Higher Order Structures For Graph Explanations

Akshit Sinha*, Sreeram Vennam*, Charu Sharma, Ponnurangam Kumaraguru

International Institute of Information Technology, Hyderabad
{akshit.sinha, sreeram.vennam}@students.iiit.ac.in

Abstract

Graph Neural Networks (GNNs) have emerged as powerful tools for learning representations of graph-structured data, demonstrating remarkable performance across various tasks. Recognizing their importance, there has been extensive research focused on explaining GNN predictions, aiming to enhance their interpretability and trustworthiness. However, GNNs and their explainers face a notable challenge: graphs are primarily designed to model pair-wise relationships between nodes, which can make it tough to capture higher-order, multi-node interactions. This characteristic can pose difficulties for existing explainers in fully representing multi-node relationships. To address this gap, we present **F**ramework For Higher-**O**rder **R**epresentations In **G**raph **E**xplanations (FORGE), a framework that enables graph explainers to capture such interactions by incorporating higher-order structures, resulting in more accurate and faithful explanations. Extensive evaluation shows that on average real-world datasets from the GraphXAI benchmark and synthetic datasets across various graph explainers, FORGE improves average explanation accuracy by 1.9x and 2.25x, respectively. We perform ablation studies to confirm the importance of higher-order relations in improving explanations, while our scalability analysis demonstrates FORGE's efficacy on large graphs.

1 Introduction

Graph Neural Networks (GNNs) (Kipf and Welling 2017) have become increasingly important in graph representation learning, as data in many real-world domains can be naturally modeled as graphs. GNNs have found applications in several sensitive fields, including information processing (Ying et al. 2018; Wang et al. 2019), criminal justice (Agarwal, Lakkaraju, and Zitnik 2021), molecular chemistry (Gilmer et al. 2017; Duvenaud et al. 2015), and bioinformatics (Zhang et al. 2021; Fout et al. 2017; Ragkousis 2022). In these sensitive domains, interpretability is crucial for ensuring transparent, justifiable, and ethical decision-making. As GNN usage expands, understanding their internal processes becomes vital for the effective and safe usage of these models in practical settings. To address these challenges, various graph explainers have been proposed. These graph explainers typically provide two types of explanations crucial for GNN prediction: (1) identification of subgraphs (Yuan et al. 2021; Schlichtkrull, Cao, and Titov 2022), and (2) determination of node features (Huang et al. 2020; Ying et al. 2019).

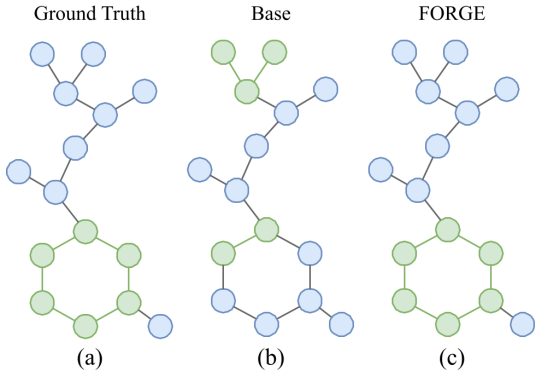

Figure 1: (a) Ground Truth for an example from BENZENE (Agarwal et al. 2023) (b) Explanation generated by GNNExplainer (Ying et al. 2019) (c) Explanation generated by using FORGE. **Green** nodes and edges signify the subgraph considered important for GNN prediction. By incorporating FORGE, we can capture important multi-node interactions, resulting in more accurate explanations.

However, graph explainers and GNNs face an inherent limitation in their ability to generate representations stemming from graphs' ability to model only pair-wise interactions between nodes. This limitation has implications on the expressivity of GNNs, which is explored in detail by Xu et al. (2019). To address this, higher-order structures such as *cell complexes* (Hansen and Ghrist 2019) have been employed as they are a higher-order generalization of graphs, capable of modeling multi-node interactions. Neural networks designed for these higher-order structures (Bodnar et al. 2021; Ebli, Defferrard, and Spreemann 2020; Yang, Sala, and Bogdan 2022) have demonstrated significant performance improvements in graph learning tasks compared to traditional graph structures, and have been shown to be more expressive than traditional GNNs. Despite these advancements in higher-order graph representations, there exists a

*These authors contributed equally.

critical gap in the field of graph explainability: the potential of higher-order structures to enhance the interpretability of graph-based models remains unexplored.

In this work, we present **F**ramework For Higher-**O**rder **R**epresentations In **G**raph **E**xplanations (FORGE), a novel, explainer-agnostic framework designed to enhance the capability of graph explainers by capturing multi-node interactions during the GNN learning process itself by internally representing the underlying graph data as a cell complex. To further refine the quality of explanations, as part of our framework we introduce Information Propagation algorithms that translate the explanations generated for cell complexes back to the underlying graph data, resulting in richer, more accurate, and faithful explanations with respect to the input data as well as the underlying GNN predictor. This is encapsulated by Figure 1, where our framework can capture the multi-node interactions of a benzene ring, which is the correct ground truth explanation. Without FORGE, the base explainer is unable to interpret these interactions, resulting in less accurate explanations.

