# Rethinking Explaining Graph Neural Networks via Non-parametric Subgraph Matching

**Fang Wu** [1 2] **Siyuan Li** [1] **Xurui Jin** [2] **Yinghui Jiang** [2] **Dragomir Radev** [3] **Zhangming Niu** [2] **Stan Z. Li** [1]

## Abstract

The success of graph neural networks (GNNs) provokes the question about explainability: *"Which fraction of the input graph is the most determinant of the prediction?"* Particularly, parametric explainers prevail in existing approaches because of their more robust capability to decipher the black-box (*i.e.*, target GNNs). In this paper, based on the observation that graphs typically share some common motif patterns, we propose a novel non-parametric subgraph matching framework, dubbed MatchExplainer, to explore explanatory subgraphs. It couples the target graph with other counterpart instances and identifies the most crucial joint substructure by minimizing the node corresponding-based distance. Moreover, we note that present graph sampling or node-dropping methods usually suffer from the false positive sampling problem. To alleviate this issue, we design a new augmentation paradigm named MatchDrop. It takes advantage of MatchExplainer to fix the most informative portion of the graph and merely operates graph augmentations on the rest less informative part. Extensive experiments on synthetic and real-world datasets show the effectiveness of our MatchExplainer by outperforming all state-of-the-art parametric baselines with significant margins. Results also demonstrate that MatchDrop is a general scheme to be equipped with GNNs for enhanced performance. The code is available at `https://github.com/smiles724/MatchExplainer`.

[1]School of Engineering, Westlake University, Hangzhou, China [2]MindrankAI, Hangzhou, China [3]Department of Computer Science, Yale University, New Haven, United States. Correspondence to: Zhangming Niu <zhangming@mindrank.ai>, Stan Z. Li <stan.zq.li@westlake.edu.cn>.

*Proceedings of the $40^{th}$ International Conference on Machine Learning*, Honolulu, Hawaii, USA. PMLR 202, 2023. 

## 1. Introduction

*Graph neural networks* (GNNs) have drawn broad interest due to their success in learning representations of graph-structured data, such as social networks (Fan et al., 2019), knowledge graphs (Schlichtkrull et al., 2018), traffic networks (Geng et al., 2019), and molecular graphs (Gilmer et al., 2017; Wu et al., 2023). Despite their remarkable efficacy, GNNs lack transparency as the rationale of their predictions is not easy for humans to comprehend. This prohibits practitioners from not only gaining an understanding of the network characteristics but correcting systematic patterns of mistakes made by models before deploying them in real-world applications.

Extensive studies have noticed this issue and great efforts are devoted to explaining GNNs (Yuan et al., 2020b). Researchers strive to answer questions like *"What knowledge of the input graph is the most dominantly important in the model's decision?"* To this end, feature attribution and selection (Selvaraju et al., 2017; Sundararajan et al., 2017; Ancona et al., 2017) becomes a prevalent paradigm. They distribute the model's outcome prediction to the input graph via gradient-like signals (Baldassarre & Azizpour, 2019; Pope et al., 2019; Schnake et al., 2020), mask or attention scores (Ying et al., 2019; Luo et al., 2020), or prediction changes on perturbed features (Schwab & Karlen, 2019; Yuan et al., 2021), and then choose a salient substructure as the explanation.

Apart from them, more recent approaches prefer relying on a deep learning network to parameterize the generation process of explanations (Vu & Thai, 2020; Wang et al., 2021b). These learning-based mechanisms empirically show superior accuracy than the above-mentioned non-parametric ones. Some explainer models are optimized toward local fidelity (Chen et al., 2018), such as GNNExplainer (Ying et al., 2019), PGM-Explainer (Vu & Thai, 2020) and SubgraphX (Yuan et al., 2021). Meanwhile, several others are committed to providing a global understanding of the model prediction, including PGExplainer (Luo et al., 2020), XGNN (Yuan et al., 2020a), and ReFine (Wang et al., 2021b).

Despite the fruitful progress and the popular trend towards

parametric explainers, we observe that different essential subgraph patterns are shared by different groups of graphs, which can be the key to deciphering the decision of GNNs. These frequently occurring motifs contain rich semantic meanings and indicate the characteristics of the whole graph instance (Henderson et al., 2012; Zhang et al., 2020; Banjade et al., 2021; Wu et al., 2023). For example, the hydroxide group (-OH) in small molecules typically results in higher water solubility and a carboxyl group (-COOH) usually contributes to better stability and higher boiling points. Besides that, the pivotal role of functional groups has also been proven in protein structure prediction (Senior et al., 2020).

Inspired by this inspection, we propose to mine the explanatory motif in a subgraph matching manner and design a novel non-parametric algorithm dubbed MatchExplainer, whose workflow is depicted in Fig. 1. For each pair of graphs, our MatchExplainer endeavors to explore the most crucial joint substructure by minimizing their node corresponding-based distance in the high-dimensional feature space. Then it marries the target graph iteratively with other counterpart graphs in the reference set to seek potential explanatory subgraphs. Consequently, unlike traditional explainers, the candidate explanation produced by MatchExplainer can be non-unique for the same target graph instance.

Taking a step further, we leverage the metric of mutual information from the information theory to analyze the working principle of our MatchExplainer. To be specific, we define the explanation that contains all shared information between paired graphs as *sufficient explanation*, while the explanation that contains the shared and eliminates the non-shared information as *minimal sufficient explanation*. We prove that the minimal sufficient explanation can be used to approximate the desired ground truth explanation with a theoretical guarantee. This strong relationship also provides a perspective for us to filter out the best-case substructure from all candidate explanatory subgraphs. To be precise, we propose to optimize the final candidate explanations by maximizing the difference in the prediction after the explanatory subgraph is removed from the original graph.

Last but not least, we exhibit a bonus of our MatchExplainer to be applied in enhancing the traditional graph augmentation methods. Though exhibiting strong power in preventing over-fitting and over-smoothing, present graph sampling or node-dropping mechanisms suffer from the false positive sampling problem. That is, nodes or edges of the most informative substructure are accidentally dropped or erased but the model is still required to forecast the original property, which can be misleading. To alleviate this obstacle, we take advantage of MatchExplainer and introduce a simple technique called MatchDrop. Specifically, it first digs out the explanatory subgraph by means of MatchExplainer and keeps this part unchanged. Then the graph sampling or node dropping is implemented solely on the remaining less informative part. As a consequence, the core fraction of the input graph that reveals the label information is not affected and the false positive sampling issue is effectively mitigated.

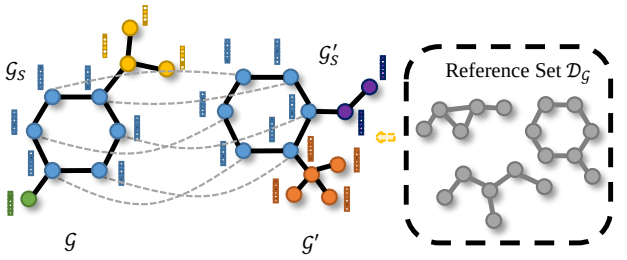


*Figure 1.* The illustration of our proposed MatchExplainer. The explanation $\mathcal{G}_S$ is attained via subgraph matching between $\mathcal{G}$ and $\mathcal{G}'$, where we minimize the accumulated node-to-node distance in the high-dimensional feature space in a greedy search manner. Since several $\mathcal{G}_S$ can be obtained by matching $\mathcal{G}$ to different counterpart graphs $\mathcal{G}'$ from the reference set $\mathcal{D}_\mathcal{G}$, we seek to find the optimal one by maximizing Equ. 10.

