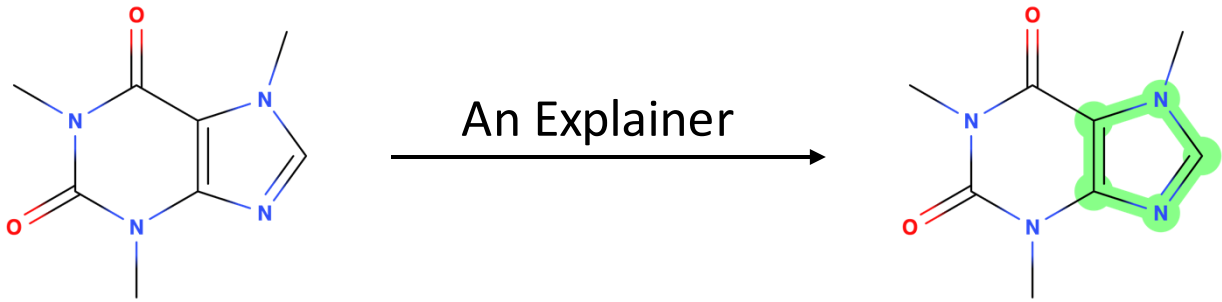
MAGE: Model-Level Graph Neural Networks Explanations via Motif-based Graph Generation

Zhaoning Yu, Hongyang Gao

IOWA STATE UNIVERSITY

Interpretability of Graph Neural Networks

1. Instance-level Explanation: pinpoint specific nodes, edges, or subgraphs crucial for a GNN model's predictions on one data instance.



Input instance molecule

Green highlight an explanation

Interpretability of Graph Neural Networks

1. Instance-level Explanation

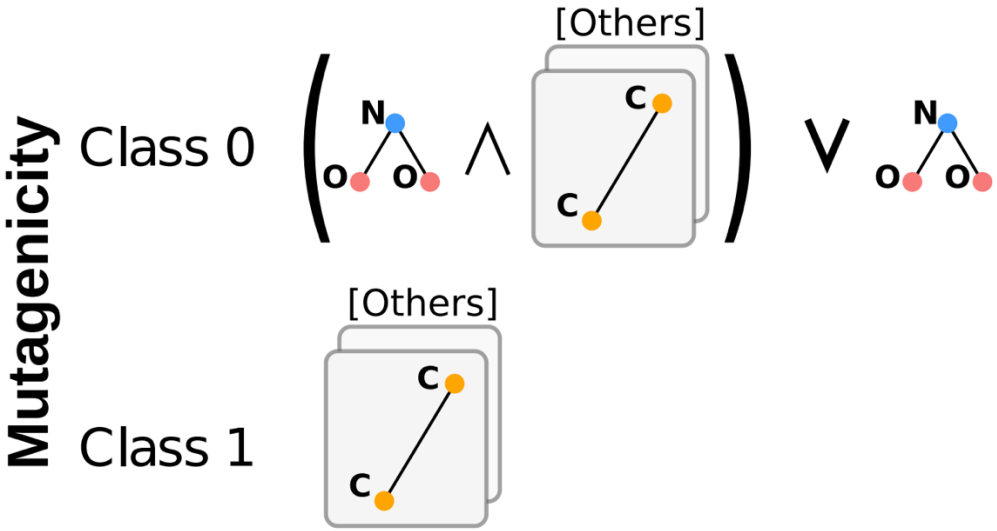
2. Model-level Explanation: seek to demystify the overall behavior of the GNN model by identifying patterns that generally lead to certain predictions.

Key Difference:

1. Instance-level methods offer detailed insights for each individual graph
2. Model-level methods provide more high-level and broader insights explanations

Two Types of Model-level Explanation

1. Concept-based methods: identify higher-level concepts that significantly influence the model's predictions and establish rules to illustrate how these concepts are interconnected, like using logical formulas



Azzolin, Steve, et al. "Global explainability of gnns via logic combination of learned concepts." *arXiv preprint arXiv:2210.07147* (2022).

Two Types of Model-level Explanation

1. Concept-based methods

2. Generation-based methods: learn a generative model that can generate synthetic graphs that are optimized to maximize the behavior of the target GNN

Key Notes:

Instead of defining the relationships between important concepts with formulas, generation-based models can generate novel graph structures that are optimized for specific properties.