The overall framework is described in Figure 2. We conduct extensive evaluations of FORGE on real-world datasets from the GraphXAI (Agarwal et al. 2023) benchmark, as well as on specially curated synthetic datasets. Our results demonstrate that incorporating FORGE consistently matches or improves the explanation accuracy and faithfulness of various graph explainers. To reaffirm our hypothesis, we perform rigorous ablations and find that each components of our framework contribute significantly to increased explainer performance. Additionally, our analysis of scalability reveals that FORGE efficiently handles dense, complex graph networks with only linear overhead in both time and space complexity, further emphasizing its practical applicability and computational efficiency.

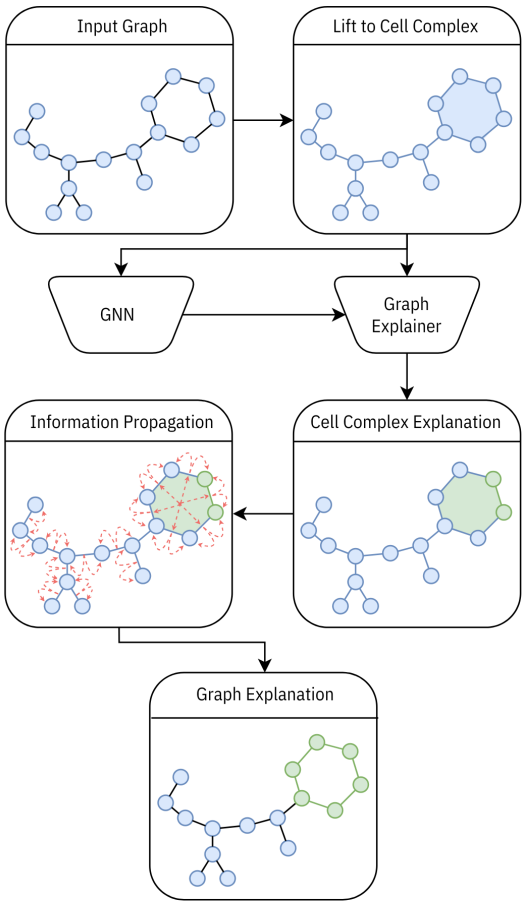

Figure 2: Visual representation of FORGE. The input graph is lifted to a cell complex, which is then given as input to (i) a GNN to train on, as well as to (ii) a graph explainer. Propagation is then done on the output cell complex explanation to map it to an explanation for the original graph. The green color on cells, nodes, and edges signifies the substructure considered important for GNN prediction (the explanation).

2 Related Work

Higher-Order Representations of Graphs Hansen and Ghrist (2019) first explored relating higher-order structures to spectral graph theory. Following this, recent advancements in GNN architecture have explored the incorporation of higher-order topological structures, such as simplicial complexes (Ebli, Defferrard, and Spreemann 2020) and cell complexes (Bodnar et al. 2021), to successfully enhance the expressive power and representational capacity of these models. This line of work is motivated by the inherent limitations of traditional GNNs, particularly their constraints in capturing complex structural patterns (Xu et al. 2019) and effectively modeling long-range dependencies (Chen, Schulz, and Borgwardt 2024).

Explainability in GNNs The increasing adoption of GNNs in critical domains has resulted in significant research into methods for explaining their predictions. A comprehensive survey by Kakkad et al. (2023) provides an extensive overview of GNN explainability techniques.

Perturbation-based methods, such as GNNExplainer (Ying et al. 2019), PGExplainer (Luo et al. 2020), SubgraphX (Yuan et al. 2021), and GraphMask (Schlichtkrull, Cao, and Titov 2022), identify important subgraphs by systematically perturbing input graphs and analyzing the impact on model outputs. Gradient-based approaches, including Grad-CAM (Selvaraju et al. 2019) and Guided Backpropagation (Springenberg et al. 2015), leverage gradient information to assess the influence of input features on model predictions. Surrogate methods, exemplified by PGM-Explainer (Vu and Thai 2020) and GraphLIME (Huang et al. 2020), approximate complex GNN behavior with interpretable local models. Recent innovations have further expanded the landscape of GNN explainability. RG-Explainer (Shan et al. 2021) employs reinforcement learning to construct explanatory subgraphs tailored to both model architecture and individual instances. MATE (Spinelli, Scardapane, and Uncini 2024) introduces a meta-learning framework that jointly optimizes GNN performance and explainability, resulting in intrinsically more interpretable representations. MotifExplainer (Yu and Gao 2022) introduces a motif-based approach for human interpretable explanations.

While MotifExplainer is most similar to our work, it is domain specific and suffers from $O(n^3)$ complexity.

Although these approaches have significantly advanced GNN explainability, they primarily operate on traditional graph structures. Our proposed framework, FORGE, distinguishes itself by leveraging higher-order topological structures, specifically cell complexes, (1) to train GNNs to inherently create more interpretable representations during the learning process, and (2) utilize their topological properties to refine explanations post-hoc.

3 Cell Complexes

Graphs are powerful structures that excel in modeling relations between objects. However, they are limited to modeling pairwise relationships. To overcome this, graphs can be generalized to work in higher dimensions and model group-wise relationships. This generalization in algebraic topology is achieved by *cell complexes* (Hansen and Ghrist 2019).

