MAGE: MODEL-LEVEL GRAPH NEURAL NETWORKS EXPLANATIONS VIA MOTIF-BASED GRAPH GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph Neural Networks (GNNs) have shown remarkable success in molecular tasks, yet their interpretability remains challenging. Traditional model-level explanation methods like XGNN and GNNInterpreter often fail to identify valid substructures like rings, leading to questionable interpretability. This limitation stems from XGNN's atom-by-atom approach and GNNInterpreter's reliance on average graph embeddings, which overlook the essential structural elements crucial for molecules. To address these gaps, we introduce an innovative **M**otif-b**A**sed **G**NN **E**xplainer (MAGE) that uses motifs as fundamental units for generating explanations. Our approach begins with extracting potential motifs through a motif decomposition technique. Then, we utilize an attention-based learning method to identify class-specific motifs. Finally, we employ a motif-based graph generator for each class to create molecular graph explanations based on these class-specific motifs. This novel method not only incorporates critical substructures into the explanations but also guarantees their validity, yielding results that are human-understandable. Our proposed method's effectiveness is demonstrated through quantitative and qualitative assessments conducted on six real-world molecular datasets.

1 INTRODUCTION

Graph Neural Networks (GNNs) (Kipf & Welling, 2016; Xu et al., 2018; Gao et al., 2018; Rong et al., 2020; Sun et al., 2022; Wang et al., 2023) have become increasingly popular tools for modeling data in the molecule field. As an effective method for learning representations from molecule data, GNNs have attained state-of-the-art results in tasks like molecular representation learning (Gao & Ji, 2019; Yu & Gao, 2022a; Fang et al., 2022; Zang et al., 2023), and molecule generation (Bongini et al., 2021; Lai et al., 2021). Despite their growing popularity, questions arise about the trustworthiness and decision-making processes of GNNs.

Explaining GNNs has become a major area of interest in recent years and existing methods can be broadly categorized into instance-level explanations and model-level explanations. Instance-level explanations (Ying et al., 2019; Baldassarre & Azizpour, 2019; Pope et al., 2019; Huang et al., 2020; Vu & Thai, 2020; Schnake et al., 2020; Luo et al., 2020; Funke et al., 2020; Zhang et al., 2021a; Shan et al., 2021; Wang et al., 2021; Yuan et al., 2021; Wang et al., 2022; Yu & Gao, 2022b) aim to pinpoint specific nodes, edges, or subgraphs crucial for a GNN model's predictions on one data instance. Model-level explanations (Yuan et al., 2020; Wang & Shen, 2022) seek to demystify the overall behavior of the GNN model by identifying patterns that generally lead to certain predictions. While instance-level methods offer detailed insights, they require extensive analysis across numerous examples to be deemed reliable. In addition, for instance-level explainer, there is no requirement for an explanation to be valid (e.g., a chemically valid molecular graph). Conversely, model-level methods provide more high-level and broader insights explanations that require less intensive human oversight.

Model-level explanation methods have two main categories: concept-based methods and generation-based methods. Concept-based explanation methods focus on identifying higher-level concepts that significantly influence the model's predictions (Azzolin et al., 2022; Xuanyuan et al., 2023). Also, these methods establish rules to illustrate how these concepts are interconnected, like using logical

formulas (Azzolin et al., 2022). Generation-based models try to learn a generative model that can generate synthetic graphs that are optimized to maximize the behavior of the target GNN (Yuan et al., 2020; Wang & Shen, 2022). Instead of defining the relationships between important concepts with formulas, generation-based models can generate novel graph structures that are optimized for specific properties. This capability is particularly crucial in the field of molecular learning.

Despite the notable benefits of generation-based model-level explanation methods for GNNs, they have received relatively low attention. Presently, the available generation-based model-level explainers for GNNs are somewhat limited in their application, especially in molecular graphs. For example, XGNN (Yuan et al., 2020) explains models by building an explanation atom-by-atom and sets a maximum degree for each atom to maintain the explanation's validity. On the other hand, GNNInterpreter (Wang & Shen, 2022) adopts a more flexible approach by learning a generative explanation graph distribution. However, it ensures the validity of explanations by maximizing the similarity between the explanation graph embedding and the average embedding of all graphs, which is not particularly effective for molecular graphs. These existing methods often fail to consider the unique aspects of molecular structures adequately. This oversight makes it challenging for these methods to include crucial substructures in their explanation graphs. It compromises the validity and reliability of the model-level explanations they provide for molecular graphs. In Figure 1, the explanations produced by both XGNN and GNNInterpreter exhibit low validity, making them ineffective for further analysis. Consequently, there's a clear need for more advanced and specialized model-level explanation methods that can effectively handle the unique characteristics of molecular graphs and provide meaningful insights into how GNNs interpret and process molecular data.

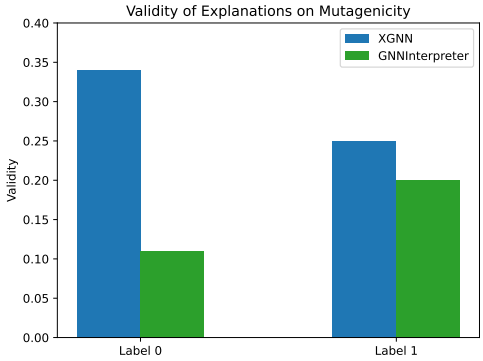


Figure 1: Validity of explanations generated by XGNN and GNNInterpreter on Mutagenicity dataset

This paper introduces a motif-based approach for model-level explanations, MAGE, specifically tailored for GNNs in molecular representation learning tasks. The method begins by identifying all possible motifs from the molecular data. Following this, our method focuses on pinpointing particularly significant motifs for a specific class. This is achieved through a novel attention-based motif learning process, which aids in selecting these key motifs. Once these important motifs are identified, we employ a motif-based graph generator trained to produce explanations based on these motifs. Our experimental results show that our method reliably produces explanations featuring complex molecular structures, achieving complete validity across various molecular datasets. Furthermore, both quantitative and qualitative evaluations demonstrate that the explanation molecules created by our method are more representative and human-understandable than those generated by baseline methods.

2 PRELIMINARY

Notations. A molecular dataset can be depicted as $\mathcal{G} = \{G_1, G_2, ..., G_i, ..., G_n\}$, where each molecular graph within this set is denoted by $G_i = (V, E)$. The label of each graph will be assigned by $l_i \in L = \{l_1, l_2, ...\}$ In this representation, V constitutes a set of atoms, each treated as a node, while E comprises the bonds, represented as edges connecting these nodes. The structural information of each graph G_i is encoded in several matrices. The adjacency matrix $\boldsymbol{A} \in \{0, 1\}^{N \times N}$ represents the connections between atoms, where a '1' indicates the presence of a bond between a pair of atoms, a '0' signifies no bond, and N represents the total number of atoms in the graph. Additionally, an atom feature matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ captures the features of each atom, with D being the number of features characterizing each atom. Similarly, a bond feature matrix $\boldsymbol{Z} \in \mathbb{R}^{N \times D_b}$ is used to describe the attributes of the bonds, where D_b denotes the number of features describing each bond.