To summarize, we are the foremost to investigate the explainability of GNNs from the perspective of non-parametric subgraph matching to the best of our knowledge. Extensive experiments on synthetic and real-world applications demonstrate that our MatchExplainer can find the explanatory subgraphs fast and accurately with state-of-the-art performance. Additionally, we empirically show that our MatchDrop, a pragmatic application of MatchExplainer, can serve as an efficient way to promote conventional graph augmentation methods.

## 2. Preliminary and Task Description

In this section, we begin with the description of the GNN explanation task and briefly review the relevant background of graph matching and graph similarity learning (GSL).

**Explanations for GNNs.** Let $h_Y : G \; ! \; Y$ denote the well-trained GNN to be explained, which gives the prediction $\hat{Y}$ to approximate the ground truth $Y$. Without loss of generality, we consider the problem of explaining a graph classification task. Our goal is to find an explainer $h_S : G \; ! \; G_S$ that discovers the subgraph $G_S$ from input graph $G$ as:

$$\min_{h_S} R(h_Y \quad h_S(G); \hat{Y}); \text{s.t.} jh_S(G)j \quad K; \qquad (1)$$

where $R(:)$ is the risk function such as a cross-entropy loss or a mean squared error (MSE) loss, and $K$ is a constraint on the size of $G_S$ to attain a compact explanation. That is, $G_S$ has at most $K$ nodes.

**Graph matching.** As a classic combinatorial problem, graph matching is known in general NP-hard (Loiola et al.,

2007). They require expensive, complex, and impractical solvers, leading to inexact solutions (Wang et al., 2020). Given two different graphs $G_1 = (V_1; E_1)$ and $G_2 = (V_2; E_2)$ with $N_1$ and $N_2$ nodes respectively, the matching between them can be generally expressed by the quadratic assignment programming (QAP) form as (Wang et al., 2019):

$$\min_{\mathbf{T} \in \{0;1\}^{N_1 \times N_2}} \text{vec}(\mathbf{T})^T \mathbf{K} \text{vec}(\mathbf{T}); \; s.t.; \mathbf{T}\mathbf{1} = \mathbf{1}; \; \mathbf{T}^T\mathbf{1} = \mathbf{1}; \tag{2}$$

where $\mathbf{T}$ is a binary permutation matrix encoding the node correspondence, and $\mathbf{1}$ denotes a column vector with all elements to be one. $\mathbf{K}$ is the so-called affinity matrix (Leordeanu & Hebert, 2005), whose elements encode the node-to-node and edge-to-edge affinity between $G_1$ and $G_2$.

**Graph similarity learning.** GSL is a general framework for graph representation learning that requires reasoning about the structures and semantics of graphs (Li et al., 2019). We need to produce the similarity score $s(G_1; G_2)$ between them. This similarity $s(:; :)$ is typically defined by either exact matches for full-graph or sub-graph isomorphism (Berretti et al., 2001; Shasha et al., 2002), or some measure of structural similarity such as the graph edit distance (Willett et al., 1998; Raymond et al., 2002). In our setting, $s(:; :)$ depends entirely on whether these two graphs belong to the same category or share very close properties. Then for $G_1$ and $G_2$ with the same type, GSL seeks to maximize the mutual information between their representations with the joint distribution $p(G_1; G_2)$ as:

$$\max_{f_1; f_2} I(f_1(G_1); f_2(G_2); T); \tag{3}$$

where $f_1$ and $f_2$ are encoding functions. They can share the same parameter (i.e., $f_1 = f_2$) or be combined into one architecture. $T$ is the random variable representing the information required for a specific task, independent of the model selection.

## 3. The MatchExplainer Approach

The majority of recent approaches lean on parametric networks to interpret GNNs, and some early methods for GNN explanations are based on local explainability and from a single-graph view (Ying et al., 2019; Baldassarre & Azizpour, 2019; Pope et al., 2019; Schwab & Karlen, 2019). Regardless of this inclination, we argue that a non-parametric graph-graph fashion can also excavate important subgraphs and may lead to better explainability. In this work, we introduce MatchExplainer to explain GNNs via identifying the joint essential substructures by means of subgraph matching (see Algorithm 1).

### 3.1. Theoretical Analysis of MatchExplainer

From the perspective of probability theory and information theory, Equ. 1 is equivalent to maximizing the mutual information between the input graph $G$ and the subgraph $G_S$ in the context of $h_Y$. Namely, the goal of an explainer is to derive a small subgraph $G_S$ such that:

$$\max_{G_S \; G; jG_S j \; K} I(G_S; T_h); \tag{4}$$

where $I(:)$ refers to the Shannon mutual information of two random variables. Unlike $T$ which is model-agnostic, $T_h$ represents the knowledge learned by the GNN predictor $h_Y$ in a concrete downstream task. Notably, instead of merely optimizing the information hidden in $G_S$, another line of research (Yuan et al., 2021) seeks to reduce the mutual information between the remaining subgraph $G \; G_S$ and the original one $G$ as:

$$\min_{G_S \; G; jG_S j \; K} I(G \; G_S; T_h): \tag{5}$$

As an approximation of directly optimizing Equ. 4, the core idea of MatchExplainer is to fetch another graph $G^0$ that shares the same predicted property as $G$ (*i.e.*, $h_Y(G) = h_Y(G^0)$) and then extract the most relevant part between them as the explanations. To be specific, we aim to search for the best counterpart $G^0$ so that the mutual information between the input graph $G$ and the subgraph $G_S$ is maximized as:

$$\max_{G^0 \in D_G; G^0 \not\simeq G} \left[ \max_{G_S \; G; jG_S j \; K} I(G_S; G^0; T_h) \right]; \tag{6}$$

where $D_G$ denotes the reference set consisting of all available graphs, and $G_S$ is obtained by subgraph matching between $G$ and $G^0$. Similar to the information bottleneck theory (Tishby & Zaslavsky, 2015; Achille & Soatto, 2018) in supervised learning, we can define the sufficient explanation and minimal sufficient explanation of $G$ with its counterpart $G^0 \not\simeq G$ in the context of subgraph matching.

**Definition 3.1** (Sufficient Explanation)**.** Given $G^0$, the explanation $G_S^{suf}$ of $G$ is sufficient if and only if $I(G_S^{suf}; G^0; T_h) = I(G; G^0; T_h)$.

The sufficient explanation $G_S^{suf}$ of $G$ keeps all joint information with $G^0$ related to the learned information $T_h$. In other words, $G_S^{suf}$ contains all the shared information between $G$ and $G^0$. Symmetrically, the sufficient explanation for $G^0$ satisfies $I(G_S^{0suf}; G^0; T_h) = I(G; G^0; T_h)$.

**Definition 3.2** (Minimal Sufficient Explanation)**.** Given $G^0$, the sufficient explanation $G_S^{min}$ of $G$ is minimal if and only if $I(G_S^{min}; G; T_h) \quad I(G_S^{suf}; G; T_h)$.