A cell complex X is constructed through a hierarchical assembly process. It begins with a set of vertices (0-cells), to which edges (1-cells) are attached by connecting these points with closed line segments, forming a graph. This structure is then expanded by attaching faces (2-cells) to corresponding 1-cells. While our focus remains within two dimensions, this process can be extended to higher dimensions[1].

While our work does not go into detail about the algebraic topology ideas linked to *cell complexes*, it is important to provide some basic definitions (Hansen and Ghrist 2019) and notations (Table 1) that are essential to understand the framework we are proposing.

Notation	Description
G	A general graph
V	Set of all vertices in a graph
E	Set of all edges in a graph
$G(V, E)$	A graph with vertices V and edges E
X	A general cell complex
$X^{(p)}$	p-skeleton of a cell complex
c, $c^{(p)}$	general cell, cell of dimension p
$C^{(p)}$	A p-chain in a given cell complex
C	Set of all p-chains in a cell complex
σ_{c_1,c_2}	A general boundary relation
Σ	Set of all boundary relations
$X(C, \Sigma)$	A cell complex defined by C and Σ

Table 1: Summary of notations used throughout the paper.

Definition 1 (p-cell)**.** A p-cell $c^{(p)}$ in a cell complex refers to an element of dimension p. In analogy to traditional graphs where we have vertices (0-dimensional) and edges (1-dimensional), cell complexes include these and extend to higher dimensions.

[1]For a more comprehensive understanding of cell complexes, we point the reader to www.math.ksu.edu/~hansen/CWcomplexes.pdf and Bodnar et al. (2021)

Definition 2 (*boundary*/*coboundary*)**.** In a cell complex, a p-cell $c^{(p)}$ is considered a *face* or *boundary* of a $(p+1)$-cell $c^{(p+1)}$ if the set of points composing $c^{(p)}$ is a subset of those composing $c^{(p+1)}$. Conversely, $c^{(p+1)}$ is referred to as the *coface* or *coboundary* of $c^{(p)}$.

Definition 3 (p-chain)**.** In a given cell complex, a p-chain $C^{(p)}$ is simply defined as the set of all p-dimensional cells. The general set union of all such $C^{(p)}$ is denoted by C.

Definition 4 (p-skeleton)**.** The p-skeleton of a cell complex X is defined as the subcomplex $X^{(p)}$ consisting of cells of dimension at most p. Using this definition, we see that $X^{(0)}$ is the set of all vertices and $X^{(1)}$ is the set of all vertices and edges which precisely make up the underlying graph.

Definition 5 (*boundary relation*)**.** Within a cell complex, a boundary relation σ is analogous to an edge in traditional graphs. It connects two cells, either of the same dimension (horizontal boundary relation) or different dimensions (vertical boundary relation). A horizontal boundary relation $\sigma_{c_1^{(p)},c_2^{(p)}}$ links two p-cells that share a common *boundary* or *coboundary*, whereas vertical boundary relations $\sigma_{c_1^{(p)},c_2^{(p+1)}}$ link a p-cell to its corresponding *boundaries* or *coboundaries*. In this work, we further restrict the boundary relations to be undirected, implying $\sigma_{c_1,c_2} = \sigma_{c_2,c_1}$.

Using the definitions provided in this section, we can analogously represent a cell complex as $X(C, \Sigma)$, like we represent a graph as $G(V, E)$. The methodology we introduce to create a cell complex from a given graph is described in Section 4.

While cell complexes are adept at handling higher-order interactions, their conventional definition requires them to be closed under taking subsets, making them inefficient data structures for scalable computation (Yang, Sala, and Bogdan 2022). To overcome this, we present the following theorem and a subsequent modification in Equation (3) to achieve an efficient, restricted form of cell complexes that is linear in space complexity. The proof follows by construction and is deferred to the Appendix.

Theorem 3.1. *For a graph $G(V, E)$ with adjacency matrix A having cycles of length at most K, let W_k represent the number of closed walks of length k which are not k-cycles. Let $deg(v)$ represent the degree of a node v in the graph. The corresponding cell complex X will have cells C and boundary relations Σ such that*

$$|C| = |V| + |E| + \sum_{k=3}^{K} \frac{1}{2k}[tr(A^{(k)}) - W_k] \tag{1}$$

$$|\Sigma| \geq 3|E| + \frac{1}{2}\sum_{k=3}^{K}[tr(A^{(k)}) - W_k] + \sum_{v \in V} \binom{deg(v)}{2} \tag{2}$$

This theorem provides us with two important bounds. Equation (1) shows that the number of cells in cell complex X grows *linearly* with the number of vertices and edges in the graph G. Equation (2) shows that asymptotically, the size