Graph Neural Networks. Graph Neural Networks (GNNs) leverage adjacency, node feature, and edge feature matrices to learn node features. Despite the existence of various GNN variants like Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), and Graph Isomorphism

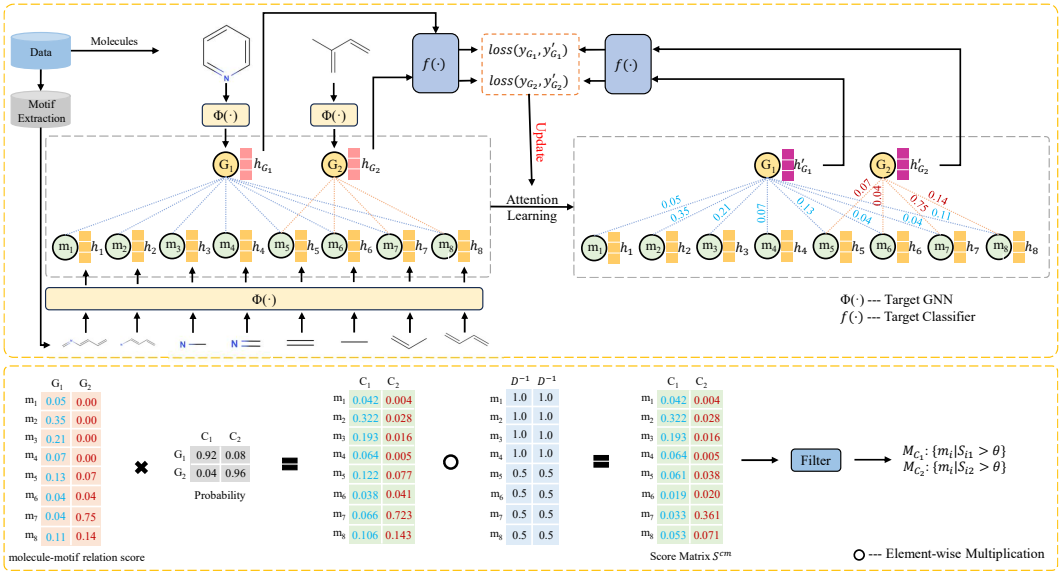


Figure 2: An illustration of the proposed MAGE framework. Given a dataset, a motif extraction algorithm is initially employed to identify all potential motifs. Each motif's feature encoding is derived from the output encoding produced by the target model, which uses the motif graph as its input. A single-layer attention operator is employed to learn the optimal motif combination, maximizing the likelihood that the target classifier will classify both the reconstructed and original molecular encodings identically. To get a score matrix, the method performs a dot product between the attention coefficient matrix and the prediction probability matrix. This score matrix is then normalized using a degree matrix. Finally, motifs whose corresponding scores exceed a specific threshold are selected.

Networks (GINs), they generally follow a message-passing mechanism. In this process, each GNN layer aggregates information from a node's neighbors to refine its representation. For a given hidden layer i, the message passing operation can be formulated as $h_i = f(h_{i-1}, \boldsymbol{A}, \boldsymbol{Z})$, where $h_0 = X$. Here, $h_i \in \mathbb{R}^{N \times D}$ is the node representation output from layer i, and D is the dimension of the output features. The function f involves computing messages based on the previous layer's node embeddings (h_{i-1}), aggregating these messages from neighboring nodes, and updating the hidden representation h_i for each node using the aggregated messages.

Motif. A motif is essentially a recurring subgraph within a graph, closely associated with the graph's property, which has been thoroughly researched across various domains, including biochemistry, ecology, neurobiology, and engineering (Milo et al., 2002; Shen-Orr et al., 2002; Alon, 2007; 2019) and has been demonstrated to be of significant importance. GSN (Bouritsas et al., 2022) introduces motif information into node features to improve the learning of GNNs. HMGNN (Yu & Gao, 2022a) generates a heterogeneous motif graph to enhance the molecular representation learning. MotifExplainer (Yu & Gao, 2022b) applies motifs to learn an instance-level explanation for GNNs.

3 Motif-based Model-Level GNN Explainer

This work introduces a Motif-bAsed GNN Explainer (MAGE), a novel approach for model-level GNN explanations. This method uses motifs as fundamental elements for interpreting the overarching behavior of a trained GNN for molecular graph classification at the model level. Instead of generating a graph at the atomic level, using motifs allows the generator to assemble graphs from these predefined blocks. This method prevents the creation of potentially invalid intermediate substructures and speeds up the generation process. It achieves this by reducing the number of configurations the generator must consider, focusing solely on the arrangement of motifs rather than the specific placement of each atom and bond.

Given a dataset, denoted as $\mathcal{G}$ with $|\mathcal{G}|$ molecules and C classes, our objective is to generate model-level explanations for a target GNN, which comprises a feature extractor $\phi(\cdot)$ and a classifier $f(\cdot)$.

$\phi(\cdot)$ takes a molecular graph or subgraph as input and outputs its feature representation. $f(\cdot)$ utilizes this representation to predict the classification of a molecule.

MAGE begins with identifying all potential motifs within $\mathcal{M}$. Following this, an attention-driven motif learning phase is employed to identify the most significant motifs for each class using $\phi(\cdot)$ and $f(\cdot)$. Finally, a graph generator uses the identified motifs to explain each class.

3.1 Motif Extraction

There are mainly three methods for motif extraction. It's important to note that these motif extraction methods can be integrated into our MAGE explanation method, demonstrating its versatility.

1) Ring and bonded pairs (R&B). Methods (Jin et al., 2018; Yu & Gao, 2022a; Bouritsas et al., 2022) in this category recognize simple rings and bonded pairs as motifs. 2) Molecule decomposition. In this category, methods recognize motifs through molecule breakdown. These methods initially establish a catalog of target bonds, either through domain-specific knowledge, such as the "breakable bonds" used by RECAP (Lewell et al., 1998; Zhang et al., 2021b) and BRICS (Degen et al., 2008; Zang et al., 2023; Jiang et al., 2023), or through hand-crafted rules, like "bridge bonds" (Jin et al., 2020). The algorithms remove these target bonds from the molecules and use the consequent molecular fragments as motifs. 3) Molecule tokenization. Certain methods (Fang et al., 2023; Kuenneth & Ramprasad, 2023) leverage the string representations of molecules, such as SMILES, and directly apply the WordPiece algorithm from the NLP domain. The resulting strings within the constructed vocabulary are regarded as motifs. 4) Data-driven method. MotifPiece (Yu & Gao, 2023) statistically identifies underlying structural or functional patterns specific to a given molecular data.

3.2 Class-wise Motif Identification

With extracted significant motifs from the previous section, this section introduces an attention-based motif identification approach for each classification class.

