The Graph Concept Bottleneck: Decoding Combinatorial Reasoning in GNNs for Interpretability

Yue Niu[4], **Zhokai Sun**[1], **Jiayi Yang**[1], **Chunchun Chen**[4]
Xiaofeng Cao[2], **Rui Fan**[1], **Xin Sun**[3], **Wei Ye**[1,4,*]
[1]College of Electronic and Information Engineering, Tongji University, China
[2]School of Computer Science and Technology, Tongji University, China
[3]Faculty of Data Science, City University of Macau, China
[4]Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, China
2411947@tongji.edu.cn, yew@tongji.edu.cn

Abstract

GNNs have achieved remarkable success in graph learning, yet their black-box nature obscures the combinatorial reasoning behind their predictions. A core challenge lies in understanding how GNNs translate topological patterns (graph concepts) into logical rules. Current works only uncover hard Boolean logical rules over graph concepts, which cannot quantify the contribution of each concept to predictions. Moreover, they are post-hoc methods that generate explanations after training via surrogate models, and thus may deviate from the true combinatorial reasoning of GNNs. In this work, we develop the graph concept bottleneck that enforces the combinatorial reasoning of GNNs to fit differentiable soft combinatorial rules over graph concepts, thereby quantifying the contribution of each concept. To further enhance the graph concept bottleneck, we treat graph concepts as "graph words" and graphs as "graph sentences", and leverage language models to learn context-aware graph concept embeddings. Extensive experiments show that our method GCBMs achieve superior performance in both interpretability and classification. Our codes are available at https://github.com/niu131/gcbm-main.

1 Introduction

Graph neural networks (GNNs) [19, 23, 53, 59, 69] are prevalent in graph learning. However, as inherent black-box models, their combinatorial reasoning behind predictions is opaque to humans. This limits their application in high-stakes fields such as drug discovery. A prediction without a logical "why" is often implausible and unusable. Humans cannot verify whether GNNs use valid chemical properties or simply exploit spurious correlations in the dataset for prediction.

Combinatorial reasoning in GNNs refers to the internal decision process that discovers, evaluates, and composes discrete structural patterns—such as subgraphs or motifs—into a unified global decision rule. Since GNNs naturally model the discrete combinatorial structure of graphs, combinatorial reasoning goes beyond simple statistical correlation; it reflects a learned structural logic that can, at least in principle, be articulated as a Boolean formula over graph concepts (subgraphs or motifs).

Most interpretable methods [63, 67, 57, 65, 64, 70, 44, 41] enhance graph learning interpretability through modulating cluster information flow, but fail to decode combinatorial reasoning in GNNs. Recently, GraphTrail [3] and GLGExplainer [4] express the combinatorial reasoning of a trained GNN Φ as C (the number of graph classes) Boolean formulas $\{f_1, \ldots, f_C\}$ of graph concepts such that for any graph G, if $\Phi(G) = \mathbf{y}_i$ (the one-hot label of the i-th class), then $f_i(\mathcal{C}) = True$ and $\forall j \neq i$, $f_j(\mathcal{C}) = False$, where $\mathcal{C}$ contains all unique graph concepts in the dataset. While these formulas are

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

human-interpretable, they are hard logical rules (i.e., they cannot quantify the importance of each concept) and just proxies (post-hoc interpretation) for a GNN's prediction.

To tackle these limitations, we develop a globally and intrinsically interpretable framework called Graph Concept Bottleneck Models (GCBMs). Its core is the graph concept bottleneck that can be integrated into various GNNs and trained end-to-end. GCBMs first predict graph concept scores, then map these scores to class labels using the concept bottleneck layer, which extracts combinatorial reasoning (a differentiable soft combinatorial rule) between graph concepts and classes as a global interpretation. Thus, the contribution of each concept to prediction is quantified. Existing work [31] has proven that linear classifiers can represent Boolean logic and its continuous relaxation, providing the foundational theory for the concept bottleneck layer to realize combinatorial reasoning. The layer forces the retention of only critical combinatorial components. Thus, it is the model's learnable reasoning engine that directly encodes the combinatorial logic of concepts for prediction. Furthermore, the concept bottleneck naturally provides an interface that enables manual intervention to improve model accuracy while retaining interpretability.