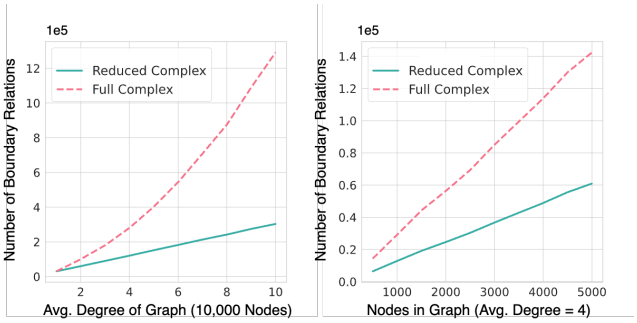

Figure 3: The variation of $|\Sigma|$ for a conventional cell complex and our proposed reduced cell complex with increasing $|E|$ (Theorem 3.1) (left) and increasing $|V|$ (right), showing that our proposed variation is more space efficient.

of Σ grows *quadratically* with $|E|$, due to the last term in the equation. This term is precisely the number of boundary relations that are present between two edges in the graph, which are represented by $\sigma_{c_1^{(1)},c_2^{(1)}}$. To preserve the scalability and complexity of FORGE, we further restrict how we create cell complexes from graphs, and drop all boundary relations of the form $\sigma_{c_1^{(p)},c_2^{(p)}}$ where $p \geq 1$ from our construction of cell complexes. Doing this reduces Equation (2) to:

$$|\Sigma| = 3|E| + \frac{1}{2}\sum_{k=3}^{K}[tr(A^{(k)}) - W_k] \tag{3}$$

This operation ensures that $|\Sigma|$ increases *linearly* with $|E|$, making cell complexes much more scalable. We empirically validate these results on large random graphs generated using the Erdős-Rényi method (Erdos, Rényi et al. 1960) and present the results in Figure 3.

Throughout the rest of this text, we use the term *cell complexes* to refer to these restricted/reduced cell complexes, unless stated otherwise.

4 Proposed Approach

Lifting Graphs to Cell Complexes

In this section, we present the lifting algorithm we introduce to create a cell complex X from its underlying graph G. Using the lifting algorithm on a given graph $G(V, E)$, we construct $C^{(0)}$, $C^{(1)}$, and $C^{(2)}$ from V, E, and cycles present in the graph respectively, and add the corresponding Σ to create $X(C, \Sigma)$. The lifting algorithm is described in Algorithm 1. For any given graph, the time complexity of the lifting algorithm is bounded by the time complexity of finding cycles in a graph[2]. Further details on the algorithm and how cell features are created to perform message passing are deferred to the Appendix.

After lifting the graph to its corresponding cell complex, the next step is to train a GNN on cell complexes. Because they are analogous to graphs, we can provide X as an input to any GNN, and obtain a trained model $\mathcal{F}$ which will generate representations for cells C.

[2]https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.cycles.simple_cycles

Algorithm 1: Lifting algorithm

Input: $G(V, E)$
Output: $X(C, \Sigma)$

$C^{(0)} \leftarrow \phi, C^{(1)} \leftarrow \phi, C^{(2)} \leftarrow \phi, \Sigma \leftarrow \phi$
for $e_{u,v} \in E$ **do**
  Add u, v to $C^{(0)}$
  Add $e_{u,v}$ to $C^{(1)}$
  Add $\sigma_{u,v}$, $\sigma_{u,e_{u,v}}$, $\sigma_{v,e_{u,v}}$ to Σ
end for
for $c \in$ Set of Cycles in G **do**
  Add c to $C^{(2)}$
  for $e_{u,v} \in c$ **do**
    Add $\sigma_{e_{u,v},c}$ to Σ
  end for
end for
$C \leftarrow C^{(0)} \cup C^{(1)} \cup C^{(2)}$
return $X(C, \Sigma)$

Generating Explanations for Cell Complexes

While graph explainers use diverse methods to generate explanations for GNN predictions, they can be abstracted as a function that takes in the trained GNN model and graph data and outputs a soft node or edge mask with values between 0 and 1 signifying the importance of each node and edge identified as the explanation. For simplicity, we continue the discussion by defining the explanation as a node mask, an analogous approach can be followed for an edge mask. Formally, let $\mathcal{M}$ be an explanation mask, then the output of a graph explainer $\mathcal{E}$ can be represented as

$$\mathcal{M} = \mathcal{E}(\mathcal{F}, G) \tag{4}$$

Graph explainers generate explanations on the *computation graph* of a node, which is the node's k-hop neighborhood for a k-layer GNN. Transitioning from traditional computation graphs, our approach utilizes cell complexes as the input to the function. This shift allows the integration of more complex structural data into the computational framework, through the introduction of *vertical message passing* happening through vertical boundary relations (Section 3) in the cell complex, in addition to the already existing (horizontal) message passing, shown in Figure 4. Consequently, the learned explanation mask is adapted to represent the computational complex, incorporating both the connectivity and hierarchical structure inherent in cell complexes. Formally, this can be represented by

$$\mathcal{M}_X = \mathcal{E}(\mathcal{F}, X) \tag{5}$$

By using a cell complex $X(C, \Sigma)$ as input, the explainer outputs an explanation mask $\mathcal{M}_X$, which is an intermediate explanation mask over C, indicating which cells are important for the model prediction.