Stage 1: molecule-motif relation score calculation.
Given the absence of a direct linkage between classification categories and motifs, we propose bridging them through molecules. To this end, we employ an attention operator to learn motif-molecule relation scores using the trained feature extractor $\phi(\cdot)$ and a classifier $f(\cdot)$.

For a given molecule graph G_i and its associated motif set M_i, we use an attention operator to identify a combination of motifs that can accurately reconstruct the representations of the molecule graph. The learned attention scores between G_i and motifs in M_i are interpreted as their relation scores. The initial step involves obtaining feature encoding for the molecule and motif graphs. This is done by feeding a molecule graph or motif graph into $\phi(\cdot)$, and the output encoding serves as the molecule or motif:

$$\boldsymbol{h}_{G_i} = \phi(G_i), \boldsymbol{h}_{m_i} = \phi(m_i), \text{ for all } m_i \in M_i.$$

Next, an attention operator is used to aggregate the motif encoding using $\boldsymbol{h}_{G_i}$ as query and $\boldsymbol{h}_{m_i}$s as keys and values. The resulting encoding, denoted as $\boldsymbol{h}'_{G_i}$, is the aggregated molecular encoding by combining motifs in M_i.

$$e_{G_i m_i} = g(\boldsymbol{W}\boldsymbol{h}_{G_i}, \boldsymbol{W}\boldsymbol{h}_{m_i}), \alpha_{G_i m_i} = \frac{\exp(e_{G_i m_i})}{\sum_{m_i \in M_i} \exp(e_{G_i m_i})}, \boldsymbol{h}'_{G_i} = \sum_{m_i \in M_i} \alpha_{G_i m_i} \cdot \boldsymbol{h}_{m_i},$$

where $g(\cdot)$ is an attention operator, and $\boldsymbol{W}$ is weight matrix. The target classifier $f(\cdot)$ then uses aggregated encoding to generate predictions. The training process is designed to minimize the cross-entropy loss between the predicted probabilities of the aggregated molecular encoding and those of the original molecular encoding:

$$\mathcal{L} = \sum_i \mathcal{L}(f(\boldsymbol{h}_{G_i}), f(\boldsymbol{h}'_{G_i})). \tag{1}$$

The attention operator is trained to identify important motifs to represent the molecule. This leads to the molecule-motif relation score $\boldsymbol{\alpha}_{G_i m_i}$ and the probabilities associated with label predictions $\hat{\boldsymbol{y}} = f(\boldsymbol{h}'_{G_i})$.

Stage 2: class-motif relation score calculation.
We repeat the process of Stage 1 to compute the molecule-motif scores for each molecular graph. After that, we create a molecule-motif score matrix $\boldsymbol{S}^{mm} \in \mathbb{R}^{V \times |\mathcal{G}|}$ and a molecule-class probability matrix $\boldsymbol{P} \in \mathbb{R}^{|\mathcal{G}| \times C}$. Here, S^{mm}_{pk} represents the molecule-motif relation score between molecule k and motif p. If the motif i does not appear in the molecule j, we set $S^{mm}_{ij} = 0$. Element P_{kr} denotes the probability that molecule k is associated with label r.

Subsequently, we perform a matrix multiplication between $\boldsymbol{S}^{mm}$ and $\boldsymbol{P}$ to get class-motif relation score matrix $\boldsymbol{S}^{cm}$:

$$\boldsymbol{S}^{cm} = \boldsymbol{S}^{mm}\boldsymbol{P}, S^{cm}_{pr} = \sum_{k=1}^{|\mathcal{G}|} S^{mm}_{pk} P_{kr}$$

which represents the relationship between motif p and class r. Then, we normalize $\boldsymbol{S}^{cm}$ by dividing each column, corresponding to a motif, by the number of molecules it appears.

Stage 3: class-wise motif filtering.
We filter motifs for each class using class-wise motif scores from the previous stage. The motif set for the class C_r is

$$M_{C_r} = \{m_i | \boldsymbol{S}^{cm}_{ir} > \theta, 1 \leq i \leq V\} \tag{2}$$

where θ is a hyper-parameter to control the size of the important motif vocabulary. An illustration of our class-wise motif identification is provided in Figure 2.

3.3 Class-wise Motif-based Graph Generation

Section 3.2 identifies motifs of high relevance for each class. Building on this, our methodology employs a motif-based generative model to construct model-level explanations tailored to each class. The motif-based molecular generation method can be divided into tree-constrained (Jin et al., 2018; 2020) and non-constrained methods (Yang et al., 2021). In our approach, we employ a tree-constrained method as a generator. However, our approach can easily adapt to non-constrained methods. The graph generation process is illustrated in Figure 3.

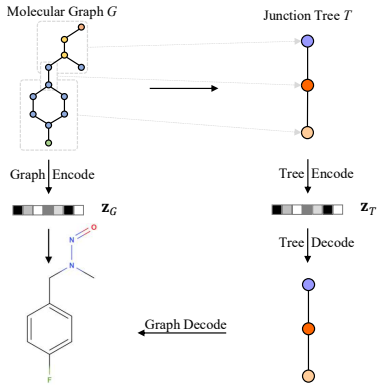


Figure 3: Illustration of class-wise motif-based graph generation. Starting with a molecular graph, we first construct a junction tree. Next, a tree encoder is applied to obtain a tree encoding, which is then decoded by a tree decoder to reconstruct the junction tree. Finally, a graph decoder uses the predicted junction tree to reproduce the molecular graph.

3.3.1 Tree Decomposition

Given a target class r, a set of important motifs M_{C_r} associated with the class r, and the set of molecules $\mathcal{G}_r$ whose predicted class by trained GNN is r. For each molecular graph $G_i \in \mathcal{G}_r$, we first identify the motifs in M_{C_r} from G_i: $M^r_i = M_i \cap M_{C_r}$. Subsequently, we pinpoint every edge in G_i that does not belong to any motif in M^r_i. We treat each motif in M^r_i and each non-motif edge as a cluster node. Then, we construct a cluster graph by connecting clusters with intersections. Finally, a spanning tree $\mathcal{T}$ is selected from this cluster graph, defined as the motif junction tree for G_i.

3.3.2 Graph Encoder

We encode the latent representation of $G = (\boldsymbol{A}, \boldsymbol{X})$ by the target GNN feature extractor $\phi(\cdot)$: $\boldsymbol{h}_G = \phi(\boldsymbol{A}, \boldsymbol{X})$. The mean $\boldsymbol{\mu}_G$ and log variance $\log \boldsymbol{\sigma}_G$ of the variational posterior approximation are computed from $\boldsymbol{h}_G$ with two separate affine layers. $\boldsymbol{z}_G$ is sampled from a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_G, \boldsymbol{\sigma}_G)$. In the variational posterior approximation, we use two different affine layers to compute the mean, represented as $\boldsymbol{\mu}_G$, and the log variance, denoted as $\log \boldsymbol{\sigma}_G$, from $\boldsymbol{h}_G$. Then, we sample $\boldsymbol{z}_G$ from a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_G, \boldsymbol{\sigma}_G)$.