Limitation of Existing Generation-based methods

Existing generation-based model-level explainers for GNNs are somewhat limited in their application, especially in molecular graphs

- They does not fully consider the validity of generated explanations

For example:

1. 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
2. GNNInterpreter (Wang & Shen, 2022) 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

Yuan, Hao, et al. "Xgnn: Towards model-level explanations of graph neural networks."
Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020.
Wang, Xiaoqi, and Han-Wei Shen. "Gnninterpreter: A probabilistic generative model-level explanation for graph neural networks."
arXiv preprint arXiv:2209.07924 (2022).

Our Motif-bAsed GNN Explainer (MAGE)

Target: 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 $f(\cdot)$.

Our method uses motifs as fundamental elements for interpreting the overarching behavior of a trained GNN for molecular graph at the model level.

Our Motif-bAsed GNN Explainer (MAGE)

Strength: prevents the creation of potentially invalid intermediate substructures, and focuses solely on the arrangement of motifs rather than the specific placement of each atom and bond

Main process:

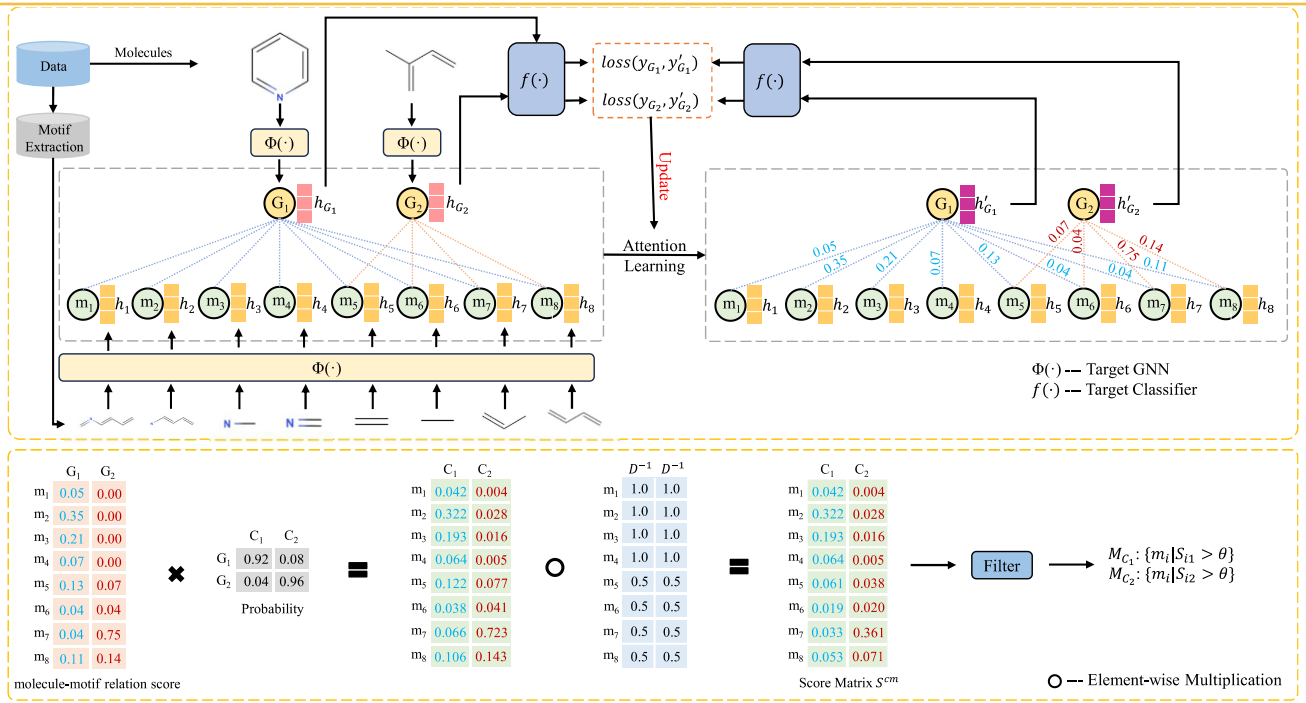
1. Identify all potential motifs within the dataset
2. An attention-driven motif learning phase is employed to identify the most significant motifs for each class
3. A graph generator uses the identified motifs to explain each class