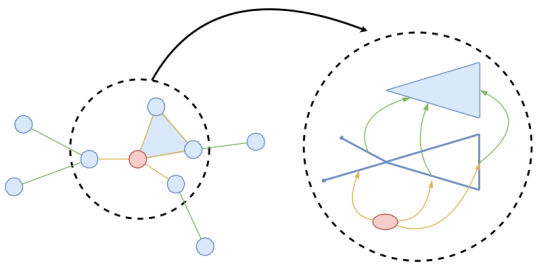

Figure 4: Example of a computation cell complex. The figure on the left shows 2-hop horizontal message passing, while the figure on the right represents 2-hop vertical message passing, introduced by FORGE.

Information Propagation

With the explainer generating an explanation mask $\mathcal{M}_X$ for the cell complex X, we must propagate this information from the higher-order structures back to the base graph structure to produce the final explanation for G. We term this process information propagation, as it involves transferring the learned importance values from the cell complex to the original domain.

We introduce multiple algorithms to perform information propagation, namely, (1) Hierarchical Propagation, (2) Direct Propagation, (3) Entropy Propagation, and (4) Activation Propagation. For brevity, we expand on (1) in detail and defer the description of other algorithms and optimizations to the Appendix. Hierarchical Propagation is described in Algorithm 2.

Algorithm 2: Hierarchical Propagation

Input: $\mathcal{M}_X$
Parameters: α_c, α_e
Output: $\mathcal{M}$

for $c^{(2)} \in C^{(2)}$ **do**
  for all $\sigma_{c^{(1)},c^{(2)}}$ containing $c^{(2)}$ **do**
    $\mathcal{M}_X[c^{(1)}] \leftarrow \mathcal{M}_X[c^{(1)}] + \big(\mathcal{M}_X[c^{(2)}] - 0.5\big) \times \alpha_c$
  end for
end for
for $c^{(1)} \in C^{(1)}$ **do**
  for all $\sigma_{c^{(0)},c^{(1)}}$ containing $c^{(1)}$ **do**
    $\mathcal{M}_X[c^{(0)}] \leftarrow \mathcal{M}_X[c^{(0)}] + \big(\mathcal{M}_X[c^{(1)}] - 0.5\big) \times \alpha_e$
  end for
end for
$\mathcal{M} \leftarrow \mathcal{M}_X[C^{(0)}]$
return $\mathcal{M}$

The main idea behind Hierarchical Propagation is to aggregate the explanations of 2-cells with their 1-cell *boundaries*, then again aggregate explanations of 1-cells with their 0-cell *boundaries*. These 0-cells represent the underlying 0-skeleton of X, which as mentioned in Section 3 is precisely the set of nodes V of $G(V, E)$. The goal is to utilize cell explanations in a way that polarises important and unimportant nodes in the final explanation mask $\mathcal{M}$. This is done in Algorithm 2 in steps 3 and 8, where we subtract 0.5 from the explanation masks of higher order cells to quantify how positively or negatively it should influence the overall explanation. They are further multiplied by parameters α_e and α_c, which are introduced to have fine-grained control over how much 1-cells and 2-cells, respectively, contribute to the graph explanation. By precomputing the required boundary relations, the time complexity of Information Propagation is reduced to $\mathcal{O}(|E|)$.

All the steps to obtain explanation $\mathcal{M}$ for a graph G and trained model $\mathcal{F}$ comprising FORGE can be summarised formally using the following equations:

$$\mathcal{M} = \text{FORGE}(\mathcal{E}, \mathcal{F}, G) \tag{6}$$

$$\text{FORGE}(\mathcal{E}, \mathcal{F}, G) = \text{PROP}(\mathcal{E}(\mathcal{F}, \text{LIFT}(G))) \tag{7}$$

Where LIFT is Algorithm 1, $\mathcal{E}$ is any graph explainer, and PROP is any information propagation algorithm.

5 Experimental Settings

Datasets

We evaluate FORGE on both real-world and synthetic datasets for a comprehensive evaluation across diverse conditions. We take real-world datasets from the graph explainability benchmark, GraphXAI (Agarwal et al. 2023), which includes Benzene, Mutagenicity, Alkyl Carbonyl, and Fluoride Carbonyl. For synthetic datasets, we generate random graphs with a distinct subgraph structure, known as a *motif*, which defines the ground truth for the graph. The task involves differentiating between two different motifs. Motifs we test include Bull, Square (4-cycle), Hexagon (6-cycle), Wheel, House, and Cube (Figure 5). The synthetic datasets are referred to as Motif1/Motif2 in the text based on the motifs present. Specific details about dataset generation can be found in the Appendix.

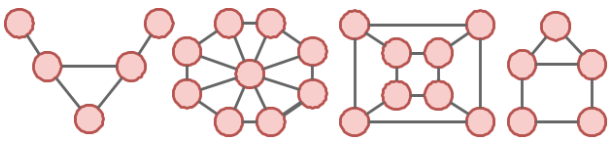

Figure 5: Different motifs used to generate synthetic graphs: (left to right) Bull, Wheel, Cube, and House.

Evaluation Criteria

Graph explainer methods produce edge and node masks which represents the most important subgraph that resulted in a model prediction. To evaluate the correctness of this mask, we compare it to the ground truth by adopting two metrics from Agarwal et al. (2023), Graph Explanation Accuracy (GEA) and Graph Explanation Faithfulness (GEF).