3.3.3 TREE ENCODER

We employ a graph neural network to encode the motif tree $\mathcal{T} = (\boldsymbol{A}_{\mathcal{T}}, \boldsymbol{X}_{\mathcal{T}})$, where every node i's feature x_i is initially encoded by feeding its corresponding motif graph to $\phi(\cdot)$. Formally, the node embedding and tree embedding are updated as follows:

$$\boldsymbol{H}_{\mathcal{T}} = \text{GNN}(\boldsymbol{A}_{\mathcal{T}}, \boldsymbol{X}_{\mathcal{T}}), \boldsymbol{h}_{\mathcal{T}} = \text{AGG}(\boldsymbol{H}_{\mathcal{T}}).$$

where $\text{GNN}(\cdot)$ is a graph neural network, and $\text{AGG}(\cdot)$ is an aggregation function. $\boldsymbol{z}_{\mathcal{T}}$ is sampled similarly as in the graph encoder, where $\boldsymbol{z}_{\mathcal{T}}$ is a tree encoding.

3.3.4 TREE DECODER

A tree decoder is used to decode a junction tree $\mathcal{T}$ from a tree encoding $\boldsymbol{z}_{\mathcal{T}}$. The decoder builds the tree in a top-down approach, where nodes are created sequentially, one after the other. The node under consideration first obtains its embedding by inputting the existing tree into the tree encoder, and the resulting output is then used as the node embedding. The node embedding is utilized for two predictions: determining whether the node has a child and identifying the child's label if a child exists. Formally, for node i

$$\boldsymbol{H}' = \text{GNN}(\boldsymbol{A}', \boldsymbol{X}'), \boldsymbol{p}_i = \text{PRED}(\text{COMB}(\boldsymbol{z}_{\mathcal{T}}, \boldsymbol{H}'_i)), \boldsymbol{q}_i = \text{PRED}^l(\text{COMB}^l(\boldsymbol{z}_{\mathcal{T}}, \boldsymbol{H}'_i))$$

where $\boldsymbol{A}'$ and $\boldsymbol{X}'$ are the existing tree's adjacent matrix and node features, respectively. $\text{COMB}(\cdot)$ and $\text{COMB}^l(\cdot)$ are combination functions, and $\text{PRED}(\cdot)$ and $\text{PRED}^l(\cdot)$ are predictors. When the current node does not have a new child, the decoder will go to the next node in the existing tree.

3.3.5 GRAPH DECODER

The final step is reproducing the molecular graph from the predicted junction tree $\mathcal{T}$. Let $\mathcal{G}(\mathcal{T})$ be a set of graphs whose junction tree is $\mathcal{T}$. The decoder aims to find $\hat{G}$ as

$$\hat{G} = \arg \max_{G' \in \mathcal{G}(\mathcal{T})} f^a(G').$$

In our approach, we aim to ensure that the reproduced molecule maintains the characteristics of the target GNN. So we design the score function as $f^a(G_i) = f(\phi(G_i))[r]$, where r is the target label.

3.3.6 LOSS FUNCTION

The loss function for graph generation comprises two components: the reconstruction loss and the property loss. The reconstruction loss ensures that the generated graph matches the distribution of the original dataset. The property loss ensures that the behavior of the generated graph aligns with the target GNN.

The reconstruction loss can be formally defined as below:

$$\mathcal{L}_{\mathcal{R}} = \sum \mathcal{L}_{child} + \sum \mathcal{L}_{label} \tag{3}$$

, where $\mathcal{L}_{child}$ corresponds to predicting whether a node has a child, while $\mathcal{L}_{label}$ represents the loss associated with predicting the label of the newly added child. We perform teacher forcing for the reconstruction loss to make the information get the correct history at each step.

The property loss is like below:

$$\mathcal{L}_{\mathcal{P}} = \sum \text{MSE}(\boldsymbol{h}_G, \boldsymbol{h}_{\mathcal{T}}) + \sum \mathcal{L}(f(\boldsymbol{h}_{\mathcal{T}}), r) \tag{4}$$

, where $\boldsymbol{h}_{\mathcal{T}}$ is the embedding of the generated tree, $\mathcal{T}$ represents the predicted motif-node tree, $\boldsymbol{h}_G$ is the corresponding graph embedding, G refers to the generated molecular graph, $f(\cdot)$ is the target classifier, and r is the target label. The final loss function is

$$\mathcal{L} = \mathcal{L}_{\mathcal{R}} + \mathcal{L}_{\mathcal{P}} \tag{5}$$

Table 1: Validity results on six real-world datasets. OOM refers to out of memory.

Datasets	Models	**Label 0**	**Label 1**	**Average**
Mutagenicity	XGNN	0.34	0.25	0.295
	GNNInterpreter	0.11	0.21	0.16
	Ours	1.00	1.00	1.00
PTC_MR	XGNN	0.70	0.21	0.45
	GNNInterpreter	0.10	0.05	0.07
	Ours	1.00	1.00	1.00
PTC_MM	XGNN	0.49	0.56	0.51
	GNNInterpreter	0.15	0.20	0.17
	Ours	1.00	1.00	1.00
PTC_FM	XGNN	0.48	0.37	0.42
	GNNInterpreter	0.13	0.08	0.10
	Ours	1.00	1.00	1.00
AIDS	XGNN	0.92	0.71	0.81
	GNNInterpreter	0.09	0.08	0.08
	Ours	1.00	1.00	1.00
NCI-H23	XGNN	OOM	OOM	OOM
	GNNInterpreter	0.59	0.62	0.60
	Ours	1.00	1.00	1.00

4 EXPERIMENT

We conduct qualitative and quantitative experiments on six real-world datasets to evaluate the effectiveness of our proposed methods.

4.1 DATASET AND EXPERIMENTAL SETUP

We evaluate the proposed methods using molecule classification tasks on six real-world datasets: Mutagenicity, PTC_MR, PTC_MM, PTC_FM, AIDS, and NCI-H23. The details of six datasets and experimental settings are represented in Appendix B and C.

Baselines. We compare our MAGE model with two state-of-the-art baselines: XGNN and GNNInterpreter. Noted that all methods are tested using official implementation and compared in a fair setting. We extract motifs by identifying "bridge bonds". We build a variant that uses the same motif extraction and graph generation processes but without our attention-based motif identification step. This variant is to demonstrate the effectiveness of our class-wise motif identification process. The details of this variant will be discussed in section 4.2.

Evaluation metrics. Our quantitative evaluation uses two distinct metrics to assess performance: Validity and Average Probability.

Validity. This metric is defined as the proportion of chemically valid molecules out of the total number of generated molecules (Bilodeau et al., 2022).

$$\text{Validity} = \frac{\#\,\text{valid molecules}}{\#\,\text{generated molecules}}$$

This metric serves as a critical indicator of the practical applicability of our method, ensuring that the generated molecules not only align with the intended class but also adhere to fundamental chemical validity criteria.