Building GCBMs involves three key challenges: graph concept extraction, discriminative concept selection, and high-quality graph concept label generation. Specifically, we first adopt WL-subtrees [45] as graph concepts, as the message-passing mechanism of most GNNs follows the aggregate and update rule of the first-order Weisfeiler-Lehman (WL) graph isomorphism test [26]. Second, we select discriminative concepts based on concept frequency via information gain [21]. Finally, we obtain the concept scores by computing the similarity between the concept frequency vectors of the graphs and the one-hot encodings of the concepts. To improve the accuracy of the concept scores, we treat concepts as "graph words" and graphs as "graph sentences", use language models (LMs) to learn concept embeddings to capture their correlations and co-occurrences, and compute accurate concept scores based on these embeddings to optimize the quality of the graph concept bottleneck.

Our contributions can be summarized as follows:

- We propose Graph Concept Bottleneck Models (GCBMs), a new globally and intrinsically interpretable paradigm for GNNs that decodes the soft combinatorial reasoning of GNNs' predictions.
- GCBMs are generic interpretation frameworks for GNNs. They can use various GNNs as the backbone, graph substructures as the concept, and LMs as the concept embedding method.
- We verify GCBMs' interpretability, classification performance, and intervenability on benchmark graph datasets across multiple domains, showing the advantages of GCBMs over baselines.

2 Related Work

Interpretable GNNs Current GNN interpretability research is mainly categorized into post-hoc explanation and intrinsic interpretability. Post-hoc methods such as GNNExplainer [63] explain predictions with auxiliary models but produce unstable results and spurious features [28, 54]. GraphTrail [3], GLGExplainer [4], and LogiX [38] are post-hoc methods that approximate hard logical rules via surrogates and fail to uncover the true combinatorial reasoning. LogiX-GIN [39] integrates graded modal logic into GNN layers to construct a logic-based self-explainable GNN, yet it cannot quantify the importance of individual logical elements. Intrinsic methods embed explanatory logic in model design, such as GIB [66] and ProtGNN [70]. While both can express the similarity between the global input graph and prototypes, they are unable to represent explicit logical relations among multiple independent subgraphs, concepts, or prototypes.

Concept Bottleneck Models Concept bottleneck models (CBMs) were first proposed for images [24] and have been extended to the text domain. Notably, all CBMs are intrinsically interpretable. In the image domain, methods such as Label-free CBM [33], LaBo [61], and VLG-CBM [47] optimize concept sets and improve their adaptability to different visual classification tasks. Graph-CBMs [58] treat image concepts as nodes to construct latent concept graphs, thus capturing concept correlations, which differs from our GCBMs designed explicitly for graph data. In the text domain, frameworks such as TBMs [27, 49] and CB-LLM [48] integrate CBMs with large language models for interpretable text-related tasks. Only a few attempts, such as CBGNN [17], introduce concept bottlenecks into GNNs, but they rely on high-cost manually defined algorithmic concepts with extremely limited numbers, resulting in poor expressiveness.