Datasets	GNNExplainer		GraphMask		GradExplainer		PGMExplainer		SubgraphX		Random
	B	F	B	F	B	F	B	F	B	F	
Benzene	0.456	**0.772**	0.276	0.378	0.167	0.353	0.109	0.198	0.450	0.614	0.194
AlkCarb	0.054	0.130	0.048	0.055	0.114	0.217	0.095	**0.304**	0.002	0.011	0.050
FluoCarb	0.207	**0.441**	0.077	0.230	0.233	0.439	0.115	0.300	0.079	0.095	0.154
Mutag	**0.380**	0.339	0.127	0.163	0.172	0.218	0.114	0.233	0.079	0.095	0.112
Bull/Square	0.126	0.433	0.082	0.241	0.179	0.298	0.080	0.150	0.355	**0.447**	0.145
House/Hex	0.107	0.478	0.115	0.271	0.199	0.410	0.088	0.158	0.346	**0.617**	0.165
Wheel/House	0.102	0.361	0.233	0.378	0.169	0.426	0.083	0.195	0.246	**0.549**	0.194
Cube/Wheel	0.114	0.339	0.183	0.402	0.119	**0.494**	0.091	0.178	0.397	0.489	0.221

Table 2: Graph Explanation Accuracy (GEA) (↑) scores for baseline explainers **(B)** against their FORGE variants (**F**) across all datasets, with FORGE improving performance across various baselines. The best result for each dataset is highlighted in **bold**. Underlined values represent the better result between a base explainer and its FORGE variant.

GEA uses Jaccard Index (Taha and Hanbury 2015) to quantify explanation accuracy, and GEF measures how faithful the explanations are to the underlying GNN predictor using KL divergence (Kullback and Leibler 1951). Further details about the metrics are present in the Appendix.

Baselines

We select a range of established explainer methods as baselines to evaluate FORGE. For perturbation methods, we select GNNExplainer (Ying et al. 2019), GraphMask (Schlichtkrull, Cao, and Titov 2022), and SubgraphX (Yuan et al. 2021). For surrogate methods, we evaluate PGMxplainer (Vu and Thai 2020), and for gradient-based methods, we select GradExplainer (Selvaraju et al. 2019). We use a random explainer as a naive baseline adapted from Agarwal et al. (2023).

6 Results

Table 2 presents the GEA scores for various explainers with and without our framework. The reported results are averaged over 10 different seeds, with standard deviations reported in the Appendix. Our experiments reveal several key insights into the performance of different GNN explainers across various datasets. FORGE consistently improves existing explainers across various datasets with improvements up to 315% (FORGE-enhanced GradExplainer on Cube/Wheel), except Mutag for GNNExplainer. The GEF scores are presented in Table 3, reinforcing the capabilities of our framework in creating explanations that are comparable or more faithful to the underlying GNN.

Our results reveal substantial variability in explainer effectiveness depending on the graph types and tasks. No single method consistently outperforms others across all datasets, suggesting the importance of choosing explainers tailored to specific problem domains. Perturbation-based methods benefit the most from our framework, with FORGE applied to GNNExplainer generally delivering the best performance on real-world datasets, while FORGE on SubgraphX excels in synthetic datasets. Surrogate and gradient-based methods also show notable improvements across all datasets when enhanced with FORGE. Interestingly, some explainers in their default form perform worse than the Random baseline, a finding supported by the GraphXAI benchmark. However, after applying FORGE, all explainers consistently surpass the Random baseline, achieving significantly improved performance.

Explainer	GEF(↓)	
	B	F
GNNExplainer	$0.189_{\pm 0.04}$	$\mathbf{0.083_{\pm 0.02}}$
GraphMask	$0.051_{\pm 0.03}$	$\mathbf{0.028_{\pm 0.02}}$
GradExplainer	$0.389_{\pm 0.13}$	$\mathbf{0.078_{\pm 0.05}}$
PGMExplainer	$\mathbf{0.204_{\pm 0.05}}$	$0.234_{\pm 0.03}$
SubgraphX	$0.008_{\pm 0.01}$	$\mathbf{0.007_{\pm 0.00}}$
Random	$0.124_{\pm 0.04}$	-

Table 3: Graph Explanation Faithfulness (GEF) Scores for the AlkaneCarbonyl Dataset. FORGE-enhanced explainers generate explanations that are comparable or more faithful to the underlying GNN. Values in **bold** indicate best performance.

For reproducibility of our results, all implementation details are contained in the Appendix. Further experiments and additional results on all datasets for presented experiments are also presented in the Appendix.