Average Probability. Following the GNNInterpreter (Wang & Shen, 2022), this metric calculates the average class probability and the standard deviation of these probabilities across 1000 explanation graphs for each class within all six datasets.

$$\text{Average Probability} = \frac{\sum \text{class probability}}{\#\,\text{generated molecules}}.$$

This approach provides a comprehensive measure of the consistency and reliability of the explanation graphs in representing different classes.

Table 2: Quantitative results for different explanation methods. We highlight the average class probability (higher is better) for six datasets. The best performances on each dataset are shown in **bold**. OOM refers to out of memory.

Datasets	Models	**Label 0**	**Label 1**	**Average**
Mutagenicity	XGNN	0.8992 ± 0.0835	0.9831 ± 0.0514	0.9411
	GNNInterpreter	0.8542 ± 0.3198	0.9938 ± 0.0191	0.9240
	Variant	0.9897 ± 0.0754	0.9888 ± 0.0302	0.9892
	Ours	$\mathbf{0.9977 \pm 0.0032}$	$\mathbf{0.9941 \pm 0.0240}$	**0.9959**
PTC_MR	XGNN	0.9906 ± 0.0718	0.9698 ± 0.0417	0.9802
	GNNInterpreter	0.9067 ± 0.1728	0.9697 ± 0.0211	0.9382
	Variant	0.9902 ± 0.0480	0.9641 ± 0.1163	0.9771
	Ours	$\mathbf{0.9961 \pm 0.0375}$	$\mathbf{0.9918 \pm 0.0621}$	**0.9939**
PTC_MM	XGNN	0.9899 ± 0.0994	0.9266 ± 0.1059	0.9582
	GNNInterpreter	0.9601 ± 0.0638	0.9541 ± 0.0207	0.9571
	Variant	0.9784 ± 0.0475	0.9693 ± 0.0279	0.9738
	Ours	$\mathbf{0.9914 \pm 0.0342}$	$\mathbf{0.9833 \pm 0.0019}$	**0.9873**
PTC_FM	XGNN	0.9967 ± 0.0309	0.9380 ± 0.0991	0.9673
	GNNInterpreter	0.9945 ± 0.0147	0.9460 ± 0.0295	0.9702
	Variant	0.9882 ± 0.0024	0.9790 ± 0.0024	0.9836
	Ours	$\mathbf{0.9979 \pm 0.0024}$	$\mathbf{0.9890 \pm 0.0024}$	**0.9934**
AIDS	XGNN	0.9259 ± 0.1861	$\mathbf{0.9977 \pm 0.0225}$	0.9618
	GNNInterpreter	0.4600 ± 0.4983	0.9973 ± 0.0116	0.7286
	Variant	0.9802 ± 0.1112	0.9939 ± 0.0564	0.9870
	Ours	$\mathbf{0.9883 \pm 0.0663}$	0.9903 ± 0.0539	**0.9893**
NCI-H23	XGNN	OOM	OOM	OOM
	GNNInterpreter	0.9883 ± 0.0711	$\mathbf{0.9997 \pm 0.0015}$	**0.9940**
	Variant	0.9874 ± 0.0685	0.9863 ± 0.01069	0.9868
	Ours	$\mathbf{0.9936 \pm 0.0329}$	0.9934 ± 0.0422	0.9935

Together, these two metrics offer a robust framework for evaluating the effectiveness of our approach from both the accuracy and applicability perspectives. Considering XGNN's limitation in generating explanation graphs with edge features, we assign the default bond type from the RDKit library as the default edge feature for explanations produced by XGNN. We also report the training time and sampling time of different models in appendix E.

4.2 Quantitative Results

Table 1 details the Validity results for various models across six datasets. A notable observation from these results is that our method consistently produces valid molecular explanation graphs for both classes in all cases. This reliability stems from our method's use of valid substructures as foundational elements, effectively overcoming the limitation seen in atom-by-atom generation methods, which often compel the model to create chemically invalid intermediate structures.

Additionally, the data reveals that XGNN exhibits better validity than GNNInterpreter. The reason for this difference lies in the design of the algorithms: XGNN incorporates a manual constraint on the maximum atom degree, which aids in maintaining chemical validity. In contrast, GNNInterpreter is to align the embedding of its generated explanations with the average embedding of the dataset, which does not inherently ensure chemical validity and leads to more invalid generated molecules.

These findings from the validity experiment highlight a principle: maximizing the inductive biases inherent in the molecular graph structure is important. It enables models to achieve higher levels of validity in the outputs, demonstrating that careful consideration of the structural characteristics of molecules can significantly enhance the performance and reliability of these models.

Table 2 shows the average probability of four models on six real-world datasets. Our method outperforms the baseline methods in ten out of twelve experiments, clearly highlighting its effectiveness. Additionally, our method consistently delivers strong results across both classes in each dataset. This contrasts XGNN and GNNInterpreter, which tend to excel in one class but fall short in the

Table 3: The qualitative results for Mutagenicity dataset. The first row of the table displays the explanation graphs corresponding to the Nonmutagen label, while the second row presents the explanation graphs for the Mutagen label. The final column in the table provides examples of actual graphs. The different colors in the nodes represent different values in the node feature

Class	Generated Model-Level Explanation Graphs for Mutagenicity Dataset			
	XGNN	GNNInterpreter	Ours	Example
Nonmutagen				
Mutagen				

other. We attribute this success of our method to its unique strategy of generating explanations in a motif-by-motif way, effectively reducing the influence of extraneous or noise atoms in the data.

Furthermore, we conducted a comparative analysis between our method and a variant model. This variant model operates by first identifying all potential motifs within the dataset. Once these motifs are extracted, it assesses them by putting each motif as an input to the target model. The evaluation of each motif is based on the output probability associated with a particular class, which is assigned as the motif's score. Following this scoring process, the method ranks all motifs in order of their scores. Only motifs whose scores exceed 0.9 are chosen to refine the selection.

The outcomes of our study indicate that our method outperforms the variant model across all classes. This significant improvement underscores the efficiency of our attention-based motif learning module. Unlike the variant model, which solely focuses on the individual motifs without considering their functional roles within the molecules, our method learns and understands the interplay between motifs and molecules. This enables our approach to effectively identify and select motifs relevant to the specific molecular context while simultaneously filtering out motifs that do not contribute meaningfully, often called noise motifs.

4.3 Qualitative Results

Table 3 presents the qualitative assessment of three different methods applied to the two classes of the Mutagenicity dataset. The table also includes actual graph examples from the dataset for reference. Our analysis reveals that our method can consistently produce human-understandable and valid explanation graphs, which XGNN and GNNInterpreter seem to lack. More qualitative results can be found in Appendix G.

We observed a trend in GNNInterpreter to produce explanations that are disconnected. This is likely due to its approach of aligning explanation embeddings with the dataset's average embedding, resulting in a focus on node features rather than structural details. XGNN, on the other hand, tends to create explanations with invalid ring structures. Although XGNN imposes constraints on the degree of each atom, its atom-by-atom generation approach makes it challenging to form valid ring structures. Our method can generate explanations with meaningful substructures like rings.