Motif Extraction

Four types of existing methods

1. Ring and bonded pairs
2. Molecule decomposition
3. Molecule tokenization
4. Data-driven method

Class-wise Motif Identification



Stage 1: molecule-motif relation score calculation

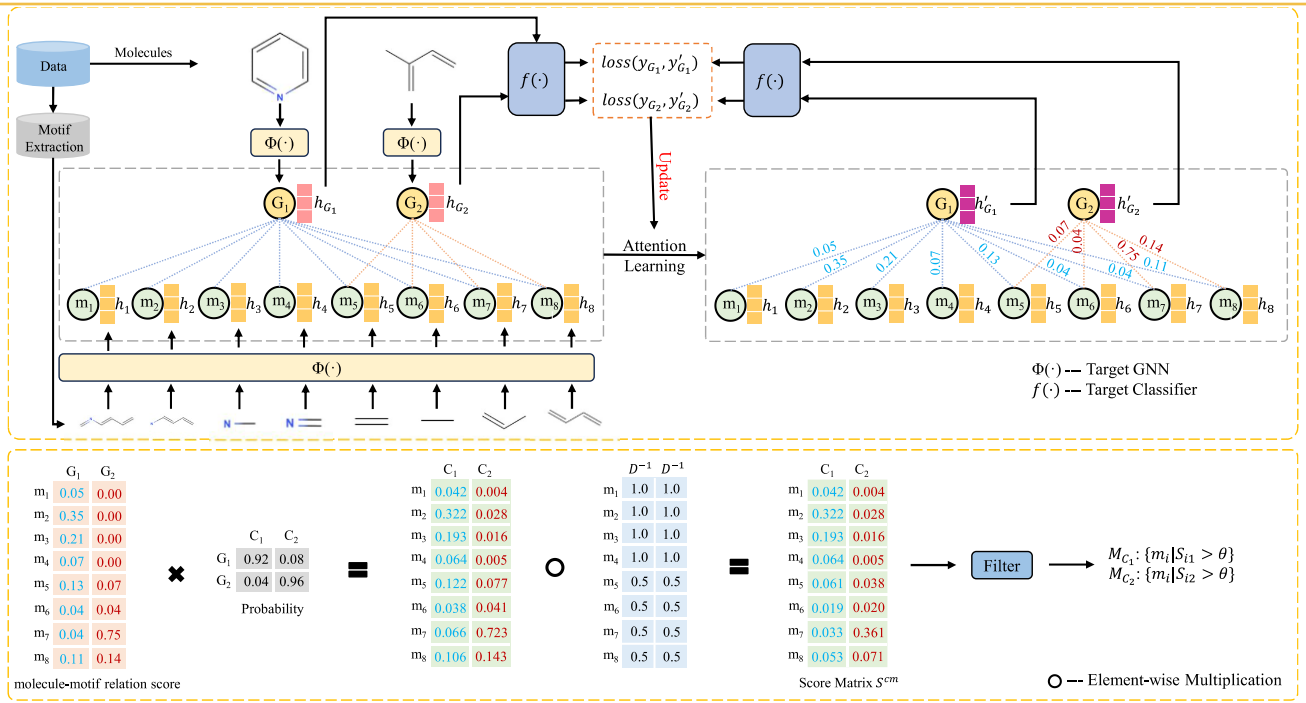
$$h_{G_i} = \phi(G_i)$$

$$h_{m_i} = \phi(m_i), for\ all\ m_i\ in\ M_i$$

$$e_{G_i m_i} = g\left(W h_{G_i}, W h_{m_i}\right), \alpha_{G_i m_i} = \exp(e_{G_i m_i}) \Big/ \sum_{m_i \in M_i} \exp(e_{G_i m_i}), h'_{G_i} = \sum_{m_i \in M_i} \alpha_{G_i m_i} \cdot h_{m_i}$$