7 Ablation Studies

FORGE Components

FORGE comprises two key components: Lifting and Information Propagation. To evaluate their contributions, we compare FORGE's performance against **Base-LIFT** (a version that applies regular explainers on lifted graphs), **FORGE-LIFT** (a version of FORGE that only performs lifting), and **Base** (which has neither Lifting nor Propagation). Figure 6 demonstrates that FORGE-LIFT, by itself, consistently outperforms the base explainers across all four methods. While Base-LIFT performs better than the baseline, we see that training the underlying GNN on the lifted graphs is important to achieve further performance gains. This improvement highlights the significant impact of the

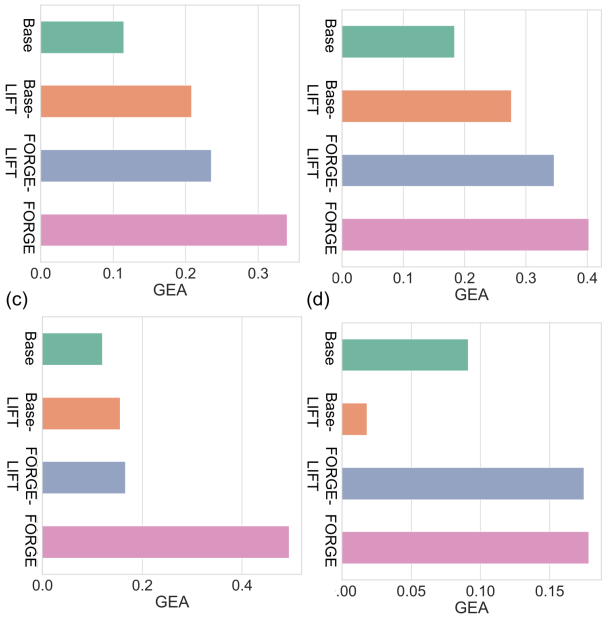

Figure 6: Results of ablations on FORGE Components for **(a)** GNNExplainer, **(b)** GraphMaskExplainer, **(c)** GradExplainer, **(d)** PGMExplainer on Synth-Wheel/Cube dataset. Both Lifting and Information Propagation contribute significantly to an increase in explanation accuracy.

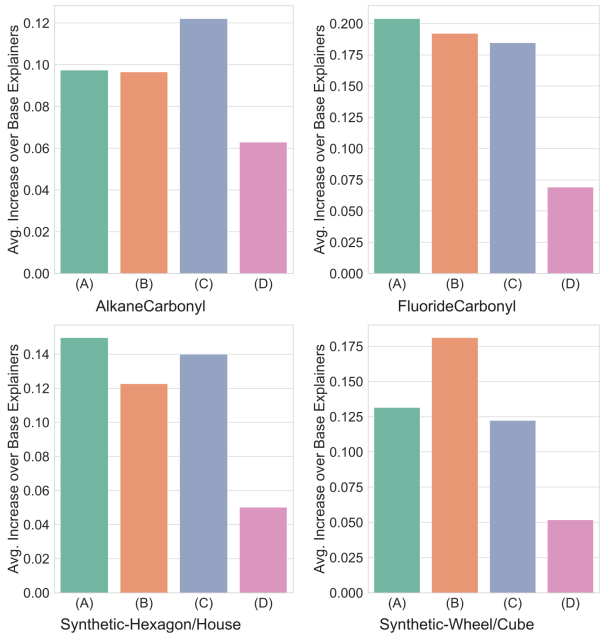

Figure 7: Ablation results for different propagation methods on various datasets, for all baseline explainers. The y-axis represents the average absolute increase in GEA over base explainers. **(A)** Hierarchical Prop, **(B)** Direct Prop, **(C)** Activation Prop, and **(D)** Entropy Prop.

Lifting component alone in generating more accurate explanations. Furthermore, when we incorporate Information Propagation to create the full FORGE framework, we observe an additional performance boost on top of Lifting. This enhancement is particularly noticeable in GNNExplainer and GradExplainer (subfigures (a) and (c)).

Interestingly, the impact of Information Propagation appears to vary across different explainers. For instance, its effect seems more pronounced in GNNExplainer and GradExplainer compared to GraphMaskExplainer and PGMExplainer. This variation suggests that the benefits of our proposed propagation strategies may depend on the underlying explanation mechanism. These findings collectively demonstrate that both Lifting and Information Propagation are crucial components of FORGE, each contributing significantly to the framework's overall performance.

Information Propagation Algorithms

Figure 7 presents our second ablation study, comparing four information propagation algorithms across four different datasets for all explainers, reporting the average GEA. This analysis reveals two key insights: **(a) Algorithm Performance Variability:** the effectiveness of propagation methods varies significantly across datasets, indicating that no single algorithm consistently outperforms the others in all scenarios, and **(b) Dataset Dependency:** the optimal choice of propagation algorithm seems to be heavily influenced by the specific dataset being analyzed. This suggests that the graph structure and data properties play an important role in determining the most effective propagation method.

While performance varies, Hierarchical and Direct Propagation methods tend to perform well across most datasets, suggesting they may be more versatile. Activation Propagation shows high effectiveness in certain cases, particularly noteworthy in the AlkaneCarbonyl dataset.

The findings from Figure 7 emphasize the importance of carefully selecting the propagation method based on the specific dataset and graph structure. They also highlight the need for adaptive or hybrid approaches that can leverage the strengths of different propagation algorithms depending on the context.