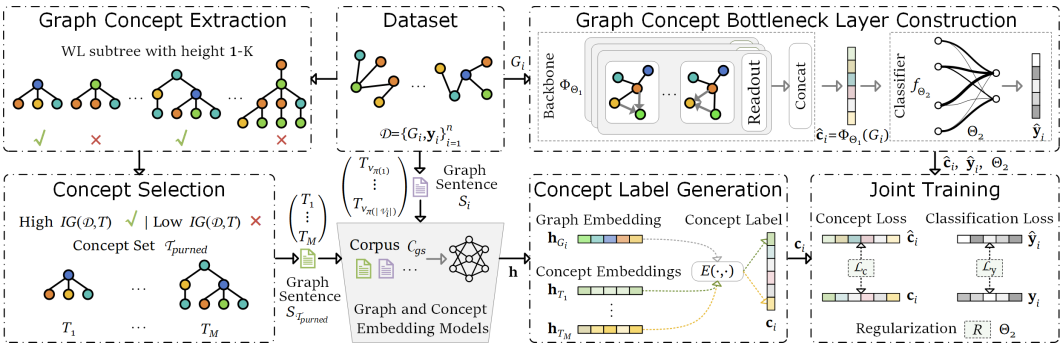

Figure 1: Overview of the GCBMs construction pipeline.

3 Preliminaries

An undirected and labeled graph is denoted by $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ is a finite node set with features $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_{|\mathcal{V}|}]^T \in \mathbb{R}^{|\mathcal{V}| \times d}$ ($\mathbf{x}_i \in \mathbb{R}^d$ is the feature vector of node v_i), and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} = \{(v_i, v_j) \mid v_i, v_j \in \mathcal{V}\}$ is the edge set, where an edge $e = (v_i, v_j)$ exists iff $e \in \mathcal{E}$. Correspondingly, each instance $G_i = (\mathcal{V}_i, \mathcal{E}_i, \mathbf{X}_i)$. For the graph classification task, let $\mathcal{D} = \{G_i, \mathbf{y}_i\}_{i=1}^n$ denote the graph dataset, where $\mathbf{y}_i$ is the one-hot class label of graph G_i, and n is the total number of graphs.

Definition 3.1 (Graph Concept Bottleneck)**.** Given a dataset $\mathcal{D} = \{G_i, \mathbf{y}_i\}_{i=1}^n$ consisting of n input-target pairs, a graph concept bottleneck decomposes the classification process of GNNs into two stages: a backbone $\Phi : G \to \mathbb{R}^M$, which projects G into an M-dimensional concept space to generate the predicted concept label vector $\hat{\mathbf{c}} = \Phi(G) \in \mathbb{R}^M$; a linear classifier $f : \mathbb{R}^M \to \mathbb{R}^C$, which maps $\hat{\mathbf{c}}$ to the final prediction $\hat{\mathbf{y}} = f(\hat{\mathbf{c}}) \in \mathbb{R}^C$.

Notably, the prediction $\hat{\mathbf{y}}$ is entirely dependent on the input G through the bottleneck $\hat{\mathbf{c}}$, where each dimension of $\hat{\mathbf{c}}$ corresponds to a predefined concept. To build the graph concept bottleneck, a predefined concept set $\mathcal{T} = \{T_m\}_{m=1}^M$ is necessary. Each concept $\mathbf{t}_m$ describes a topologically meaningful subgraph of the samples in $\mathcal{D}$. For each input G_i, the ground-truth concept label $\mathbf{c}_i \in \mathbb{R}^M$ is computed by quantifying the relevance of G_i to all concepts. The dataset is then augmented with these concept labels as $\mathcal{D} = \{(G_i, \mathbf{c}_i, \mathbf{y}_i)\}_{i=1}^n$ for training both Φ and f. The main mathematical notations and definitions are listed in Appendix B.

4 Method: GCBMs

This section presents the construction pipeline of GCBMs. We elaborate on graph concept extraction, discriminative concept selection, and ground-truth concept label generation to build the graph concept bottleneck in Sec. 4.1. We further propose a concept embedding method to refine concept labels in Sec. 4.2. The overall architecture of this pipeline is illustrated in Figure 1.