This inability of XGNN and GNNInterpreter to effectively identify complex molecular structures may contribute to their lower validity in explanation generation. Our method, in contrast, successfully navigates these complexities, offering more meaningful, accurate and structurally coherent explanations. This difference highlights the importance of considering both node features and molecular structures in the generation of explanation graphs, a balance that our approach seems to strike effectively.

4.4 Ablation Study: Loss Fuction

In this section, we evaluate performance when using each loss component independently. Table 4 shows the results. From the table, we observe that using only the reconstruction loss results in high accuracy, though it is slightly lower than configurations that include property alignment. This indicates that, while the model is proficient at preserving input structure, it may lack alignment with target-specific

Table 4: Comparison of different components of the loss function.

Loss function	**Label 0**	**Label 1**
$\mathcal{L}_R$	0.9827 ± 0.0580	0.9892 ± 0.0482
$\mathcal{L}_P$	0.9881 ± 0.0518	0.9805 ± 0.0589
$\mathcal{L}_R + \mathcal{L}_P$	0.9977 ± 0.0032	0.9941 ± 0.0240

characteristics. The table also shows that using only the property loss results in slightly lower performance than the combined loss. This suggests that, without the reconstruction loss, the model may lose some structural accuracy, leading to a minor decrease in overall performance.

5 Related Work

Model-level explanation methods in the field are currently under-researched, with only a few studies addressing this issue. These methods can be divided into two categories. The first category is concept-based methods. GCExplainer (Magister et al., 2021) introduced the incorporation of concepts into GNN explanations, using the k-Means clustering algorithm on GNN embeddings to identify clusters, each representing a concept. PAGE (Shin et al., 2022) provides clustering on the embedding space to discover propotypes as explanations. GLGExplainer (Azzolin et al., 2022) adopts prototype learning to identify data prototypes and then uses these in an E-LEN model to create a Boolean formula that replicates the GNN's behavior. GCNeuron (Xuanyuan et al., 2023), on the other hand, employs human-defined rule in natural language and considers graphs with certain masked nodes as concepts, using compositional concepts with the highest scores for global explanations. The second category comprises generation-based methods. XGNN (Yuan et al., 2020) uses deep reinforcement learning to generate explanation graphs node by node. GNNInterpreter (Wang & Shen, 2022), alternatively, learns a probabilistic model that identifies the most discriminative graph patterns for explanations.

6 Conclusion

This work proposes a novel motif-based model-level explanation method for Graph Neural Networks, specifically applied to molecular datasets. Initially, the method employs a motif extraction technique to identify all potential motifs. Following this, a one-layer attention mechanism is used to evaluate the relevance of each motif, filtering out those that are not pertinent. Finally, a graph generator is trained to produce explanations based on the chosen motifs. Compared to two baseline methods, XGNN and GNNInterpreter, our proposed method offers explanations that are more human-understandable, accurate, and structurally consistent.

References

Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6): 450–461, 2007.

Uri Alon. *An introduction to systems biology: design principles of biological circuits*. CRC press, 2019.

Steve Azzolin, Antonio Longa, Pietro Barbiero, Pietro Liò, and Andrea Passerini. Global explainability of gnns via logic combination of learned concepts. *arXiv preprint arXiv:2210.07147*, 2022.

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

Camille Bilodeau, Wengong Jin, Tommi Jaakkola, Regina Barzilay, and Klavs F Jensen. Generative models for molecular discovery: Recent advances and challenges. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 12(5):e1608, 2022.

Pietro Bongini, Monica Bianchini, and Franco Scarselli. Molecular generative graph neural networks for drug discovery. *Neurocomputing*, 450:242–252, 2021.

Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):657–668, 2022.

Jörg Degen, Christof Wegscheid-Gerlach, Andrea Zaliani, and Matthias Rarey. On the art of compiling and using'drug-like'chemical fragment spaces. *ChemMedChem: Chemistry Enabling Drug Discovery*, 3(10):1503–1507, 2008.

Lei Fang, Junren Li, Ming Zhao, Li Tan, and Jian-Guang Lou. Single-step retrosynthesis prediction by leveraging commonly preserved substructures. *Nature Communications*, 14(1):2446, 2023.

Xiaomin Fang, Lihang Liu, Jieqiong Lei, Donglong He, Shanzhuo Zhang, Jingbo Zhou, Fan Wang, Hua Wu, and Haifeng Wang. Geometry-enhanced molecular representation learning for property prediction. *Nature Machine Intelligence*, 4(2):127–134, 2022.

Thorben Funke, Megha Khosla, and Avishek Anand. Hard masking for explaining graph neural networks. 2020.

Hongyang Gao and Shuiwang Ji. Graph u-nets. In *international conference on machine learning*, pp. 2083–2092. PMLR, 2019.

Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1416–1424, 2018.

Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216*, 2020.

Yinghui Jiang, Shuting Jin, Xurui Jin, Xianglu Xiao, Wenfan Wu, Xiangrong Liu, Qiang Zhang, Xiangxiang Zeng, Guang Yang, and Zhangming Niu. Pharmacophoric-constrained heterogeneous graph transformer model for molecular property prediction. *Communications Chemistry*, 6(1):60, 2023.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Hierarchical generation of molecular graphs using structural motifs. In *International conference on machine learning*, pp. 4839–4848. PMLR, 2020.

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

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

Christopher Kuenneth and Rampi Ramprasad. polybert: a chemical language model to enable fully machine-driven ultrafast polymer informatics. *Nature Communications*, 14(1):4099, 2023.

Xin Lai, Peisong Yang, Kunfeng Wang, Qingyuan Yang, and Duli Yu. Mgrnn: Structure generation of molecules based on graph recurrent neural networks. *Molecular informatics*, 40(10):2100091, 2021.

Xiao Qing Lewell, Duncan B Judd, Stephen P Watson, and Michael M Hann. Recap retrosynthetic combinatorial analysis procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry. *Journal of chemical information and computer sciences*, 38(3):511–522, 1998.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *arXiv preprint arXiv:2011.04573*, 2020.

Lucie Charlotte Magister, Dmitry Kazhdan, Vikash Singh, and Pietro Liò. Gcexplainer: Human-in-the-loop concept-based explanations for graph neural networks. *arXiv preprint arXiv:2107.11889*, 2021.

Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

Phillip E Pope, Soheil Kolouri, Mohammad Rostami, Charles E Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10772–10781, 2019.

Kaspar Riesen and Horst Bunke. Iam graph database repository for graph based pattern recognition and machine learning. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pp. 287–297. Springer, 2008.

Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 33:12559–12571, 2020.

Thomas Schnake, Oliver Eberle, Jonas Lederer, Shinichi Nakajima, Kristof T Schütt, Klaus-Robert Müller, and Grégoire Montavon. Higher-order explanations of graph neural networks via relevant walks. *arXiv preprint arXiv:2006.03589*, 2020.

Caihua Shan, Yifei Shen, Yao Zhang, Xiang Li, and Dongsheng Li. Reinforcement learning enhanced explainer for graph neural networks. *Advances in Neural Information Processing Systems*, 34: 22523–22533, 2021.