Among all sufficient explanations, the minimal sufficient explanation $G_S^{min}$ contains the least information about $G$

with regards to the learned knowledge $T_h$. Normally, it is usually assumed that $G_S^{min}$ only maintains the shared information between $G$ and $G^0$, and eliminates other non-shared one, i.e., $I(G_S^{min}; G|G^0) = 0$.

**Theorem 3.3** (Task Relevant Information in Explanations)**.** *(Wang et al., 2022a) Given $G^0$, the minimal sufficient explanation $G_S^{min}$ contains less task-relevant information learned by $h_Y$ from input $G$ than any other sufficient explanation $G_S^{suf}$. Formally, we have:*

$$I(G; T_h) = I(G_S^{min}; T_h) + I(G; T_h|G^0)$$
$$I(G_S^{suf}; T_h) = I(G_S^{min}; T_h) + I(G_S^{suf}; G; T_h|G^0)$$
$$I(G_S^{min}; T_h). \tag{7}$$

Theorem 3.3 indicates that the mutual information between $G$ and $T_h$ can be divided into two fractions. One is $G_S^{min}$, which is the interaction between $G$ and $G^0$ associated with the learned knowledge $T_h$. The other is determined by the disjoint structure of $G$ and $G^0$ with respect to the learned information $T_h$. Our subgraph matching is committed to maximizing $I(G_S^{min}; T_h)$, which is the lower bound of $I(G; T_h)$. Notably, $I(G; T_h|G^0)$ is not completely independent to $I(G_S^{min}; T_h)$, but is instead the offset of $I(G_S^{min}; T_h)$ to $I(G; T_h)$. Hence, if we increase $I(G_S^{min}; T_h)$, $I(G; T_h|G^0)$ is minimized simultaneously. Consequently, $I(G_S^{min}; T_h)$ can be used to not only improve the lower bound of $I(G; T_h)$ but approximate $I(G; T_h)$, which is exactly our final explanatory object. This provides a firm theoretical foundation for our MatchExplainer to mine the most explanatory substructure via the subgraph matching approach.

### 3.2. Non-parametric Subgraph Exploration

**Preamble.** It is remarkable that our excavation of explanations through subgraph matching has some significant differences from either graph matching or GSL. On the one hand, graph matching algorithms (Zanfir & Sminchisescu, 2018; Sarlin et al., 2020; Wang et al., 2020; 2021a) typically establish node correspondence from a whole graph $G_1$ to another whole graph $G_2$. However, we seek to construct partial node correspondence between the subgraph of $G_1$ and the subgraph of $G_2$. On the other hand, GSL concentrates on the graph representations encoded by $f_1$ and $f_2$, as well as the ground truth information $T$ rather than the information $T_h$ learned by the GNN predictor $h_Y$.

Besides, most existing graph matching architectures (Zanfir & Sminchisescu, 2018; Li et al., 2019; Wang et al., 2020; Papakis et al., 2020; Liu et al., 2021a) are deep learning-based. They utilize a network to forecast the relationship between nodes or graphs, which has several flaws. For instance, the network needs tremendous computational resources to be trained. More importantly, its effectiveness is unreliable and may fail in certain circumstances if the network is not delicately designed. To overcome these limitations, we employ a non-parametric subgraph matching paradigm, which is totally training-free and fast to explore the most informatively joint substructure shared by any pair of input instances.

**Subgraph matching framework.** We break the target GNN $h_Y$ into two consecutive parts: $h_Y = \quad_G \quad_X$, where $\quad_G$ is the aggregator to compute the graph-level representation and predict the properties, and $\quad_X$ is the feature function to update both the node and edge features. Given a graph $G$ with node features $\mathbf{h}_i \in \mathbb{R}^{\quad v}; \forall i \in V$ and edge features $\mathbf{e}_{ij} \in \mathbb{R}^{\quad e}; \forall (i,j) \in E$, the renewed output is calculated as $\{\mathbf{h}_i^0\}_{i\in V}, \{\mathbf{e}_{ij}^0\}_{(i,j)\in E} = \quad_X \left(\{\mathbf{h}_i\}_{i\in V}, \{\mathbf{e}_{ij}\}_{(i,j)\in E}\right)$, which is forwarded into $\quad_G$ afterwards.

As analyzed before, our primary goal is to find a subgraph $G_S$ with $K$ nodes to maximize $I(G_S; G^0; T_h)$. Due to the hypothesis that the optimal counterpart $G^0$ ought to share the same explanatory substructure as $G$. Our target is equivalent to optimize $I(G_S; G_S^0; T_h)$ with $G_S \quad G$ and $G_S^0 \quad G^0$. There we utilize the node correspondence-based distance $d_G$ as a substitution for measuring $I(G_S; G_S^0; T_h)$, the shared learned information between $G_S$ and $G_S^0$. Then given a pair of $G$ and $G^0$, $d_G$ is defined and minimized as follows:

$$\min_{G_S} \min_{G; G_S^0} \quad_{G^0} d_G(G_S; G_S^0) = \min_{G_S} \min_{G; G_S^0} \quad_{G^0} \left( \min_{\mathbf{T} \in (G_S; G_S^0)} \left\langle \mathbf{T}; \mathbf{D}^{\quad x} \right\rangle \right); \tag{8}$$

where $\mathbf{D}^{\quad x}$ is the matrix of all pairwise distances between node features of $G_S$ and $G_S^0$. Its element is calculated as $\mathbf{D}_{ij}^{\quad x} = d_X(\mathbf{h}_i^0; \mathbf{h}_j^0) \ \forall i \in V; j \in V^0$, where $d_X$ is the standard vector space similarity such as the Euclidean distance and the Hamming distance. The inner optimization is conducted over $(:;:)$, which is the set of all matrices with prescribed margins defined as:

$$(G_S; G_S^0) = \left\{ \mathbf{T} \in \{0;1\}^{K \quad K} \mid \mathbf{T}\mathbf{1} = \mathbf{1}; \ \mathbf{T}^T\mathbf{1} = \mathbf{1} \right\}. \tag{9}$$

Due to the NP-hard nature of graph matching (Loiola et al., 2007), we adopt the greedy strategy to optimize $d_G(G_S; G_S^0)$ and attain the subgraph $G_S$. It is worth noting that the greedy algorithm does not guarantee to reach the globally optimal solution (Bang-Jensen et al., 2004), but can yield locally optimal solutions in a reasonable amount of time with the complexity of $O(K)$.

After that, we feed $G_S$ into $h_Y$ and examine its correctness. If $h_Y(G_S) = h_Y(G)$, then $G_S$ is regarded as the candidate explanation. Otherwise, $G_S$ is abandoned since it cannot recover the information required by $h_Y$ to predict $G$.