4.1 Constructing Graph Concept Bottleneck

Graph Concept Extraction The message-passing mechanism in most GNNs is a continuous relaxation of the first-order WL graph isomorphism test [26]. Thus, we use WL-subtrees as graph concepts. Figure 2 shows an undirected and labeled graph and a WL-subtree of height 2 rooted at the node with label 1. The WL graph isomorphism test iteratively compares the neighborhood of each node between two graphs, and as a byproduct, the encoding of the WL-subtree is generated. In Figure 2, the WL-subtree encoding can be represented as "1,(2,13)(3,123)", which is an integration of the encodings of three WL-subtrees ("1,23", "2,13", "3,123") of height 1 rooted at the nodes with labels 1, 2, 3, respectively.

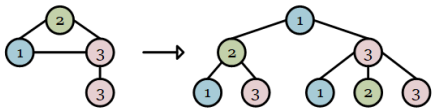

Figure 2: On the left side is a graph, and on the right side is its WL-subtree of height 2 rooted at the node with label 1.

For a graph dataset $\mathcal{D} = \{(G_i, \mathbf{y}_i)\}_{i=1}^n$, we first collect WL-subtrees of height k for all nodes in each graph and keep them in the graph concept corpus $\mathcal{C}^{(k)}$. Thus, $\mathcal{C}^{(k)}$ is a global-level corpus that spans all graphs in $\mathcal{D}$. Subsequently, we extract non-isomorphic WL-subtrees from $\mathcal{C}^{(k)}$ to form the graph

concept vocabulary $\mathcal{T}^{(k)} = \{T_1^{(k)}, \ldots, T_{|\mathcal{T}^{(k)}|}^{(k)}\}$, where all the non-isomorphic graph concepts are sorted lexicographically by their encodings.

Discriminative Concept Selection Since not all graph concepts in $\mathcal{T}^{(k)}$ are relevant for predicting graph class labels, we need to identify an effective graph concept set $\mathcal{T}_{pruned}^{(k)}$ from $\mathcal{T}^{(k)}$ to construct the graph concept bottleneck for GNNs. Specifically, $\mathcal{T}_{pruned}^{(k)}$ is selected by measuring the mutual information of each concept to the graph class label. Thus, we use information gain $IG(\mathcal{D}, T^{(k)})$ to quantify the relevance of graph concept $T^{(k)}$ to class label. For each graph G_i, we compute its frequency vector $\mathbf{x}_{G_i}^{(k)}$, where the j-th entry corresponds to the occurrence frequency of concept $T_j^{(k)}$ in G_i. For the dataset $\mathcal{D}$, we then construct a frequency matrix $\mathbf{X}^{(k)}$ with the i-th row $\mathbf{x}_{i,:}^{(k)}$ denoting the frequency vector of the i-th graph and the j-th column $\mathbf{x}_{:,j}^{(k)}$ denoting the occurrence frequency of the j-th graph concept across all graphs. The information gain for the j-th graph concept $T_j^{(k)}$ can be computed as:

$$IG(\mathcal{D}, T_j^{(k)}) = IG(\mathcal{D}, \mathbf{x}_{:,j}^{(k)}) = H(\mathcal{D}) - H(\mathcal{D}|\mathbf{x}_{:,j}^{(k)}), \tag{1}$$

where $H(\mathcal{D})$ is the entropy of the class distribution in $\mathcal{D}$, and $H(\mathcal{D}|\mathbf{x}_{:,j}^{(k)})$ is the conditional entropy of the class distribution given graph concept $T_j^{(k)}$. We select the top M graph concepts with the highest information gain to form the graph concept set $\mathcal{T}_{pruned}^{(k)}$.