$$\mathcal{L} = \sum_i \mathcal{L}(f(h_{G_i}), f(h'_{G_i}))$$

Class-wise Motif Identification

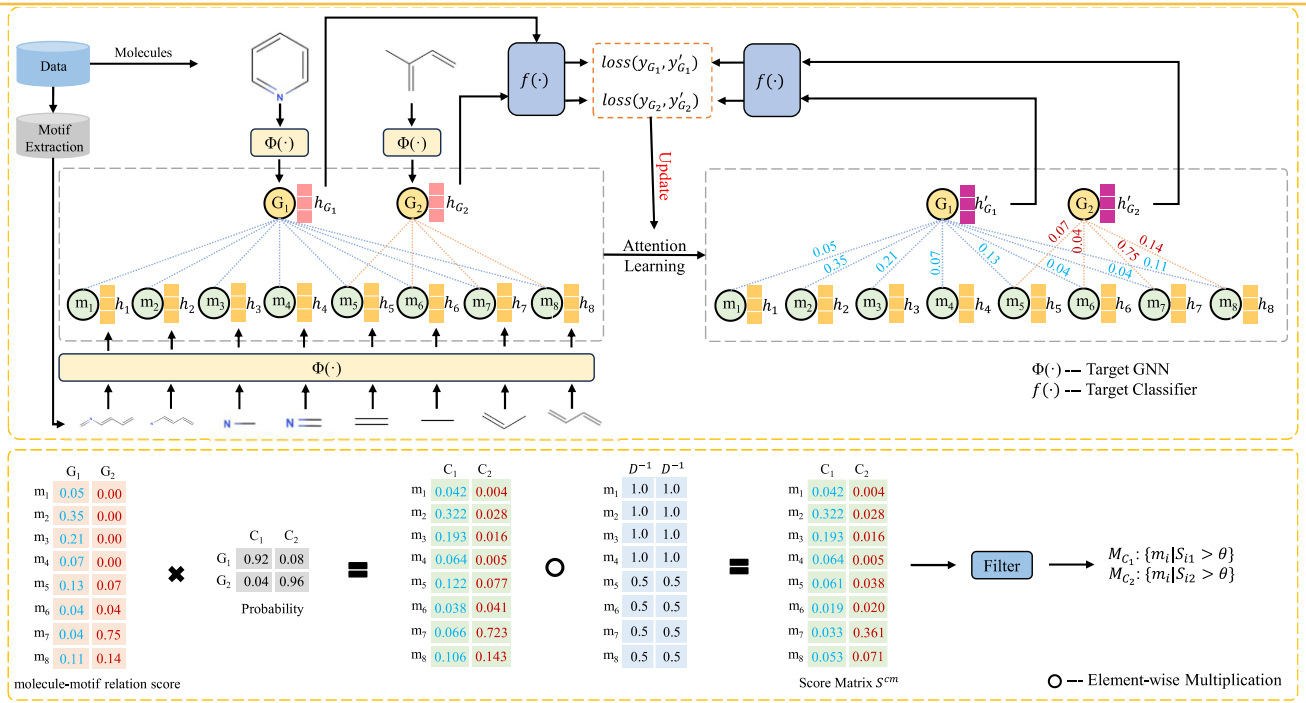


Stage 2: class-motif relation score calculation

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

1. S^{mm}_{pk} represents the molecule-motif relation score between molecule k and motif p
2. Element P_{kr} denotes the probability that molecule k is associated with label r
3. We normalize S^{cm} by dividing each column by the number of molecules it appears.

Class-wise Motif Identification



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

$$M_{C_r} = \{m_i | S_{ir}^{cm} > \theta, 1 \leq i \leq |M|\}$$

where θ is a hyper-parameter to control the size of the important motif vocabulary