**Algorithm 1** Workflow of MatchExplainer

**Input:** target GNN $h_Y$, graph $G$, reference set $D_G$
Initialize an empty candidate list $D_S$.
**for** $G^0 \in D_G$ **do**
  $G_S \leftarrow \min_{G_S \subset G; G_S^0 \subset G^0} d_G(G_S; G_S^0)$ in Equ. 8
  **if** $h_Y(G_S) = h_Y(G)$ **then**
    add $G_S$ to $D_S$
  **end if**
**end for**
$G_S^+ \leftarrow \max_{G^0 \in D_S; G^0 \not\simeq G} \ _G(G^0; h_Y)$ in Equ. 10
**Return:** $G_S^+$

**Non-uniqueness of GNN explanations.** Unlike prior learning-based GNN explanation methods (Vu & Thai, 2020; Wang et al., 2021b; 2022b) that generate a unique subgraph $G_S$ for $G$, our selection of $G_S$ varies according to the choice of the counterpart $G^0 \in D_G$. Therefore, MatchExplainer can provide many-to-one explanations for a single graph $G$ once a bunch of counterparts is given. This offers a new understanding that the determinants for GNNs' predictions are non-unique, and GNNs can gain correct predictions based on several different explanatory subgraphs of the same size.

**Optimization of GNN explanations.** Since our MatchExplainer is able to discover a variety of possible explanatory subgraphs, how to screen out the most informative one becomes a critical issue. As indicated in Theorem 3.3, $I(G_S^{min}; T_h)$ is the lower bound of $I(G; T_h)$, and their difference $I(G; T_h | G^0)$ entirely depends on the selection of the matching counterpart $G^0$. Ideally, $G^0$ ought to share the exact same explanatory substructure with $G$, i.e., $G_S = G_S^0$. Meanwhile, $G$ conditioned on $G^0$ is independent to the learned knowledge $T_h$, i.e., $I(G; T_h | G^0) = 0$. Therefore, there are two distinct principles for selecting the counterpart graphs.

The first line is to seek $G^0$ that has as close the explanatory subgraph as possible to $G$. The second line is to ensure that $G$ conditioned on $G^0$ maintains little information relevant to the learned information $T_h$. Nevertheless, without sufficient domain knowledge regarding which substructure is majorly responsible for the graph property, it would be impossible for us to manually select the counterpart graph $G^0$ that satisfies $G_S \not\simeq G_S^0$.

As a remedy, we consider optimizing an opposite objective described in Equ. 5. That is, we desire to minimize the intersection between $G \setminus G_S$ and $T_h$, *i.e.*, $I(G \setminus G_S; T_h)$. Towards this goal, we remove the extracted subgraph $G_S$ from $G$ and aspire to confuse GNNs' predictions on the remaining part $G \setminus G_S$. Mathematically, the optimal $G^0$ maximizes the difference between the prediction of the whole graph and the prediction of the graph that is subtracted by $G_S$. In other words, we wish to retrieve the best explanation $G_S^+$ via:

$$\max_{G^0 \in D_S; G^0 \not\simeq G} \ _G(G^0; h_Y) = \max_{G^0 \in D_S; G^0 \not\simeq G} \left[ h_Y^c(G) - h_Y^c(G \setminus G_S) \right]; \quad (10)$$

where $c$ is the ground truth class of $G$ and $G_S$ is the substructure via subgraph matching with $G^0$. $D_S$ is the candidate subgraph set.

To summarize, given any graph $G$ and a reference graph set $D_G$, we first acquire all possible subgraphs via matching $G$ to available counterparts in $D_G$. After the pairwise subgraph matching, we calculate their corresponding $_G(\cdot; h_Y)$ and pick up the one that leads to the largest $_G(\cdot; h_Y)$ as the optimal counterpart graph. Notably, not all graphs in $D_G$ are qualified counterparts and there are several intuitive conditions that $G^0$ has to satisfy. First, $G$ and $G^0$ should belong to the same category predicted by $h_Y$. Besides, $G^0$ needs to have at least $K$ nodes. Otherwise, $G_S$ would be smaller than the given constrained size.

**Effectiveness vs. efficiency.** The time-complexity is always a vital topic to evaluate the practicability of explainers. For our MatchExplainer, the size of the reference set, *i.e.*, $|D_G|$, plays a vital role in determining the time cost since the total time cost is $O(K|D_G|)$. However, a limited number of counterpart graphs can also prohibit it from exploring better explanatory subgraphs. Thus, it is non-trivial to balance the effectiveness and efficiency of MatchExplainer by choosing an appropriate size of $D_G$.

## 4. The MatchDrop Methodology

**Prevention of the false positive sampling.** Deep graph learning faces unique challenges, such as feature data incompleteness, structural data sparsity, and over-smoothing. To address these issues, a growing number of data augmentation techniques (Hamilton et al., 2017; Rong et al., 2019) have been proposed in the graph domain and shown promising outcomes. Graph sampling and node dropping (Feng et al., 2020; Xu et al., 2021) are two commonly used mechanisms. However, most previous approaches are completely randomized, resulting in false positive sampling and injecting spurious information into the training process. For instance, *1,3-dinitrobenzene* ($C_6H_4N_2O_4$) is a mutagen molecule and its explanation is the $NO_2$ groups (Debnath et al., 1991). If any edge or node of the $NO_2$ group is accidentally dropped or destroyed, the mutagenicity property no longer exists. Therefore, it will misguide GNNs if the original label is assigned to this molecular graph after node or edge sampling.

To tackle this drawback, recall that our MatchExplainer offers a convenient way to discover the most essential part

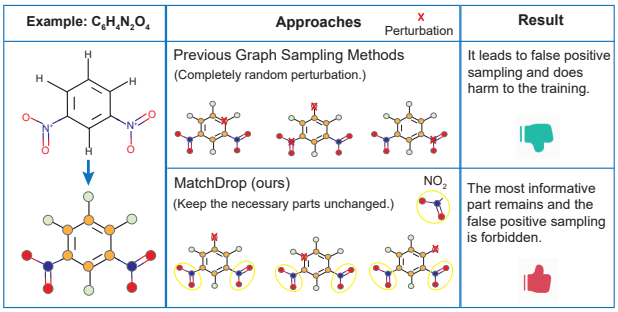


*Figure 2.* Comparison between graph augmentation with and without MatchDrop.

of a given graph. It is natural to keep this crucial portion unchanged and only drop nodes or edges in the remaining portion. Based on this idea, we propose a simple but effective method dubbed MatchDrop, which keeps the most informative part of graphs found by our MatchExplainer and alters the less informative part (see Fig. 2).

The procedure of our MatchDrop is described as follows. To begin with, we train a GNN $h_Y$ for several epochs until it converges to an acceptable accuracy, which guarantees the effectiveness of the subsequent subgraph selection. Then for each graph $G$ in training set $D_{\text{train}}$, we randomly select another graph $G' \in D_{\text{train}}$ with the same class as the counterpart graph. Afterwards, we explore its subgraph $G_S$ via MatchExplainer with a retaining ratio (i.e., $|G_S| = |G|$) and use it as the model input to train $h_Y$.

Notably, similar to the typical image augmentation skills such as rotation and flapping (Shorten & Khoshgoftaar, 2019), MatchDrop is a novel data augmentation technique for GNN training. However, instead of randomly augmenting $G$, MatchDrop reserves the most informative part and only changes the less important substructure. This significantly reduces the possibility of false positive sampling. Additionally, unlike other learnable mechanisms to inspect subgraphs, our MatchDrop is entirely parameter-free and, therefore, can be deployed at any stage of the training period.