Graph Concept Label Generation Each graph concept $T_j^{(k)} \in \mathcal{T}_{pruned}^{(k)}$ $(1 \leq j \leq M)$ can be represented as a one-hot vector $\mathbf{t}_j^{(k)}$ of length $|\mathcal{T}^{(k)}|$ with the entry on its position in $\mathcal{T}^{(k)}$ as 1 and others 0. We define the ground-truth concept label of graph G_i as the vector:

$$\mathbf{c}_i^{(k)} = [E(\mathbf{x}_{G_i}^{(k)}, \mathbf{t}_1^{(k)}), \ldots, E(\mathbf{x}_{G_i}^{(k)}, \mathbf{t}_M^{(k)})], \tag{2}$$

where $\mathbf{c}_i^{(k)} \in \mathbb{R}^M$ and its j-th concept score $E(\mathbf{x}_{G_i}^{(k)}, \mathbf{t}_j^{(k)})$ is the similarity between $\mathbf{x}_{G_i}^{(k)}$ and $\mathbf{t}_j^{(k)}$:

$$E(\mathbf{x}_{G_i}^{(k)}, \mathbf{t}_j^{(k)}) = \frac{\langle \mathbf{x}_{G_i}^{(k)}, \mathbf{t}_j^{(k)} \rangle}{\|\mathbf{x}_{G_i}^{(k)}\| \cdot \|\mathbf{t}_j^{(k)}\|}, \tag{3}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product, and $\| \cdot \|$ denotes the norm.

We concatenate all $\mathbf{c}_i^{(k)}$ $(1 \leq k \leq K)$ to obtain the global ground-truth graph concept label $\mathbf{c}_i = \text{Concat}(\mathbf{c}_i^{(1)}, \ldots, \mathbf{c}_i^{(K)})$, which integrates the scores of G_i across multi-height concepts. This concatenation guides the model to capture multi-scale topological patterns, and K is a tunable hyperparameter that enables adaptive structural modeling for different datasets.

Graph Concept Bottleneck Construction We adopt the K-layer graph isomorphism network (GIN) [59] as the GNN backbone Φ, as its discriminative power is theoretically equivalent to the first-order WL graph isomorphism test. We obtain the global graph embedding $\hat{\mathbf{c}}_i = \Phi(G_i)$ by concatenating the output of each GIN layer. Then, we align the hierarchical outputs with the global ground-truth graph concept labels. This alignment ensures each layer's output retains graph structural properties and can be interpreted as the extent to which the selected concepts exist in the graph.

To simultaneously optimize the graph concept learning and classification accuracy, GCBM's objective function comprises three components:

$$\mathcal{L} = \lambda_c \mathcal{L}_c + \mathcal{L}_y + \lambda_r R, \tag{4}$$

where $\lambda_c, \lambda_r > 0$ denote trade-off parameters that balance concept learning, classification performance, and concept sparsity. The concept loss $\mathcal{L}_c = \frac{1}{|\mathcal{D}|} \sum_{G_i \in \mathcal{D}} \|\Phi_{\Theta_1}(G_i) - \mathbf{c}_i\|^2$ facilitates alignment between graph embeddings and their concept labels. The cross-entropy classification loss $\mathcal{L}_y = \frac{1}{|\mathcal{D}|} \sum_{G_i \in \mathcal{D}} CE\left(f_{\Theta_2}(\Phi_{\Theta_1}(G_i)), \mathbf{y}_i\right)$ ensures consistency between predicted and true class labels. The sparsity regularization term $R = \mathcal{L}_1(\Theta_2)$ enhances the sparsity of the weight matrix Θ_2 of the concept bottleneck layer, where $\mathcal{L}_1(\cdot)$ denotes the L_1 regularization.

The above design endows GCBMs with explicit soft combinatorial reasoning. First, WL-subtrees serve as discrete combinatorial units that align with the structural patterns of graphs. Second, the

graph concept bottleneck acts as a differentiable soft logic learner, learning soft logical rules over the important graph concepts (i.e., their weights in the concept bottleneck layer are not 0). For formal definitions of soft Boolean logic, see Appendix C.

4.2 Context-Aware Graph Concept Embedding

In the above section, the ground-truth graph concept labels derived from the one-hot embeddings of concepts fail to capture correlations between concepts and their probability distributions in and across graphs, resulting in inaccurate concept supervision signals. In image and text domains, concepts exhibit inherent correlations, such as "bone spurs" and "intervertebral space" in images, or "history" and "archaeology" in text. Similarly, graph concepts may share analogous semantic correlations and co-occurrence patterns (e.g., the structural similarity between pyridine rings and pyrimidine rings in molecules), but orthogonal one-hot embeddings cannot capture such relational information.

