 aims to address the above problem. In our ENGAGE method, when thresholds θ_e and θ_f are set higher in Equation 6 and 8, more information in U_1 and U_2 is deleted, making $I(U_1, \boldsymbol{z}_1 | \boldsymbol{y})$ further reduced after training for $\boldsymbol{z}_1$. Thus, assuming $I(\boldsymbol{z}; \boldsymbol{y})$ is untouched (i.e., U_1 and U_2 remain mutually redundant) by setting appropriate thresholds, our proposed method reduces $I(X; \boldsymbol{z})$ by suppressing the superfluous information contained in $\boldsymbol{z}$, as depicted in Figure 3(d). On the other hand, when θ_e and θ_f values are set smaller, we keep more information in $\boldsymbol{z}$ after contrastive learning.

5 Experiments

We answer the following research questions in experiments. **RQ1**: How effective is the proposed ENGAGE method in building contrastive views and learning graph representations? **RQ2**: How does the proposed method influence representation learning as indicated by explanations? **RQ3**: What is the effect of hyperparameters (e.g., λ_e and λ_f) on the proposed method?

5.1 Experimental Setup

We apply ENGAGE to contrastive learning models for both unsupervised graph-level and node-level classification tasks. For graph classification, we follow the evaluation pipeline in [31] by training an SVM as the classifier. The whole dataset is used to learn graph-level representations, which are evaluated under the SVM classifier with cross-validation. Eight benchmark datasets are all selected from the TUDataset. We modify SimCLR and Simsiam to adapt to graph representation learning and select GIN [48] as the encoder network. Besides using contrastive learning models with random perturbation as baselines, we compare our proposed model with graph kernel methods like WL [29] and DGK [49]. We also choose state-of-the-art graph self-supervised learning methods including GraphCL [52], JOAO [51], JOAO(v2) [51], SimGRACE [46] and MVGRL [10].

For node classification, we follow [57] where the whole dataset is used to learn node representations, which are evaluated under a logistic regression classifier

Table 1: Graph classification performance comparison.

Method	NCI1	PROTEINS	DD	PTC-MR	COLLAB	RDT-B	RDT-M5K	IMDB-B	A.R.
WL	80.01 ± 0.50	72.92 ± 0.56	74.02 ± 2.28	58.00 ± 0.50	69.30 ± 3.44	68.82 ± 0.41	46.06 ± 0.21	**72.30 ± 3.44**	9.62
DGK	**80.31 ± 0.46**	73.30 ± 0.82	74.85 ± 0.74	60.10 ± 2.60	64.66 ± 0.50	78.04 ± 0.39	41.27 ± 0.18	66.96 ± 0.56	9.88
node2vec	54.89 ± 1.61	57.49 ± 3.57	—	58.60 ± 8.00	56.10 ± 0.20	—	—	—	6.25
sub2vec	52.84 ± 1.47	53.03 ± 5.55	54.33 ± 2.44	60.00 ± 6.40	55.26 ± 1.54	71.48 ± 0.41	36.68 ± 0.42	55.26 ± 1.54	13.12
graph2vec	73.22 ± 1.81	73.30 ± 2.05	70.32 ± 2.32	60.20 ± 6.90	71.10 ± 0.54	75.78 ± 1.03	47.86 ± 0.26	71.10 ± 0.54	9.62
MVGRL	77.00 ± 0.80	—	—	**62.50 ± 1.70**	**76.00 ± 1.20**	84.50 ± 0.60	—	**74.20 ± 0.70**	**3.12**
InfoGraph	76.20 ± 1.06	74.44 ± 0.31	72.85 ± 1.78	**61.70 ± 1.40**	70.65 ± 1.13	82.50 ± 1.42	53.46 ± 1.03	**73.03 ± 0.87**	7.62
GraphCL	77.87 ± 0.41	74.39 ± 0.45	**78.62 ± 0.40**	61.30 ± 2.10	71.36 ± 1.15	**89.53 ± 0.84**	55.99 ± 0.28	71.14 ± 0.44	5.38
JOAO	78.07 ± 0.47	74.55 ± 0.41	77.32 ± 0.54	—	69.50 ± 0.36	85.29 ± 1.35	55.74 ± 0.63	70.21 ± 3.08	6.75
JOAOv2	78.36 ± 0.53	74.07 ± 1.10	77.40 ± 1.15	—	69.33 ± 0.34	86.42 ± 1.45	**56.03 ± 0.27**	70.83 ± 0.25	6.12
SimGRACE	79.12 ± 0.44	75.35 ± 0.09	77.44 ± 1.11	—	71.72 ± 0.82	**89.51 ± 0.89**	55.91 ± 0.34	71.30 ± 0.77	4.00
RD-SimCLR	79.02 ± 0.52	74.61 ± 0.56	77.40 ± 0.81	58.55 ± 2.01	69.32 ± 1.54	85.67 ± 5.40	55.52 ± 0.74	69.08 ± 2.47	8.00
EG-SimCLR	**82.97 ± 0.20**	**75.44 ± 0.65**	**78.86 ± 0.51**	61.51 ± 2.41	**76.60 ± 1.26**	**90.70 ± 0.46**	**56.22 ± 0.56**	71.78 ± 0.34	**2.12**
RD-Simsiam	79.62 ± 0.59	**75.42 ± 0.50**	77.20 ± 1.33	58.55 ± 1.85	66.50 ± 2.31	86.12 ± 1.80	54.30 ± 0.64	69.86 ± 0.86	7.88
EG-Simsiam	**81.49 ± 0.19**	**76.06 ± 0.51**	**78.35 ± 0.83**	**62.67 ± 1.50**	**74.73 ± 0.79**	89.17 ± 0.43	**56.37 ± 0.17**	71.93 ± 0.40	**2.38**