**Training objective.** The training of GNNs is supervised by the cross entropy (CE) loss. Suppose there are $M$ classes in total, then the loss takes the following form:

$$L_S = \frac{1}{|D_{\text{train}}|} \sum_{G \in D_{\text{train}}} \sum_{c=1}^{M} Y_G \log\left(h_Y^c\left(h_S(G;\ )\right)\right); \quad (11)$$

where $h_Y^c(\cdot)$ indicates the predicted probability of $G_S$ to be of class $c$ and $Y_G$ is the ground truth value. $h_S$ employs MatchExplainer to mine the subgraph $G_S$ by matching $G$ to a randomly selected counterpart graph $G'$ in the training set $D_{\text{train}}$ with a pre-defined ratio .

## 5. Experimental Analysis

### 5.1. Datasets and Experimental Settings

Following Wang et al. (2021b), we use four standard datasets with various target GNNs.

- **Molecule graph classification**: MUTAG (Debnath et al., 1991; Kazius et al., 2005) is a molecular dataset for the graph classification problem. Each graph represents a molecule with nodes for atoms and edges for bonds. The labels are determined by their mutagenic effect on a bacterium. The well-trained Graph Isomorphism Network (GIN) (Xu et al., 2018) has approximately achieved 82% testing accuracy.
- **Motif graph classification.**: Wang et al. (2021b) create a synthetic dataset, BA-3Motif, with 3000 graphs. They use the Barabasi-Albert (BA) graphs as the base and attach each base with one of three motifs: house, cycle, or grid. We train an ASAP model (Ranjan et al., 2020) that realizes a 99.75% testing accuracy.
- **Handwriting graph classification**: Knyazev et al. (2019) transforms the MNIST images into 70K superpixel graphs with at most 75 nodes for each graph. The nodes are superpixels, and the edges are the spatial distances between them. There are ten types of digital labels. We adopt a Spline-based GNN (Fey et al., 2018) that gains around 98% accuracy in the testing set.
- **Scene graph classification**: Wang et al. (2021b) select 4443 pairs of images and scene graphs from Visual Genome (Krishna et al., 2017) to construct the VG-5 dataset (Pope et al., 2019). Each graph is labeled with one of five categories: stadium, street, farm, surfing, and forest. The regions of objects are represented as nodes, while edges indicate the relationships between object nodes. We train an AAPNP (Klicpera et al., 2018) that reaches 61.9% testing accuracy.

We compare our MatchExplainer with several state-of-the-art and popular explanation baselines, which are listed below:

- **SA** (Baldassarre & Azizpour, 2019) directly uses the gradients of the model prediction concerning the adjacency matrix of the input graph as the importance of edges.
- **Grad-CAM** (Selvaraju et al., 2017; Pope et al., 2019) uses the gradients of any target concept, such as the motif in a graph flowing into the final convolutional layer, to produce a coarse localization map highlighting the critical regions in the graph for predicting the concept.

[1]These results are reproduced

| | MUTAG | | VG-5 | | MNIST | | BA-3Motif | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | ACC-AUC | | ACC-AUC | | ACC-AUC | | ACC-AUC | | Recall@ 5 | |
| SA | 0:769 | | 0:769 | | 0:559 | | 0:518 | | 0:243 | |
| Grad-CAM | 0:786 | 0:011 | 0:909 | 0:005 | 0:581 | 0:009 | 0:533 | 0:003 | 0:212 | 0:002 |
| GNNExplainer | 0:895 | 0:010 | 0:895 | 0:003 | 0:535 | 0:013 | 0:528 | 0:005 | 0:157 | 0:002 |
| PG-Explainer | 0:631 | 0:008 | 0:790 | 0:004 | 0:504 | 0:010 | 0:586 | 0:004 | 0:293 | 0:001 |
| PGM-Explainer | 0:714 | 0:007 | 0:792 | 0:001 | 0:615 | 0:003 | 0:575 | 0:002 | 0:250 | 0:000 |
| ReFine | 0:955 | 0:005 | 0:914 | 0:001 | 0:636 | 0:003 | 0:576 | 0:013[1] | 0:297 | 0:000[1] |
| MatchExplainer | **0.997** | | **0.993** | | **0.938** | | **0.634** | | **0.305** | |
| Relative Impro. | 4:5% | | 8:6% | | 48:9% | | 8:1% | | 2:6% | |

*Table 1.* Comparisons of our MatchExplainer with other baseline explainers.

- **GNNExplainer** (Ying et al., 2019) optimizes soft masks for edges and node features to maximize the mutual information between the original predictions and new predictions.
- **PGExplainer** (Luo et al., 2020) hires a parameterized model to decide whether an edge is essential, which is trained over multiple explained instances with all edges.
- **PGM-Explainer** (Vu & Thai, 2020) collects the prediction change on the random node perturbations and then learns a Bayesian network from these perturbation-prediction observations to capture the dependencies among the nodes and the prediction.
- **ReFine** (Wang et al., 2021b) exploits the pre-training and fine-tuning idea to develop a multi-grained GNN explainer. It has a global understanding of model workings and local insights on specific instances.

As the ground-truth explanations are usually unknown, it is tough to evaluate the excellence of explanations quantitatively. There, we follow Wang et al. (2021b) and employ **the predictive accuracy (ACC@$\rho$)** and **Recall@$N$** as the metrics. Specifically, ACC@ measures the fidelity of the explanatory subgraphs by forwarding them into the target model and examining how well it recovers the target prediction. ACC-AUC is reported as the area under the ACC curve over different selection ratios $2\ f0{:}1; 0{:}2; :::; 1{:}0g$. Recall@$N$ is computed as $\mathsf{E}_G\,[jG_S \setminus G_S j = jG_S j]$, where $G_S$ is the ground-truth explanatory subgraph. Remarkably, Recall@$N$ is only suitable for BA3-motif since this dataset is synthetic and the motifs are foregone.

### 5.2. Can MatchExplainer Find Better Explanations?

**Quantitative results.** To investigate the effectiveness of MatchExplainer, we conduct broad experiments on four datasets, and the comparisons are reported in Table 1. For MUTAG, VG-5, and BA3-Motif, we exploit the full training and validation data as the reference set. For MNIST, we randomly select 10% available samples as the reference set to speed up matching. It can be found that MatchExplainer outperforms every baseline in all cases. Particularly, previous explainers fail to explain GNNs well in MNIST with ACC-AUCs lower than 65%, but MatchExplainer can reach as high as 93.8%. And if we use the whole training and validation data in MNIST as the reference, its ACC-AUC can increase to 97.2%. This phenomenon demonstrates the advantage of subgraph matching in explaining GNNs when the dataset has clear patterns of explanatory subgraphs. Additionally, MatchExplainer also achieves significant relative improvements over the strongest baseline by 8.6% and 8.1% in VG-5 and BA3-Motif, respectively.

Furthermore, it is also worth noting that MatchExplainer realizes nearly 100% ACC-AUCs in each task but BA-3Motif. For BA-3Motif, we find that its predictive accuracy are $[0{:}31; 0{:}31; 0{:}31; 0{:}34; 0{:}49; 0{:}71; 0{:}97; 1{:}0; 1{:}0; 1{:}0]$ with different selection ratios. This aligns with the fact that most motifs in this task occupy a large fraction of the whole graph. Once the selection ratio is greater than 0.7, MatchExplainer can figure out the correct explanatory subgraph.