Shai S Shen-Orr, Ron Milo, Shmoolik Mangan, and Uri Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature genetics*, 31(1):64–68, 2002.

Yong-Min Shin, Sun-Woo Kim, and Won-Yong Shin. Page: Prototype-based model-level explanations for graph neural networks. *arXiv preprint arXiv:2210.17159*, 2022.

Ruoxi Sun, Hanjun Dai, and Adams Wei Yu. Does gnn pretraining help molecular representation? *Advances in Neural Information Processing Systems*, 35:12096–12109, 2022.

Minh N Vu and My T Thai. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. *arXiv preprint arXiv:2010.05788*, 2020.

Xiang Wang, Yingxin Wu, An Zhang, Xiangnan He, and Tat-Seng Chua. Towards multi-grained explainability for graph neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

Xiang Wang, Yingxin Wu, An Zhang, Fuli Feng, Xiangnan He, and Tat-Seng Chua. Reinforced causal explainer for graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Xiaoqi Wang and Han-Wei Shen. Gnninterpreter: A probabilistic generative model-level explanation for graph neural networks. *arXiv preprint arXiv:2209.07924*, 2022.

Yuyang Wang, Zijie Li, and Amir Barati Farimani. Graph neural networks for molecules. In *Machine Learning in Molecular Sciences*, pp. 21–66. Springer, 2023.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Han Xuanyuan, Pietro Barbiero, Dobrik Georgiev, Lucie Charlotte Magister, and Pietro Liò. Global concept-based interpretability for graph neural networks via neuron analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 10675–10683, 2023.

Soojung Yang, Doyeong Hwang, Seul Lee, Seongok Ryu, and Sung Ju Hwang. Hit and lead discovery with explorative rl and fragment-based molecule generation. *Advances in Neural Information Processing Systems*, 34:7924–7936, 2021.

Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32:9240, 2019.

Zhaoning Yu and Hongyang Gao. Molecular representation learning via heterogeneous motif graph neural networks. In *International Conference on Machine Learning*, pp. 25581–25594. PMLR, 2022a.

Zhaoning Yu and Hongyang Gao. Motifexplainer: a motif-based graph neural network explainer. *arXiv preprint arXiv:2202.00519*, 2022b.

Zhaoning Yu and Hongyang Gao. Motifpiece: A data-driven approach for effective motif extraction and molecular representation learning. *arXiv preprint arXiv:2312.15387*, 2023.

Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. 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.

Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. In *International conference on machine learning*, pp. 12241–12252. PMLR, 2021.

Xuan Zang, Xianbing Zhao, and Buzhou Tang. Hierarchical molecular graph self-supervised learning for property prediction. *Communications Chemistry*, 6(1):34, 2023.

Yue Zhang, David Defazio, and Arti Ramesh. Relex: A model-agnostic relational model explainer. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 1042–1049, 2021a.

Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Chee-Kong Lee. Motif-based graph self-supervised learning for molecular property prediction. *Advances in Neural Information Processing Systems*, 34:15870–15882, 2021b.

A DETAILS OF DECODE A VALID MOLECULE EXPLANATION

To decode a valid molecule explanation from a tree structure, we have the following steps:

1. Start with a decoded Tree: The decoding process begins with a decoded tree $\mathcal{T}$, which represents the hierarchical structure of motifs.
2. Enumerate motif Combinations: For each motif node in the tree and its neighboring motifs, we enumerate different combinations to find feasible arrangements. Certain combinations lead to chemically infeasible molecules, and are discarded from further consideration.
3. Rank and Combine combined subgraphs: At each node, we rank the combined subgraphs based on our target GNN, ensuring that the final graph is following the same distribution of our target class.
4. The final graph is decoded by putting together all the predicted subgraphs.

Algorithm 1 Decoding a Molecule from a Tree Structure

Require: Decoded tree $\mathcal{T}$ representing the hierarchical structure of motifs.
Ensure: Final molecular graph G.
1: **Initialization**: Set the final graph G to an empty graph.
2: **for** each motif node m in the tree $\mathcal{T}$ **do**
3: Identify neighboring motifs of m in $\mathcal{T}$.
4: Enumerate all feasible combinations of m with its neighboring motifs.
5: Discard combinations that lead to chemically infeasible molecules.
6: Rank the remaining combined subgraphs using the target GNN model to assess alignment with the target class distribution.
7: Select the top-ranked combined subgraph.
8: Integrate the selected subgraph into the final graph G.
9: **end for**
10: **Output**: The final molecular graph G obtained by assembling all predicted subgraphs.

Algorithm shows the details of our graph decoder.

B DETAILS OF DATASETS

Table 5: Statistics and properties of four real-world molecule datasets.

	Mutag	**PTC_MR**	**PTC_MM**	**PTC_FM**	**AIDS**	**NCI-H23**
# Nodes (avg)	30.32	14.29	14.32	14.48	15.69	26.07
# Edges (avg)	30.77	14.69	13.97	14.11	16.20	28.10
# Graphs	4337	344	336	349	2000	40353
# Classes	2	2	2	2	2	2

The statistics and properties of four datasets are summarized in Table 5.

Mutagenicity (Kazius et al., 2005; Riesen & Bunke, 2008) is a chemical compound dataset containing 4,337 molecule graphs. Each of these graphs is classified into either mutagen or non-mutagen categories, signifying their mutagenic impact on the Gram-negative bacterium Salmonella typhimurium.

PTC_MR, PTC_MM, and PTC_FM (Morris et al., 2020) are three collections of chemical compounds, each respectively documenting carcinogenic effects on male rats (MR), male mice (MM), and female mice (FM).

AIDS (Morris et al., 2020) comprises 2,000 graphs that represent molecular compounds, all of which are derived from the AIDS Antiviral Screen Database of Active Compounds.

NCI-H23 (Morris et al., 2020) is a collection of 40353 molecular compounds to evaluate the efficacy against lung cancer cells.

C EXPERIMENTAL SETTINGS

By following baseline approaches, our experiments adopt a simple GCN model and focus on explanation results. For the target GNN, we use a 3-layer GCN as a feature extractor and a 2-layer MLP as a classifier on all datasets. The model is pre-trained and achieves reasonable performances on all datasets. Following the (Xu et al., 2018), the hidden dimension is selected from $\{16, 64\}$. We employ mean-pooling as the readout function and ReLU as the activation function. We use Adam optimizer for training. The target model is trained for 100 epochs, and the learning rate is set to 0.01. The CPU in our setup is an AMD Ryzen Threadripper 2990WX, accompanied by 256 GB of memory, and the GPU is an RTX 4090. For training an explainer, the number of epochs is selected from {50, 100}, the learning rate is selected from {0.01, 0.0001, 0.0005}, the batch size is selected from {32, 320}.

D STUDY OF MOTIF EXTRACTION TIME

In this section, we study the motif extraction time for all six datasets.