To mitigate this issue, we analogize each graph concept as a "graph word" and draw inspiration from LMs. Such models capture correlations and co-occurrences between tokens ("graph words") through full-sequence context modeling. The analogy is supported by prior observations [36, 60]: graph substructure distributions follow a power law that is consistent with word distributions in natural languages. As shown in Figure 3, WL-subtree distributions in graphs approximate the power law distribution. Thus, LMs such as Transformer [52] can learn informative WL-subtree (graph concept) embeddings while modeling their correlations and co-occurrences.

Specifically, for graph G_i, we build its "graph sentence" $S_i^{(k)} = (T_{v_{\pi(1)}}^{(k)}, \ldots, T_{v_{\pi(|\mathcal{V}_i|)}}^{(k)})$, where each "graph word" $T_{v_{\pi(u)}}^{(k)} (1 \leq u \leq |\mathcal{V}_i|)$ is the WL-subtree of height k rooted at node $v_{\pi(u)}$, and the permutation function $\pi(\cdot)$ reorders node indices to sort all "graph words" lexicographically by the encodings of their WL-subtrees. For concept $T_j^{(k)} (1 \leq j \leq M)$, we also construct a "graph sentence" $S_{\mathcal{T}_{pruned}^{(k)}} = (T_1^{(k)}, \ldots, T_M^{(k)})$ based on $\mathcal{T}_{pruned}^{(k)}$, and the "graph words" are sorted in the same way as above. Then, for graph dataset $\mathcal{D} = \{(G_i, \mathbf{y}_i)\}_{i=1}^n$, we add all "graph sentences" at levels $k = 1, \ldots, K$ into a corpus $\mathcal{C}_{gs}$, and use the LM to learn dense embeddings of nodes and graph concepts:

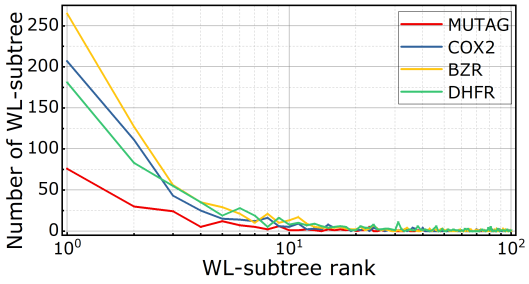

Figure 3: Distributions of the WL-subtrees (height $\leq$ 2) in graph datasets MUTAG, COX2, BZR, and DHFR. The x-axis denotes WL-subtree rank (i.e., the number of occurrences of a WL-subtree), and the y-axis denotes the number of WL-subtrees with the same rank.

$$\mathbf{h}_{T_{v_{\pi(u)}}^{(k)}} = \text{LM}(T_{v_{\pi(u)}}^{(k)}, \mathcal{C}_{gs}), \quad \mathbf{h}_{T_j^{(k)}} = \text{LM}(T_j^{(k)}, \mathcal{C}_{gs}), \tag{5}$$

where $\mathbf{h}_{T_{v_{\pi(u)}}^{(k)}} \in \mathbb{R}^d$ denotes the embedding of WL-subtree $T_{v_{\pi(u)}}^{(k)}$, which also serves as the embedding of node $v_{\pi(u)}$, and $\mathbf{h}_{T_j^{(k)}} \in \mathbb{R}^d$ denotes the embedding of concept $T_j^{(k)}$. The embedding $\mathbf{h}_{G_i}^{(k)}$ of G_i is obtained by aggregating all embeddings of WL-subtrees of height k rooted at each node in G_i:

$$\mathbf{h}_{G_i}^{(k)} = \sum_{T_{v_{\pi(u)}}^{(k)} \in S_{G_i}^{(k)}} \mathbf{h}_{T_{v_{\pi(u)}}^{(k)}}. \tag{6}$$

The graph concept label $\mathbf{c}_i^{(k)}$ in Equation (2) is updated by replacing $\mathbf{x}_{G_i}^{(k)}$ with the graph embedding $\mathbf{h}_{G_i}^{(k)}$ and $\mathbf{t}_j^{(k)}$ with the graph concept embedding $\mathbf{h}_{T_j^{(k)}}$.