**Visualization.** In addition, we envision the explanations of MatchExplainer on MUTAG in Appendix C for qualitative evaluations. We also compare the efficiency of our MatchExplainer with other parametric methods in Appendix 3. It can be discovered that MatchExplainer enjoys a competitive fast inference speed with no additional training cost, making it possible for large-scale deployment.

### 5.3. Can MatchDrop Improve the Performance of GNNs?

**Implementations.** We take account of two backbones: GCN (Kipf & Welling, 2016), and GIN (Xu et al., 2018) with a depth of 6. Similar to Rong et al. (2019), we adopt a random hyper-parameter search for each architecture to enable more robust comparisons. There, *DropNode* stands for randomly sampling subgraphs, which can be also treated as a specific form of node dropping. False-positive drop (*FP-Drop*) is the opposite operation of our MatchDrop, where the subgraph sampling or node dropping is only performed in the explanatory subgraphs while the rest remains the same.

| Dataset | Backbone | Original | | FPDrop | | DropNode | | PGDrop | | MatchDrop | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| MUTAG | GCN | 0:828 | 0:004 | 0:803 | 0:017 | 0:832 | 0:008 | 0:825 | 0:02 | **0.844** | **0.006** |
| | GIN | 0:832 | 0:003 | 0:806 | 0:020 | 0:835 | 0:009 | 0:828 | 0:01 | **0.845** | **0.007** |
| VG-5 | GCN | 0:619 | 0:003 | 0:587 | 0:014 | 0:623 | 0:007 | 0:604 | 0:002 | **0.638** | **0.008** |
| | GIN | 0:621 | 0:004 | 0:593 | 0:018 | 0:622 | 0:006 | 0:600 | 0:004 | **0.630** | **0.003** |
| MNIST | GCN | 0:982 | 0:001 | 0:955 | 0:008 | 0:982 | 0:002 | 0:975 | 0:003 | **0.986** | **0.002** |
| | GIN | 0:988 | 0:001 | 0:959 | 0:005 | 0:989 | 0:001 | 0:979 | 0:002 | **0.990** | **0.001** |

*Table 2.* Testing accuracy (%) comparisons on different backbones with and without MatchDrop.

We add FPDrop as a baseline to help unravel the reason why MatchDrop works. *PGDrop* is similar to MatchDrop, but uses a fixed PGExplainer (Luo et al., 2020) to explore the informative substructure. The selection ratios for FPDrop, PGDrop, and MatchDrop are all set as 0.95.

**Overall results.** Table 2 documents the performance on all datasets except BA-3Motif, since its testing accuracy has already approached 100%. It can be observed that MatchDrop consistently promotes the testing accuracy for all cases. Exceptionally, FPdrop imposes a negative impact over the performance of GNNs. This indicates that false positive sampling does harm to the conventional graph augmentation methods, which can be surmounted by our MatchDrop effectively. On the other hand, PGDrop also gives rise to the decrease of accuracy. One possible reason is that parameterized explainers like PGExplainr are trained on samples that GNNs predict correctly, so they are incapable to explore explanatory subgraphs on unseen graphs that GNNs forecast mistakenly.

## 6. Related Work

### 6.1. Explainability of GNNS

Interpretability and feature selection have been attached to growing significance in demystifying complicated deep learning models, and increasing interests have been appealed in explaining GNNs (Ying et al., 2019; Wu et al., 2022). Despite fruitful progress, the study in this area is still insufficient compared to the domain of images and natural languages. Generally, there are two mainstream lines of research. The widely-adopted one nowadays is the parametric explanation method. They run a parameterized model to dig out informative substructures or generate the saliency maps. For example, GNNExplainer (Ying et al., 2019) learns soft masks for each instance and applies them to the adjacency matrix. PGExplainer (Luo et al., 2020) collectively explains multiple samples with a probabilistic graph generative model. XGNN (Yuan et al., 2020a) utilizes a graph generator to output class-wise graph patterns to explain GNNs for each class. PGM-Explainer (Vu & Thai, 2020) proposes a Bayesian network on the pairs of graph perturbations and prediction changes. The other line is the non-parametric explanation methods, which do not involve any additional trainable models. They employ some heuristics like gradient-like scores obtained by backpropagation as the feature contributions of a specific instance (Baldassarre & Azizpour, 2019; Pope et al., 2019; Schnake et al., 2020). As mentioned, the latter is usually less favored because their performance is much poorer than the former parametric methods. In contrast, our MatchExplainer procures state-of-the-art results astonishingly.

### 6.2. Graph Augmentations

Data augmentation has recently attracted growing attention in graph representation learning to counter issues like data noise and data scarcity (Zhao et al., 2022). The related work can be roughly broken down into *feature-wise* (Zhang et al., 2017; Liu et al., 2021b; Taguchi et al., 2021), *structure-wise* (You et al., 2020; Zhao et al., 2021a), and *label-wise* (Verma et al., 2019) categories based on the augmentation modality (Ding et al., 2022). Among them, many efforts are raised to augment the graph structures. Compared with adding or deleting edges (Xu et al., 2022), the augmentation operations on node sets are more complicated. A typical application is to promote the propagation of the whole graph by inserting a supernode (Gilmer et al., 2017), while Zhao et al. (2021b) interpolate nodes to enrich the minority classes. On the contrary, some implement graph or subgraph sampling by dropping nodes for different purposes, such as scaling up GNNs (Hamilton et al., 2017), enabling contrastive learning (Qiu et al., 2020), and prohibiting over-fitting and over-smoothing (Rong et al., 2019). Nonetheless, few of those graph sampling or node dropping approaches manage to find augmented graph instances from the input graph that best preserve the original properties.

## 7. Conclusion

This paper proposes a promising subgraph matching technique called MatchExplainer for GNN explanations. Distinct from the popular trend of using a parameterized network that lacks interpretability, we design a non-parametric algorithm to search for the most informative joint subgraph between a pair of graphs with theoretical guarantees. Furthermore, we combine MatchExplainer with the classic

graph augmentation method and show its great capacity in ameliorating the false positive sampling challenge. Experiments convincingly demonstrate the efficacy of our MatchExplainer by winning over parametric approaches with significant margins. Our work hopes to push the frontier of non-parametric methods to explain deep learning models.

## References


Achille, A. and Soatto, S. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980, 2018.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.

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

Bang-Jensen, J., Gutin, G., and Yeo, A. When the greedy algorithm fails. *Discrete optimization*, 1(2):121–127, 2004.

Banjade, H. R., Hauri, S., Zhang, S., Ricci, F., Gong, W., Hautier, G., Vucetic, S., and Yan, Q. Structure motif–centric learning framework for inorganic crystalline systems. *Science advances*, 7(17):eabf1754, 2021.

Berretti, S., Del Bimbo, A., and Vicario, E. Efficient matching and indexing of graph models in content-based retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1089–1105, 2001.

Chen, J., Song, L., Wainwright, M., and Jordan, M. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pp. 883–892. PMLR, 2018.

Debnath, A. K., Lopez de Compadre, R. L., Debnath, G., Shusterman, A. J., and Hansch, C. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry*, 34 (2):786–797, 1991.

Ding, K., Xu, Z., Tong, H., and Liu, H. Data augmentation for deep graph learning: A survey. *arXiv preprint arXiv:2202.08235*, 2022.