	Average Number of Nodes	Motif Extraction Time (ms/graph)
Mutagenicity	30.23	0.799
PTC_MR	14.29	0.428
PTC_MM	13.97	0.415
PTC_FM	14.11	0.444
AIDS	15.69	0.457
NCI-H23	26.07	0.877

The table demonstrates that the bridge bond cut-off method achieves decomposition times in the millisecond range. This approach allows us to process large datasets with minimal computational cost.

E STUDY OF TRAINING AND INFERENCE TIME

We also present a detailed comparison of the training and sampling times between our method and other baseline approaches in table 6. Here, sampling time is the average time to generate a single explanation. Compared to the two baselines, a key distinction of our method lies in its requirement for an initial training phase for the explanation graph generator. This phase confers a significant benefit: the ability to efficiently sample explanations. Our experimental data underscores this advantage. Following the completion of the training phase for the explanation generator, our method demonstrated the capacity to generate a large number of explanations rapidly. Specifically, it could sample 1000 explanations in a matter of seconds. This starkly contrasts other baseline methods, which required at least 3.3 hours to produce the same number of examples. This dramatic reduction in sampling time not only highlights the efficiency of our method but also suggests its practical applicability in scenarios where quick generation of explanations is crucial. Due to the space limit, we put the study of hyper-parameter θ in Appendix F.

F STUDY OF HYPERPARAMETER

Table 7 shows the study of hyperparameter θ, we found that as θ increases, the number of motifs selected for the Mutagen class decreases. The average probability rises until θ reaches 0.10, where peak performance is observed, after which it starts to decline. This trend occurs because when θ is low, some irrelevant motifs are included in the explanations. As θ increases, these 'noise' motifs

Table 6: Training and inference time for different explanation methods. XGNN and GNNInterpreter do not have training time because they only have sampling process.

Datasets	Models	**Training Time**	**Sampling Time**
Mutagenicity	XGNN	—	54s
	GNNInterpreter	—	29s
	Variant	528s	0.0015s
	Ours	539s	0.0029s
PTC_MR	XGNN	—	62s
	GNNInterpreter	—	27s
	Variant	338s	0.0005s
	Ours	368s	0.0006s
PTC_MM	XGNN	—	46s
	GNNInterpreter	—	17s
	Variant	303s	0.0023s
	Ours	367s	0.0002s
PTC_FM	XGNN	—	47s
	GNNInterpreter	—	12s
	Variant	307s	0.0027s
	Ours	386s	0.0018s
AIDS	XGNN	—	19s
	GNNInterpreter	—	32s
	Variant	468s	0.0065s
	Ours	513s	0.0076s
NCI-H23	XGNN	—	OOM
	GNNInterpreter	—	21s
	Variant	10189s	0.0115s
	Ours	13657s	0.0126s

Table 7: Study of Hyperparameter θ on Mutagen class of Mutagenicity dataset

	$\theta = \mathbf{0.01}$	$\theta = \mathbf{0.05}$	$\theta = \mathbf{0.10}$	$\theta = \mathbf{0.20}$
# Selected Motifs	1591	1328	1206	1073
Average Probability	0.9621 ± 0.1151	0.9850 ± 0.0784	0.9977 ± 0.0032	0.9753 ± 0.0981

are filtered out, enhancing performance. However, if θ is set too high, the resulting small number of selected motifs is insufficient for forming a comprehensive explanation.

G MORE QUALITATIVE RESULTS

Table 8: More qualitative results for Nonmutagen class on Mutagenicity dataset.

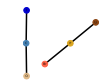
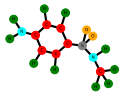
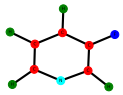
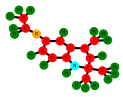
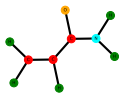
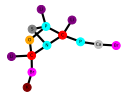
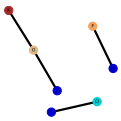
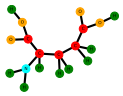
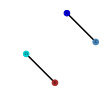
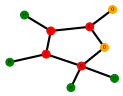
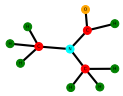
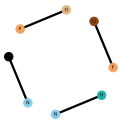
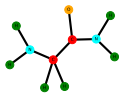
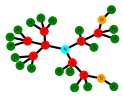
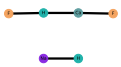
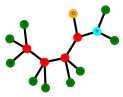
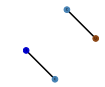
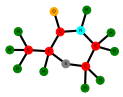
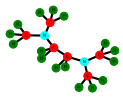
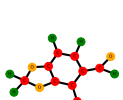
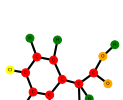
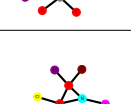
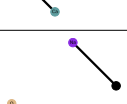
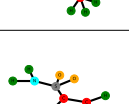
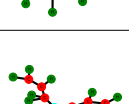
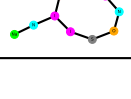
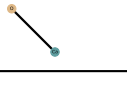
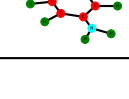
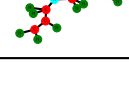
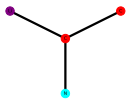
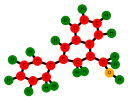
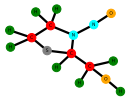
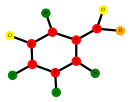
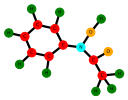
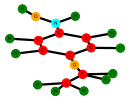
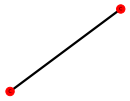
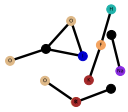
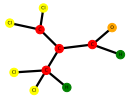
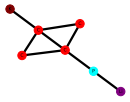
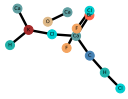
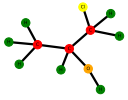
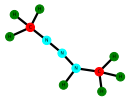
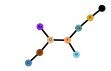
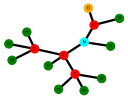
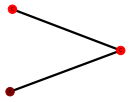
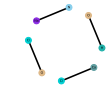
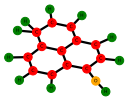
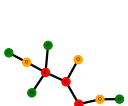
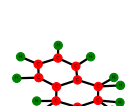
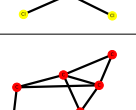
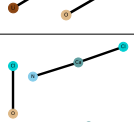
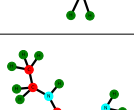
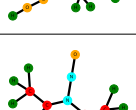
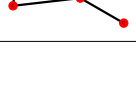
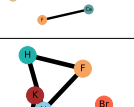
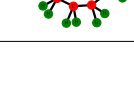
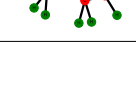
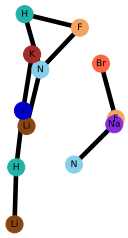
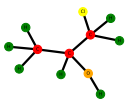
Class	XGNN	GNNInterpreter	Ours	Example
Nonmutagen				
				
				
				
				
				
				
				
				
				
				

Table 9: More qualitative results for Mutagen class on Mutagenicity dataset.

Class	XGNN	GNNInterpreter	Ours	Example
Mutagen