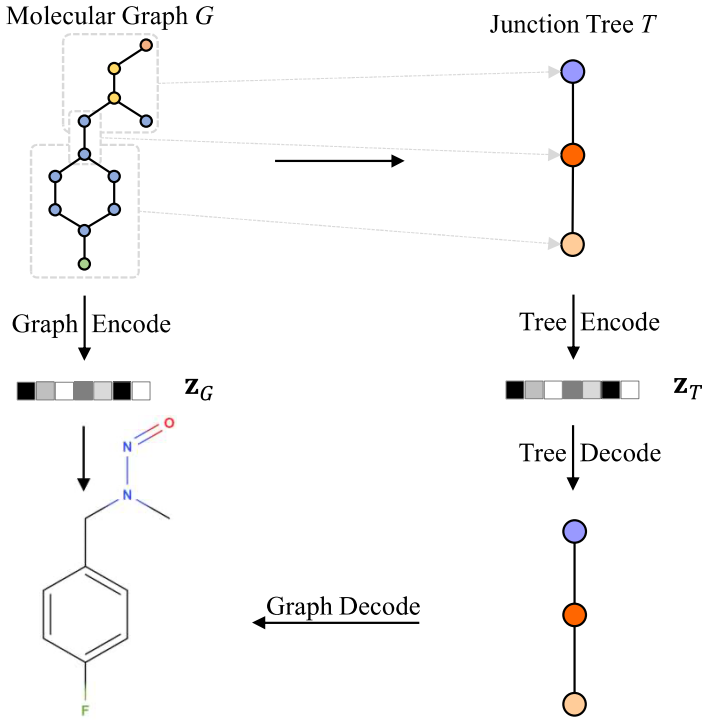
Class-wise Motif-based Graph Generation

We employ a VAE-based method as a generator to generate model-level explanation

Class-wise Motif-based Graph Generation

Component 1: Tree Decomposition

Given an input molecular graph, we decompose it into a junction tree.

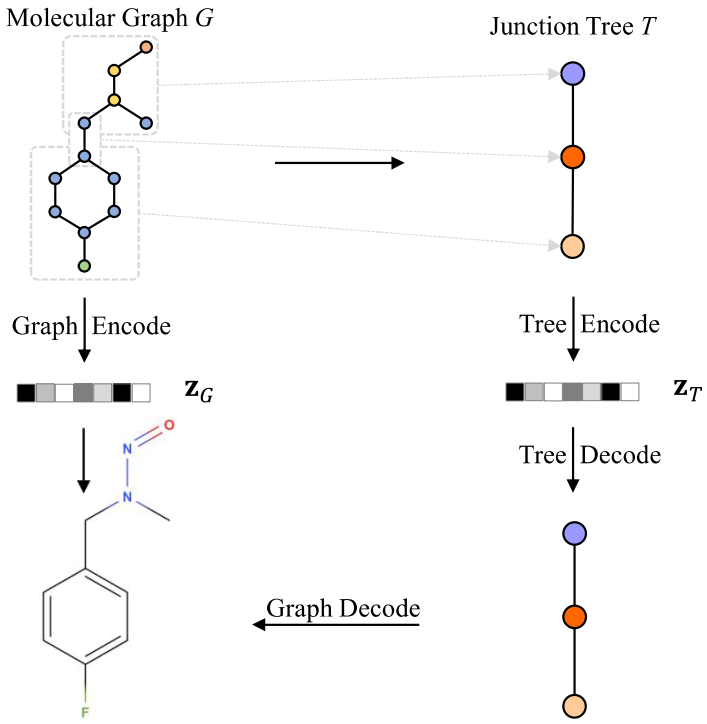


Class-wise Motif-based Graph Generation

Component 1: Tree Decomposition

Component 2: Graph Encoder

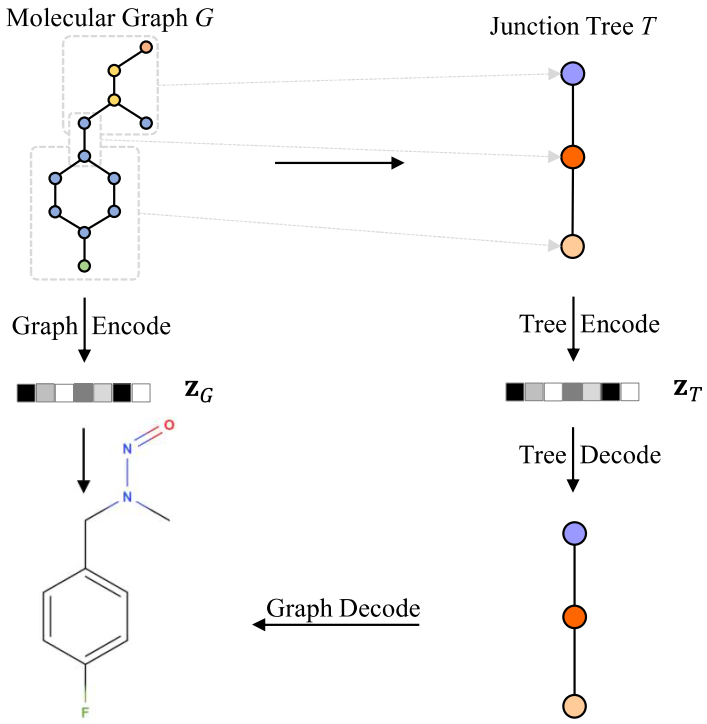
A graph encoder is used to encode the latent representation of input graph, here we use the target GNN feature extractor $\phi(\cdot)$ as the graph encoder