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

Feng, W., Zhang, J., Dong, Y., Han, Y., Luan, H., Xu, Q., Yang, Q., Kharlamov, E., and Tang, J. Graph random neural networks for semi-supervised learning on graphs. *Advances in neural information processing systems*, 33: 22092–22103, 2020.

Fey, M., Lenssen, J. E., Weichert, F., and Müller, H. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 869–877, 2018.

Geng, X., Li, Y., Wang, L., Zhang, L., Yang, Q., Ye, J., and Liu, Y. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 3656–3663, 2019.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Henderson, K., Gallagher, B., Eliassi-Rad, T., Tong, H., Basu, S., Akoglu, L., Koutra, D., Faloutsos, C., and Li, L. Rolx: structural role extraction & mining in large graphs. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1231–1239, 2012.

Kazius, J., McGuire, R., and Bursi, R. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of medicinal chemistry*, 48(1):312–320, 2005.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Klicpera, J., Bojchevski, A., and Günnemann, S. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018.

Knyazev, B., Taylor, G. W., and Amer, M. Understanding attention and generalization in graph neural networks. *Advances in neural information processing systems*, 32, 2019.

Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.-J., Shamma, D. A., et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

Leordeanu, M. and Hebert, M. A spectral technique for correspondence problems using pairwise constraints. 2005.

Li, Y., Gu, C., Dullien, T., Vinyals, O., and Kohli, P. Graph matching networks for learning the similarity of graph structured objects. In *International conference on machine learning*, pp. 3835–3845. PMLR, 2019.

Liu, L., Hughes, M. C., Hassoun, S., and Liu, L. Stochastic iterative graph matching. In *International Conference on Machine Learning*, pp. 6815–6825. PMLR, 2021a.

Liu, S., Dong, H., Li, L., Xu, T., Rong, Y., Zhao, P., Huang, J., and Wu, D. Local augmentation for graph neural networks. *arXiv preprint arXiv:2109.03856*, 2021b.

Loiola, E. M., de Abreu, N. M. M., Boaventura-Netto, P. O., Hahn, P., and Querido, T. A survey for the quadratic assignment problem. *European journal of operational research*, 176(2):657–690, 2007.

Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., and Zhang, X. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

Papakis, I., Sarkar, A., and Karpatne, A. Gcnnmatch: Graph convolutional neural networks for multi-object tracking via sinkhorn normalization. *arXiv preprint arXiv:2010.00067*, 2020.

Pope, P. E., Kolouri, S., Rostami, M., Martin, C. E., and 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.

Qiu, J., Chen, Q., Dong, Y., Zhang, J., Yang, H., Ding, M., Wang, K., and Tang, J. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1150–1160, 2020.

Ranjan, E., Sanyal, S., and Talukdar, P. Asap: Adaptive structure aware pooling for learning hierarchical graph representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5470–5477, 2020.

Raymond, J. W., Gardiner, E. J., and Willett, P. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal*, 45(6):631–644, 2002.

Rong, Y., Huang, W., Xu, T., and Huang, J. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019.

Sarlin, P.-E., DeTone, D., Malisiewicz, T., and Rabinovich, A. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4938–4947, 2020.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Berg, R. v. d., Titov, I., and Welling, M. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pp. 593–607. Springer, 2018.

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

Schwab, P. and Karlen, W. Cxplain: Causal explanations for model interpretation under uncertainty. *Advances in Neural Information Processing Systems*, 32, 2019.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Senior, A. W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Žídek, A., Nelson, A. W., Bridgland, A., et al. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792): 706–710, 2020.

Shasha, D., Wang, J. T., and Giugno, R. Algorithmics and applications of tree and graph searching. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 39–52, 2002.

Shorten, C. and Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

Taguchi, H., Liu, X., and Murata, T. Graph convolutional networks for graphs containing missing features. *Future Generation Computer Systems*, 117:155–168, 2021.

Tishby, N. and Zaslavsky, N. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pp. 1–5. IEEE, 2015.

Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., and Bengio, Y. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pp. 6438–6447. PMLR, 2019.

Vu, M. and Thai, M. T. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. *Advances in neural information processing systems*, 33: 12225–12235, 2020.

Wang, H., Guo, X., Deng, Z.-H., and Lu, Y. Rethinking minimal sufficient representation in contrastive learning. *arXiv preprint arXiv:2203.07004*, 2022a.

Wang, R., Yan, J., and Yang, X. Learning combinatorial embedding networks for deep graph matching. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3056–3065, 2019.

Wang, R., Yan, J., and Yang, X. Combinatorial learning of robust deep graph matching: an embedding based approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Wang, R., Yan, J., and Yang, X. Neural graph matching network: Learning lawler's quadratic assignment problem with extension to hypergraph and multiple-graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021a.

Wang, X., Wu, Y., Zhang, A., He, X., and Chua, T.-S. Towards multi-grained explainability for graph neural networks. *Advances in Neural Information Processing Systems*, 34, 2021b.

Wang, X., Wu, Y., Zhang, A., Feng, F., He, X., and Chua, T.-S. Reinforced causal explainer for graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022b.

Willett, P., Barnard, J. M., and Downs, G. M. Chemical similarity searching. *Journal of chemical information and computer sciences*, 38(6):983–996, 1998.

Wu, F., Li, S., Wu, L., Radev, D. R., and Li, S. Z. Discovering and explaining the representation bottleneck of graph neural networks from multi-order interactions. *arXiv preprint arXiv:2205.07266*, 2022.

Wu, F., Radev, D., and Li, S. Z. Molformer: motif-based transformer on 3d heterogeneous molecular graphs. *Rn*, 1:1, 2023.

Xu, D., Cheng, W., Luo, D., Chen, H., and Zhang, X. Infogcl: Information-aware graph contrastive learning. *Advances in Neural Information Processing Systems*, 34, 2021.

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

Xu, Z., Du, B., and Tong, H. Graph sanitation with application to node classification. In *Proceedings of the ACM Web Conference 2022*, pp. 1136–1147, 2022.

Ying, Z., Bourgeois, D., You, J., Zitnik, M., and Leskovec, J. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33: 5812–5823, 2020.

Yuan, H., Tang, J., Hu, X., and 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, 2020a.

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

Yuan, H., Yu, H., Wang, J., Li, K., and Ji, S. On explainability of graph neural networks via subgraph explorations. In *International Conference on Machine Learning*, pp. 12241–12252. PMLR, 2021.

Zanfir, A. and Sminchisescu, C. Deep learning of graph matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2684–2693, 2018.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Zhang, S., Hu, Z., Subramonian, A., and Sun, Y. Motif-driven contrastive learning of graph representations. *arXiv preprint arXiv:2012.12533*, 2020.

Zhao, T., Liu, Y., Neves, L., Woodford, O., Jiang, M., and Shah, N. Data augmentation for graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11015–11023, 2021a.

Zhao, T., Zhang, X., and Wang, S. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pp. 833–841, 2021b.

Zhao, T., Liu, G., Günnemann, S., and Jiang, M. Graph data augmentation for graph machine learning: A survey. *arXiv preprint arXiv:2202.08871*, 2022.