with cross-validation. Popular datasets like Cora and CiteSeer are selected, and other real-world datasets, including Wiki-CS, Amazon-Computers, and Amazon-Photo are also used. Similar to graph-level tasks, we modify SimCLR and Simsiam frameworks to adapt to node representation learning. Both GCN [15] and GAT [37] models are used as encoders. We also compare with other self-supervised learning models including GCA [58], MVGRL [10], DGI [38]. We evaluate model performance using the *accuracy* metric. For each model, we report mean accuracy and standard deviation of five runs. The '—' in Table 1 and Table 2 means that these results are not available in currently published papers. The implementation details of our experiments can be found in Appendix G.

5.2 Experiment Results and Comparisons

Table 1 lists the result for graph classification, where the top three results are highlighted in bold. The proposed ENGAGE models are abbreviated as EG-SimCLR and EG-Simsiam, corresponding to their vanilla versions RD-SimCLR and RD-Simsiam, respectively. The results show that the average ranks (A.R.) of EG-SimCLR and EG-Simsiam are highest as 2.12 and 2.38, respectively, compared with other state-of-the-art methods. Among them, NCI1 has the biggest improvement of 2.66% over the most competitive baselines. Most importantly, ENGAGE methods consistently outperform random perturbation based models (e.g., RD-SimCLR and RD-Simsiam) on the eight datasets, with an average improvement of 2.90%. In addition, ENGAGE reduces performance variance compared with random perturbation, indicating that representation learning becomes more stable. These observations validate the effectiveness of using explanation toward a more adaptive and intelligent data augmentation scheme.

The result for the node classification is presented in Table 2 with the top three results highlighted. Our ENGAGE models are named as EG-SimCLR and EG-Simsiam, while their vanilla counterparts are GRACE [57] and RD-Simsiam, respectively. Many observations are similar to graph classification, such

Table 2: Node classification performance comparison.