Class-wise Motif-based Graph Generation

Component 1: Tree Decomposition
Component 2: Graph Encoder
Component 3: Tree Encoder

- A graph neural network is employed to encode the junction tree.
- In the variational posterior approximation, we use two different affine layers to compute the mean, represented as μ_G, and the log variance, denoted as $log\sigma_G$, from h_G. Then, we sample z_G from a Gaussian distribution $\mathcal{N}(\mu_G, \sigma_G)$.

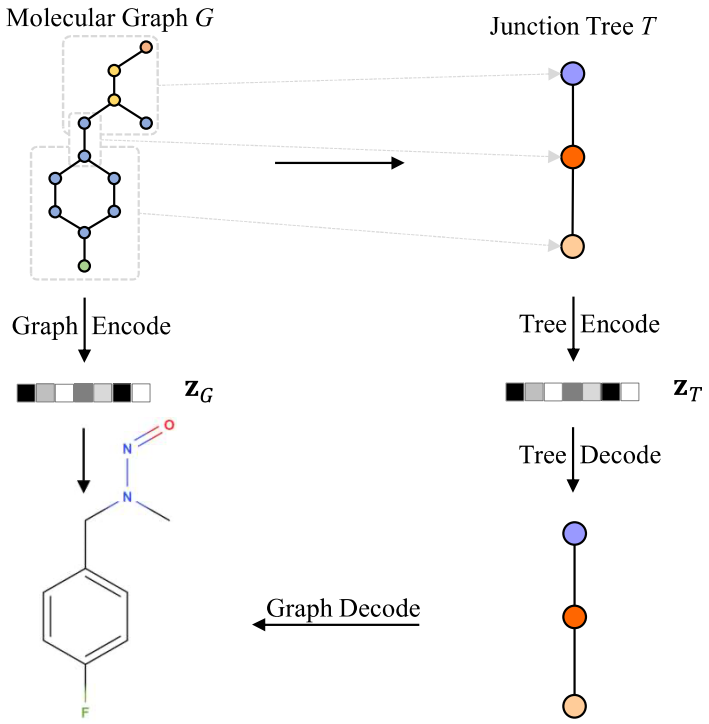


Class-wise Motif-based Graph Generation

Component 1: Tree Decomposition
Component 2: Graph Encoder
Component 3: Tree Encoder
Component 4: Tree Decoder

- The decoder builds the tree in a top-down approach, where nodes are created sequentially, one after the other.
- The decoder gets the node embedding of current node, then use the embedding to determining whether the node has a child and identifying the child's label if a child exists.

$$H' = GNN(A', X')$$
$$p_i = PRED(COMB(z_{\mathcal{T}}, H_i'))$$
$$q_i = PRED^l(COMB^l(z_{\mathcal{T}}, H_i'))$$