## A. The Greedy Algorithm for Subgraph Matching

Here we provide the pseudo-code of subgraph matching in a greedy way. Given two graphs $G$ and $G'$, we first feed them to a GNN $h_Y$ and obtain their corresponding node features as $\{\mathbf{h}_i\}_{i \in V}$ and $\{\mathbf{h}'_i\}_{i \in V'}$. Our goal is to find subgraphs $G_S$ and $G'_S$ such that $d_G(G_S, G'_S)$ is minimized.

**Algorithm 2** Greedy Algorithm to Explore Shared Subgraphs

**Input:** node features $\{\mathbf{h}_i\}_{i \in V}$ and $\{\mathbf{h}'_i\}_{i \in V'}$, subgraph size $K$
Initialize an empty list $g$ to store the selected nodes.
Compute the distance matrix $\mathbf{D}^{x}$, where $\mathbf{D}_{ij}^{x} = d_X(\mathbf{h}_i, \mathbf{h}'_j) \forall i \in V, \forall j \in V'$.
**for** $t = 1, \ldots, K$ **do**
  $(i, j) \leftarrow \min_{i \in V, j \in V'} \mathbf{D}_{ij}^{x}$,
  add $i$ to $g$,
  remove $i$ from $V$,
  remove $j$ from $V'$,
**end for**
**Return:** $g$

## B. Experimental Details and Additional Results

**Explaining GNNs.** All experiments are conducted on a single A100 PCIE GPU (40GB). For the parametric methods containing GNNExplainer, PGExplainer, PGM-Explainer, and Refine, we use the reported performance in Wang et al. (2021b). Regarding the re-implementation of Refine in BA-3Motif, we use the original code with the same hyperparameters, and we adopt Adam optimizer (Kingma & Ba, 2014) and set the learning rate of pre-training and fine-tuning as 1e-3 and 1e-4, respectively.

**Graph augmentations.** All experiments are also implemented on a single A100 PCIE GPU (40GB). We employ three sorts of different GNN variants (GCN, GAT, and GIN) to fit these datasets and verify the efficacy of various graph augmentation methods. We employ Adam optimizer for model training. For MUTAG, the batch size is 128, and the learning rate is 1e-3. For BA3-Motif, the batch size is 128, and the learning rate is 1e-3. For VG-5, the batch size is 256, and the learning rate is 0.5 * 1e-3. We fix the number of epochs to 100 for all datasets.

**Efficiency studies.** We compute the average inference time cost for each dataset with different methods to obtain explanations of a single graph. We also count the overall training and inference time expenditure and summarize the results in Table 3. Specifically, we train GNNExplainer and PG-Explainer for 10 epochs, and pre-train ReFine for 50 epochs before evaluation. It can be observed that though prior approaches enjoy fast inference speed, they suffer from long-term training phases. As an alternative, our MatchExplainer is completely training-free. When comparing the total time, MatchExplainer is the least computationally expensive in MUTAG, VG-5, and MNIST. However, as most motifs in BA-3Motif are large-size, MatchExplainer has to traverse a large reference set to obtain appropriate counterpart graphs, which unavoidably results in spending far more time.

## C. Explanations for Graph Classification Models

In this section, we report visualizations of explanations in Figure 3.

*Table 3.* Efficiency studies of different methods (in seconds).

| Method | Phase | MUTAG | VG-5 | MNIST | BA-3Motif |
|---|---|---|---|---|---|
| GNNExplainer | Training | 186.0 | 1127.2 | 1135.4 | 66.1 |
| | Inference (per graph) | 1.290 | 0.565 | 0.732 | 0.517 |
| | Training + Inference (total) | 703.4 | 1644.6 | 1782.1 | 271.6 |
| PG-Explainer | Training | 186.3 | 286.3 | 1154.1 | 112.4 |
| | Inference (per graph) | 0.056 | 0.094 | 0.025 | 0.020 |
| | Training + Inference (total) | 208.6 | 309.5 | 1162.1 | **120.4** |
| ReFine | Training | 1191.6 | 1933.3 | 5025.8 | 763.0 |
| | Inference (per graph) | 0.068 | 0.107 | 0.026 | 0.027 |
| | Training + Inference (total) | 1218.9 | 1959.7 | 5051.2 | 773.8 |
| MatchExplainer | Training | – | – | – | – |
| | Inference (per graph) | 0.485 | 0.732 | 0.682 | 7.687 |
| | Training + Inference (total) | **194.6** | **180.3** | **667.8** | 224.7 |

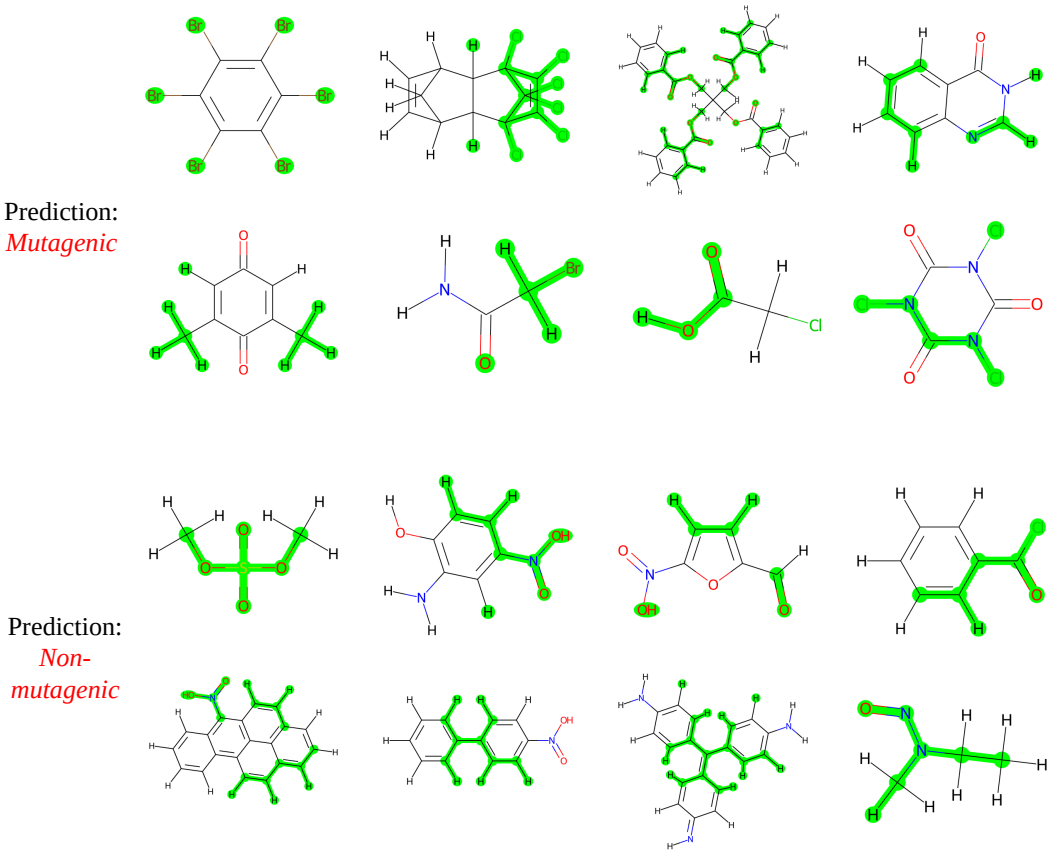


*Figure 3.* Explanatory subgraphs in Mutagenicity, where 50% nodes are highlighted.