Method	Cora	Citeseer	Wiki-CS	Amazon-Computers	Amazon-Photo	A.R
GCN	81.50 ± 0.00	70.30 ± 0.00	77.19 ± 0.12	86.51 ± 0.54	92.42 ± 0.22	11.6
GAT	83.00 ± 0.70	**72.50 ± 0.70**	77.65 ± 0.11	86.93 ± 0.29	92.56 ± 0.35	7.2
DeepWalk	70.70 ± 0.60	51.40 ± 0.50	77.21 ± 0.03	86.28 ± 0.07	90.05 ± 0.08	13.6
GAE	71.50 ± 0.40	65.80 ± 0.40	70.15 ± 0.01	85.27 ± 0.19	91.62 ± 0.13	13.6
DGI	82.30 ± 0.60	71.80 ± 0.70	75.35 ± 0.14	83.95 ± 0.47	91.61 ± 0.22	11.4
MVGRL	**86.80 ± 0.50**	**73.30 ± 0.50**	77.52 ± 0.08	87.52 ± 0.11	91.74 ± 0.07	6.8
GCA	—	—	78.35 ± 0.05	87.85 ± 0.31	92.53 ± 0.16	8.3
GRACE	82.01 ± 0.56	71.51 ± 0.33	**79.12 ± 0.15**	88.36 ± 0.18	92.52 ± 0.26	6.8
EG-SimCLR(GCN)	**84.07 ± 0.18**	**72.40 ± 0.46**	**79.21 ± 0.12**	**88.53 ± 0.13**	92.65 ± 0.19	**3.0**
GRACE(GAT)	83.57 ± 0.55	71.70 ± 0.55	78.48 ± 0.18	88.09 ± 0.15	**92.74 ± 0.20**	5.4
EG-SimCLR(GAT)	**83.78 ± 0.53**	72.28 ± 0.40	78.76 ± 0.07	**88.56 ± 0.17**	**92.97 ± 0.07**	**3.2**
RD-Simsiam(GCN)	82.65 ± 0.62	70.72 ± 0.64	78.79 ± 0.10	87.40 ± 0.31	92.53 ± 0.17	8.2
EG-Simsiam(GCN)	83.46 ± 0.56	71.49 ± 0.31	**79.27 ± 0.34**	**88.66 ± 0.19**	**92.77 ± 0.14**	**3.4**
RD-Simsiam(GAT)	80.31 ± 0.34	70.18 ± 0.54	78.61 ± 0.27	87.96 ± 0.27	92.63 ± 0.12	8.8
EG-Simsiam(GAT)	82.31 ± 0.43	70.73 ± 0.17	78.93 ± 0.20	88.28 ± 0.18	92.73 ± 0.28	6.0

as improved performance and representation learning stability. In addition, we observe that ENGAGE works better for the GCN architecture than GAT. The reason could be that the attention mechanism in GAT already equips it with some abilities to select graph components during training. The proposed ENGAGE is compatible with various GNN backbones and contrastive learning models.

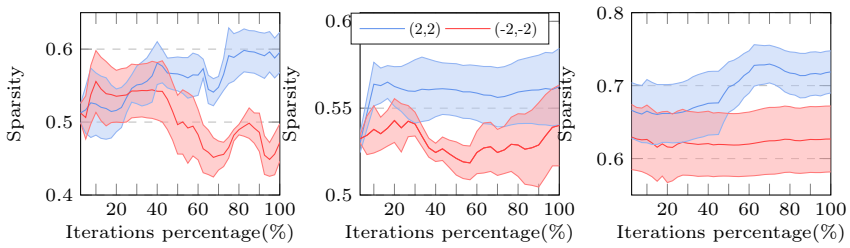
5.3 Ablation Study

We conduct ablation studies to further verify the effect of using SAM explanations for guiding graph representation learning, and the influence of hyperparameters.

Does ENGAGE remove redundant information? The explanation extracted from the model should become sparser as training goes on, since ENGAGE gradually removes superfluous information from representations. Thus, we define explanation *sparsity* [24,54] for graph G_n as $\mathcal{S}_n = 1 - (\sum_{i=1}^{|\mathcal{V}_n|} \mathbb{1}[\psi_i > \mu_\psi])/\left|\mathcal{V}_n\right|$, where $|\mathcal{V}_n|$ is the number of nodes in G_n. ψ_i is the importance score of node v_i. μ_ψ is the mean importance score of all nodes in graphs $\mathcal{G}$. The changes of explanation sparsity on DD, PROTEINS, and RDT-B at different training iterations are shown in Figure 4. We can observe that if we choose to perturb the original view radically ($\lambda_e = 2$, $\lambda_f = 2$), then sparsity tends to increase during the training process. In this mode, only the most crucial graph components will be kept, while other information will be discarded. If we perturb graphs softly ($\lambda_e = -2$, $\lambda_f = -2$), then the sparsity will remain stable or even decrease slightly, which means most of graph information is kept intact during training. We also provide visualization and quantitative analysis for explanation results of SAM in Appendix H and I, respectively.

Table 3: Effect of proposed SAM guided data augmentation.