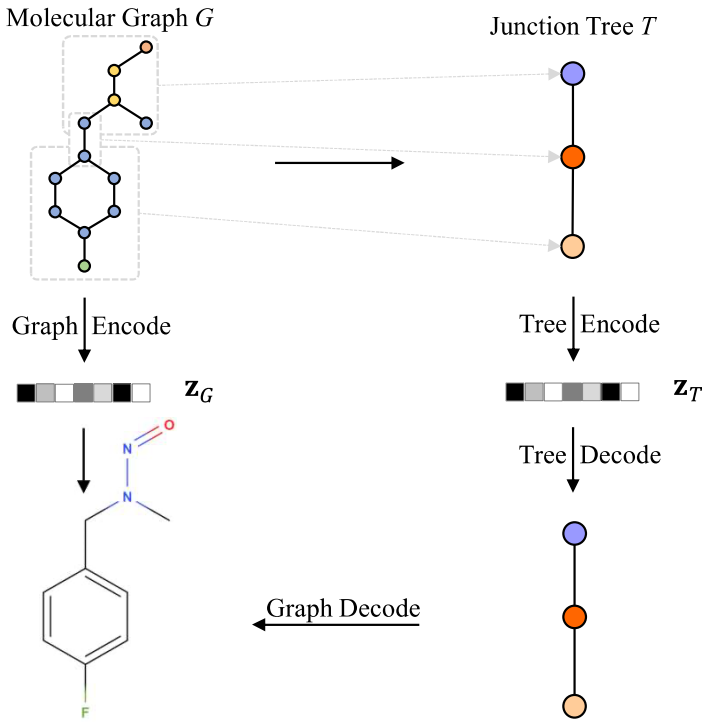


Class-wise Motif-based Graph Generation

Component 1: Tree Decomposition
Component 2: Graph Encoder
Component 3: Tree Encoder
Component 4: Tree Decoder
Component 5: Graph Decoder

- The decoder aims to find $\hat{G}$ as

$$\hat{G} = argmax_{G' \in \mathcal{G}(\mathcal{T})} f^a(G')$$
$$f^a(G') = f\big(\phi(G_i)\big)[r]$$



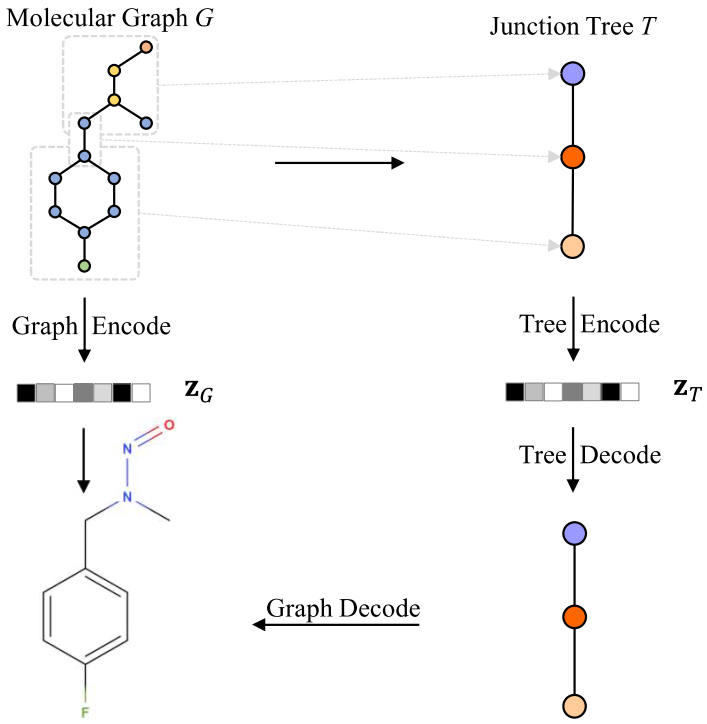
Class-wise Motif-based Graph Generation

Component 1: Tree Decomposition
Component 2: Graph Encoder
Component 3: Tree Encoder
Component 4: Tree Decoder
Component 5: Graph Decoder
Component 6: Loss Function

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

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

$$\mathcal{L}_{\mathcal{P}} = \sum MSE(h_G, h_{\mathcal{T}}) + \sum \mathcal{L}(f(h_{\mathcal{T}}), r)$$



Experiments (Datasets and 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.
- We compare our MAGE model with two state-of-the-art baselines: XGNN and GNNInterpreter.
- We uses two distinct metrics to assess performance: Validity and Average Probability.

$$Validity = \frac{\#\, Valid\ Molecules}{\#\, Generated\ Molecules}$$

$$Average\ Probability = \frac{\sum Class\ Probability}{\#\, Generated\ Molecules}$$

Experiments (Quantitative Results)

- Validity

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

Experiments (Quantitative Results)

- Average Probability

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