Such context-aware embeddings produce continuous similarity-based supervision signals instead of binary labels, and capture correlations and co-occurrences between concepts, while mitigating the high-dimensional sparsity of one-hot vectors for nodes and concepts, as the embedding dimension d is much smaller than the dimension of one-hot vectors ($|\mathcal{T}^{(k)}|$). The refined concept labels better align with the intrinsic graph structure, providing a more effective concept bottleneck for GNNs. The generation process of the context-aware embedding-based graph concept labels is detailed in Algorithm 1, which is presented in Appendix D.

5 Experimental Evaluation

In this section, we verify the interpretability of the graph concept bottleneck through human evaluation and quantitative assessment, and validate the graph classification performance of GCBMs. We further analyze the contributions of core components via ablation studies, conduct parameter sensitivity analysis, and perform intervention experiments.

5.1 Experimental Setup

We conduct interpretability experiments on 2 specialized interpretation datasets, and perform classification experiments on 9 benchmark datasets and 3 large-scale class-imbalanced datasets. We select 5 groups of representative advanced methods as baselines: interpretable GNN models (Group 1), WL-based graph kernels (Group 2), optimal transport-based models (Group 3), mainstream GNN models (Group 4), and graph Transformer models (Group 5). Detailed dataset statistics, baselines, parameter settings, and hardware configurations are provided in Appendix F.

5.2 Experimental Results and Analysis

We denote the GCBM built with one-hot concept embeddings (Sec. 4.1) as GCBM, and its enhanced version equipped with Transformer-based dense concept embeddings (Sec. 4.2) as GCBM-E.

5.2.1 Interpretability Evaluation

To verify the interpretability of GCBMs, we conduct human evaluation and quantitative assessment.

Human Evaluation We perform human evaluation from two perspectives, taking the DHFR dataset as an example. First, we visualize the concept bottleneck layer weights of GCBM-E with a sankey diagram [33] in Figure 4. To observe the molecular substructures corresponding to these concepts in the sankey diagram, we map the nodes of the WL-subtrees (graph concepts) back to their original molecules and extract the relevant atoms and bonds, as shown in Figure 5. From Figure 4, we observe that graph concepts 20, 42, 47, and 59 show high weights across both classes, indicating their critical roles in classification. Molecules from different classes exhibit distinct probability distributions of concept labels for these cross-class concepts. GCBMs prioritize these concepts when making predictions. This visualization provides a clear global explanation that decodes how GCBMs make decisions through the linear combinatorial reasoning of these concepts.

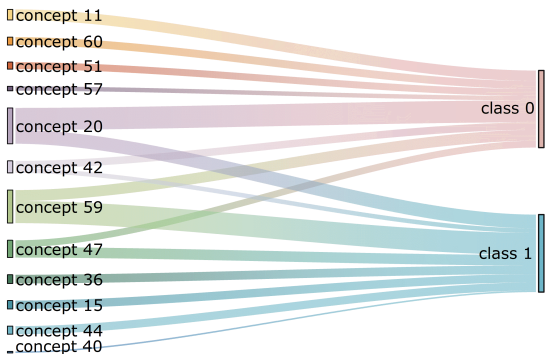

Figure 4: The sankey diagram of the graph concept bottleneck layer weights of GCBM-E on the DHFR dataset. The width of the connection between each concept and class corresponds to the exponent of the absolute value of the learned concept weight. To highlight core concepts, only the top 8 concepts with the largest absolute values of weights in each class are shown.