8 Conclusion and Future Work

We introduce FORGE, a novel framework that enhances GNN explainability by leveraging cell complexes. Our framework employs a novel lifting algorithm to convert graphs to cell complexes. Furthermore, we introduce information propagation algorithms to create more interpretable internal data representations. Our extensive evaluations demonstrate that FORGE consistently enhances explanation accuracy and faithfulness across both real-world and synthetic datasets. Future work could explore adaptive propagation approaches and investigate FORGE's applicability to diverse graph learning tasks. We hope this work motivates further research into applying higher-order structures for enhanced interpretability.

Acknowledgments

We are deeply grateful to the Precog research group for their collective support throughout this work. We extend special thanks to Shashwat Singh and Vamshi Krishna Bonagiri for their invaluable insights.

References

Agarwal, C.; Lakkaraju, H.; and Zitnik, M. 2021. Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*, 2114–2124. PMLR.

Agarwal, C.; Queen, O.; Lakkaraju, H.; and Zitnik, M. 2023. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1): 144.

Bodnar, C.; Frasca, F.; Otter, N.; Wang, Y.; Liò, P.; Montufar, G. F.; and Bronstein, M. 2021. Weisfeiler and Lehman Go Cellular: CW Networks. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 2625–2640. Curran Associates, Inc.

Chen, D.; Schulz, T. H.; and Borgwardt, K. 2024. Learning Long Range Dependencies on Graphs via Random Walks. arXiv:2406.03386.

Duvenaud, D.; Maclaurin, D.; Aguilera-Iparraguirre, J.; Gómez-Bombarelli, R.; Hirzel, T. D.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. volume 28. MIT Press.

Ebli, S.; Defferrard, M.; and Spreemann, G. 2020. Simplicial neural networks. *arXiv preprint arXiv:2010.03633*.

Erdos, P.; Rényi, A.; et al. 1960. On the evolution of random graphs. *Publ. math. inst. hung. acad. sci*, 5(1): 17–60.

Fout, A.; Byrd, J.; Shariat, B.; and Ben-Hur, A. 2017. Protein Interface Prediction using Graph Convolutional Networks. volume 30. Colorado State University. Libraries.

Gilmer, J.; Schoenholz, S. S.; Riley, P.; Vinyals, O.; and Dahl, G. E. 2017. Neural Message Passing for Quantum Chemistry. PMLR.

Hansen, J.; and Ghrist, R. 2019. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3(4): 315–358.

Huang, Q.; Yamada, M.; Tian, Y.; Singh, D.; Yin, D.; and Chang, Y. 2020. GraphLIME: Local Interpretable Model Explanations for Graph Neural Networks. arXiv:2001.06216.

Kakkad, J.; Jannu, J.; Sharma, K.; Aggarwal, C.; and Medya, S. 2023. A Survey on Explainability of Graph Neural Networks. arXiv:2306.01958.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. arXiv:1609.02907.

Kullback, S.; and Leibler, R. A. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1): 79–86.

Luo, D.; Cheng, W.; Xu, D.; Yu, W.; Zong, B.; Chen, H.; and Zhang, X. 2020. Parameterized Explainer for Graph Neural Network. arXiv:2011.04573.

Ragkousis, M., Flogera. 2022. MFSE: A Meta-Fusion Model for Polypharmacy Side-Effect Prediction with Graph Neural Networks.

Schlichtkrull, M. S.; Cao, N. D.; and Titov, I. 2022. Interpreting Graph Neural Networks for NLP With Differentiable Edge Masking. arXiv:2010.00577.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2019. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *International Journal of Computer Vision*, 128(2): 336–359.

Shan, C.; Shen, Y.; Zhang, Y.; Li, X.; and Li, D. 2021. Reinforcement Learning Enhanced Explainer for Graph Neural Networks. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 22523–22533. Curran Associates, Inc.

Spinelli, I.; Scardapane, S.; and Uncini, A. 2024. A Meta-Learning Approach for Training Explainable Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4): 4647–4655.

Springenberg, J. T.; Dosovitskiy, A.; Brox, T.; and Riedmiller, M. 2015. Striving for Simplicity: The All Convolutional Net. arXiv:1412.6806.

Taha, A. A.; and Hanbury, A. 2015. Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC medical imaging*, 15: 1–28.

Vu, M. N.; and Thai, M. T. 2020. PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks. arXiv:2010.05788.

Wang, X.; He, X.; Cao, Y.; Liu, M.; and Chua, T.-S. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. ACM.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? arXiv:1810.00826.

Yang, R.; Sala, F.; and Bogdan, P. 2022. Efficient Representation Learning for Higher-Order Data with Simplicial Complexes. In *Learning on Graphs Conference*, 13–1. PMLR.

Ying, R.; Bourgeois, D.; You, J.; Zitnik, M.; and Leskovec, J. 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. arXiv:1903.03894.

Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; and Leskovec, J. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. ACM.

Yu, Z.; and Gao, H. 2022. Motifexplainer: a motif-based graph neural network explainer. *arXiv preprint arXiv:2202.00519*.

Yuan, H.; Yu, H.; Wang, J.; Li, K.; and Ji, S. 2021. On Explainability of Graph Neural Networks via Subgraph Explorations. arXiv:2102.05152.

Zhang, X.-M.; Liang, L.; Liu, L.; and Tang, M.-J. 2021. Graph Neural Networks and Their Current Applications in Bioinformatics. *Frontiers in Genetics*, 12.