Method	NCI1	PROTEINS	DD	PTC-MR	COLLAB	RDT-B	RDT-M5K	IMDB-B
RD-SimCLR	79.02 ± 0.52	74.61 ± 0.56	77.40 ± 0.81	58.55 ± 2.01	69.32 ± 1.54	85.67 ± 5.40	55.52 ± 0.74	69.08 ± 2.47
RD-Simsiam	79.62 ± 0.59	75.42 ± 0.50	77.20 ± 1.33	58.55 ± 1.85	66.50 ± 2.31	86.12 ± 1.80	54.30 ± 0.64	69.86 ± 0.86
HG-SimCLR	80.48 ± 0.44	74.93 ± 0.62	78.40 ± 0.69	56.62 ± 3.16	72.53 ± 0.97	89.92 ± 0.53	56.01 ± 0.25	69.88 ± 2.29
HG-Simsiam	80.70 ± 0.64	75.45 ± 0.72	77.87 ± 1.10	58.61 ± 1.99	72.84 ± 0.91	89.63 ± 0.55	55.92 ± 0.60	71.28 ± 0.82
EG-SimCLR	**82.97 ± 0.20**	75.44 ± 0.65	**78.86 ± 0.51**	61.51 ± 2.41	**76.60 ± 1.26**	**90.70 ± 0.46**	56.22 ± 0.56	71.78 ± 0.34
EG-Simsiam	81.49 ± 0.19	**76.06 ± 0.51**	78.35 ± 0.83	**62.67 ± 1.50**	74.73 ± 0.79	89.17 ± 0.43	**56.37 ± 0.17**	**71.93 ± 0.40**



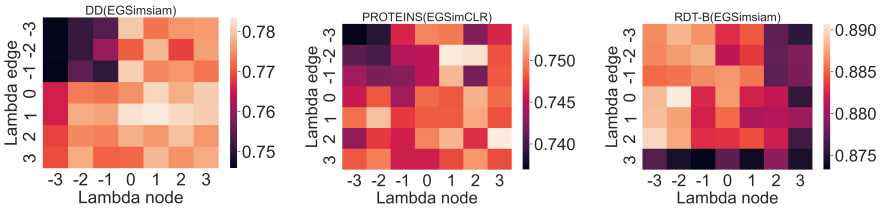
(a) DD (b) PROTEINS (c) RDT-B

Fig. 4: Evolution of explanation sparsity of three datasets during training.

SAM guided data augmentation. We first compare our proposed EG-SimCLR, EG-Simsiam models with RD-SimCLR and RD-Simsiam using random perturbation. Besides that, we formulate two variants of our method, called HG-SimCLR and HG-Simsiam ("HG" refers to "heatmap guided"), which do not employ local smoothing to obtain explanations. The comparison is in Table 3. The result shows that the explanation guided models (HG and EG models) generally perform better than models with random data augmentation, which means the explanations can be successfully leveraged to improve representation learning with contrastive views. In addition, it can be observed that our proposed SAM-based methods consistently outperform other baselines, showing that local smoothing helps capture the important information more accurately.

Hyperparameter Analysis. We now explore the effect of different (λ_e, λ_f) combinations on model performance. We conduct a detailed study on 13 datasets, where λ_e and λ_f are changed, and all the other settings are the same. Larger λ means a greater portion of data is perturbed. The results of DD, PROTEINS, and RDT-B datasets are shown in Figure 5. The results on other datasets are provided in Appendix C.

Some observations are made as follows. First, the optimal data augmentation threshold varies for different datasets. For example, Figure 5(a) and (c) show that DD benefits from more perturbation, but RDT-B does not. The reason could be that, without data augmentation and contrastive learning, the vanilla representations in different datasets contain different levels of superfluous in-



(a) DD (b) PROTEINS (c) RDT-B

Fig. 5: Model performances with different combinations of λ_e and λ_f.

formation. Second, the sensitivity to λ_e and λ_f varies across different datasets. Here we define performance gap as the difference between the best accuracy and the worst one. For graph-level and node-level tasks, COLLAB and CiteSeer have a maximum performance gap of 9.55% and 2.45%, respectively, while the RDT-M5K and Amazon-Photo have a minimum performance gap of 0.97% and 0.23%. The possible reason could be that in some datasets, the task-relevant information is redundant, so it does not hurt performance when more graph components are removed. We thus benefit more from searching for the optimal hyperparameters for datasets like COLLAB and CiteSeer. Third, Simsiam-based models generally show better stability than SimCLR-based models. The average performance gap of Simsiam-based models is 3.39% and 0.75% in graph-level and node-level tasks, respectively. By comparison, the corresponding numbers are 3.96% and 1.08% for SimCLR ones.

6 Conclusion and Future Work

In this paper, we propose an explanation guided data augmentation methods for graph representation learning. First, we design a new explanation method for interpreting unsupervised graph learning models by leveraging the information shared by local embeddings. Then we propose guided data augmentation using explanation results. We will preserve the graph structure and features whose importance exceeds the importance threshold. For the other part, we try to keep the mutual information between two views as low as possible. The final results show that our proposed methods achieve superior performance on multiple datasets. The ablation study also shows that the optimal thresholds vary across different datasets, while a general random perturbing may hurt task performance. For future work, we will (1) design more faithful unsupervised GNN explanation methods; (2) design intelligent explanation guided augmentation methods for other applications with domain knowledge.

7 Acknowledgements

The work is in part supported by NSF grant IIS-2223768. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

Ethical Statement

Our team acknowledges the importance of ethical considerations in the development and deployment of our ENGAGE framework. We ensure that our work does not lead to any potential negative societal impacts. We only use existing datasets and cite the creators of those datasets to ensure they receive proper credit. Additionally, we do not allow our work to be used for policing or military purposes. We believe it is essential to prioritize ethical considerations in all aspects of machine learning and data mining to ensure that these technologies are used for the benefit of society as a whole.

References

1. Baldassarre, F., Azizpour, H.: Explainability techniques for graph convolutional networks. arXiv preprint arXiv:1905.13686 (2019)
2. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020)
3. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15750–15758 (2021)
4. Cheng, P., Hao, W., Yuan, S., Si, S., Carin, L.: Fairfil: Contrastive neural debiasing method for pretrained text encoders. arXiv preprint arXiv:2103.06413 (2021)
5. Fang, Y., Zhang, Q., Yang, H., Zhuang, X., Deng, S., Zhang, W., Qin, M., Chen, Z., Fan, X., Chen, H.: Molecular contrastive learning with chemical element knowledge graph. arXiv preprint arXiv:2112.00544 (2021)
6. Federici, M., Dutta, A., Forré, P., Kushman, N., Akata, Z.: Learning robust representations via multi-view information bottleneck. arXiv preprint arXiv:2002.07017 (2020)
7. Gao, H., Ji, S.: Graph u-nets. In: ICML. pp. 2083–2092. PMLR (2019)
8. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)
9. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS. pp. 1024–1034 (2017)
10. Hassani, K., Khasahmadi, A.H.: Contrastive multi-view representation learning on graphs. In: International Conference on Machine Learning. pp. 4116–4126. PMLR (2020)
11. Hassani, K., Khasahmadi, A.H.: Learning graph augmentations to learn graph representations. arXiv preprint arXiv:2201.09830 (2022)
12. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020)
13. Huang, Q., Yamada, M., Tian, Y., Singh, D., Yin, D., Chang, Y.: Graphlime: Local interpretable model explanations for graph neural networks. arXiv preprint arXiv:2001.06216 (2020)
14. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with gpus. IEEE Transactions on Big Data **7**(3), 535–547 (2019)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
16. Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016)
17. Li, S., Wang, X., Zhang, A., Wu, Y.X., He, X., Chua, T.S.: Let invariant rationale discovery inspire graph contrastive learning. In: ICML (2022)
18. Liu, G., Zhao, T., Xu, J., Luo, T., Jiang, M.: Graph rationalization with environment-based augmentations. ArXiv **abs/2206.02886** (2022)
19. Lucic, A., ter Hoeve, M., Tolomei, G., de Rijke, M., Silvestri, F.: Cf-gnnexplainer: Counterfactual explanations for graph neural networks. arXiv preprint arXiv:2102.03322 (2021)
20. Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., Zhang, X.: Parameterized explainer for graph neural network. Advances in neural information processing systems **33**, 19620–19631 (2020)

21. Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., Zhang, C.: Adversarially regularized graph autoencoder for graph embedding. arXiv preprint arXiv:1802.04407 (2018)
22. Peng, X., Wang, K., Zhu, Z., You, Y.: Crafting better contrastive views for siamese representation learning. arXiv preprint arXiv:2202.03278 (2022)
23. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710 (2014)
24. Pope, P.E., Kolouri, S., Rostami, M., Martin, C.E., Hoffmann, H.: Explainability methods for graph convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10772–10781 (2019)
25. Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., Tang, J.: Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In: Proceedings of the eleventh ACM international conference on web search and data mining. pp. 459–467 (2018)
26. Schlichtkrull, M.S., De Cao, N., Titov, I.: Interpreting graph neural networks for nlp with differentiable edge masking. arXiv preprint arXiv:2010.00577 (2020)
27. Schnake, T., Eberle, O., Lederer, J., Nakajima, S., Schütt, K.T., Müller, K.R., Montavon, G.: Xai for graphs: Explaining graph neural network predictions by identifying relevant walks. arXiv preprint arXiv:2006.03589 (2020)
28. Schwarzenberg, R., Hübner, M., Harbecke, D., Alt, C., Hennig, L.: Layerwise relevance visualization in convolutional text graph classifiers. arXiv preprint arXiv:1909.10911 (2019)
29. Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-lehman graph kernels. Journal of Machine Learning Research **12**(9) (2011)
30. Shi, Y., Ma, H., Zhong, W., Mai, G., Li, X., Liu, T., Huang, J.: Chatgraph: Interpretable text classification by converting chatgpt knowledge to graphs. arXiv preprint arXiv:2305.03513 (2023)
31. Sun, F.Y., Hoffmann, J., Verma, V., Tang, J.: Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. arXiv preprint arXiv:1908.01000 (2019)
32. Tan, Q., Liu, N., Huang, X., Choi, S.H., Li, L., Chen, R., Hu, X.: S2gae: Self-supervised graph autoencoders are generalizable learners with graph masking. In: Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining. pp. 787–795 (2023)
33. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web. pp. 1067–1077 (2015)
34. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. In: European conference on computer vision. pp. 776–794. Springer (2020)
35. Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., Isola, P.: What makes for good views for contrastive learning? Advances in Neural Information Processing Systems **33**, 6827–6839 (2020)
36. Tishby, N., Pereira, F.C., Bialek, W.: The information bottleneck method. arXiv preprint physics/0004057 (2000)
37. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
38. Veličković, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D.: Deep graph infomax. arXiv preprint arXiv:1809.10341 (2018)
39. Vu, M.N., Thai, M.T.: Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. arXiv preprint arXiv:2010.05788 (2020)

40. Wang, H., Guo, X., Deng, Z.H., Lu, Y.: Rethinking minimal sufficient representation in contrastive learning. arXiv preprint arXiv:2203.07004 (2022)
41. Wang, X., Wu, Y., Zhang, A., He, X., seng Chua, T.: Causal screening to interpret graph neural networks (2021)
42. Wiltschko, A.B., Sanchez-Lengeling, B., Lee, B., Reif, E., Wei, J., McCloskey, K.J., Colwell, L., Qian, W., Wang, Y.: Evaluating attribution for graph neural networks (2020)
43. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: ICML. pp. 6861–6871. PMLR (2019)
44. Wu, X., Zhou, K., Sun, M., Wang, X., Liu, N.: A survey of graph prompting methods: Techniques, applications, and challenges. arXiv preprint arXiv:2303.07275 (2023)
45. Wu, Y., Wang, X., Zhang, A., He, X., Chua, T.S.: Discovering invariant rationales for graph neural networks. In: International Conference on Learning Representations (2022)
46. Xia, J., Wu, L., Chen, J., Hu, B., Li, S.Z.: Simgrace: A simple framework for graph contrastive learning without data augmentation. arXiv preprint arXiv:2202.03104 (2022)
47. Xu, D., Cheng, W., Luo, D., Chen, H., Zhang, X.: Infogcl: Information-aware graph contrastive learning. Advances in Neural Information Processing Systems **34** (2021)
48. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
49. Yanardag, P., Vishwanathan, S.: Deep graph kernels. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1365–1374 (2015)
50. Ying, Z., Bourgeois, D., You, J., Zitnik, M., Leskovec, J.: Gnnexplainer: Generating explanations for graph neural networks. Advances in neural information processing systems **32** (2019)
51. You, Y., Chen, T., Shen, Y., Wang, Z.: Graph contrastive learning automated. In: International Conference on Machine Learning. pp. 12121–12132. PMLR (2021)
52. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. Advances in Neural Information Processing Systems **33**, 5812–5823 (2020)
53. Yuan, H., Tang, J., Hu, X., Ji, S.: Xgnn: Towards model-level explanations of graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 430–438 (2020)
54. Yuan, H., Yu, H., Gui, S., Ji, S.: Explainability in graph neural networks: A taxonomic survey. arXiv preprint arXiv:2012.15445 (2020)
55. Zhang, Y., Defazio, D., Ramesh, A.: Relex: A model-agnostic relational model explainer. arXiv preprint arXiv:2006.00305 (2020)
56. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2921–2929 (2016)
57. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Deep graph contrastive representation learning. arXiv preprint arXiv:2006.04131 (2020)
58. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Graph contrastive learning with adaptive augmentation. In: Proceedings of the Web Conference 2021. pp. 2069–2080 (2021)