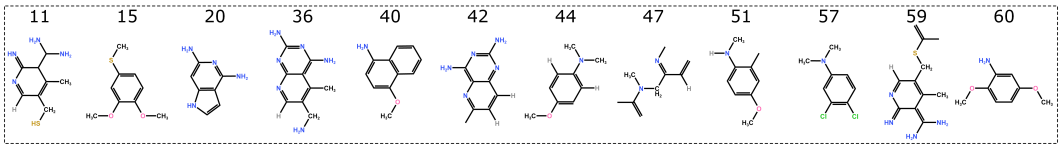

Figure 5: Concept indices and corresponding specific substructures in DHFR molecules.

Second, we verify that these discriminative concepts correspond to human-understandable semantic information. Specifically, we restore the concepts to their original graph structures and validate their consistency with domain knowledge. In Figure 6, we present the graph structures of sample molecules from two classes that contain concepts 20 and 42. As observed in Figure 6, concept 20 corresponds to a purine ring structure and concept 42 to a pteridine ring structure. Both substructures have been confirmed by biochemical studies as key active functional groups of dihydrofolate reductase

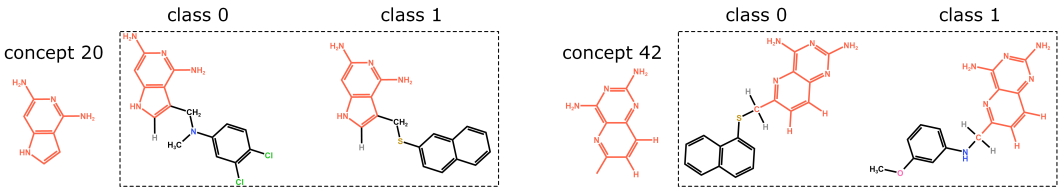

Figure 6: Key concepts and sample molecules containing these concepts for each class in the DHFR dataset. The red region highlights the substructure corresponding to each concept in the molecule.

inhibitors [46, 50, 37, 20], and are highly relevant to the target task labels. These results confirm that GCBMs can identify meaningful graph concepts consistent with real-world domain knowledge through concept selection, verifying the interpretability of the graph concept bottleneck.

Quantitative Assessment We conduct quantitative evaluations on 2 interpretation datasets, Solubility and Benzene, which are annotated with true interpretable subgraphs. We adopt the AUC score [55, 28] as the interpretability metric, which measures the alignment between ground-truth and predicted interpretable subgraphs. Specifically, we utilize the concept selection method of GCBMs to filter the top $M = 2$ key concepts from the graph concept vocabulary $\mathcal{T}^{(k)}$. These concepts are mapped back to each original graph to obtain their specific substructures, and the union of nodes from all extracted substructures is regarded as the interpretable subgraph predicted by GCBMs for each graph. Table 1 reports the AUC scores of GCBMs on these datasets.

Table 1: AUC scores (mean%) of each method on Solubility and Benzene for interpretability assessment. The selected M concepts are shared for GCBM and GCBM-E, leading to the same results. Green Yellow Red mean the best, the second-best, and the third-best result, respectively (This notation applies to all subsequent tables).

Dataset	Solubility	Benzene
GNNExplainer	51.8	54.7
PGExplainer	77.9	48.2
ProtGNN	56.2	66.4
VGIB	67.5	62.2
PGIB	65.0	61.3
ConfExplainer	80.5	75.3
GraphTrail	82.6	74.9
LogiX	82.1	81.4
LogicXGNN	83.5	83.2
LogiX-GIN	84.0	82.5
DC-GNN	83.6	80.5
GCBM/GCBM-E	91.2	83.6

The results show that GCBMs achieve higher AUC scores, confirming that the concepts used to construct the explicit concept bottleneck can accurately correspond to the interpretable substructures. In particular, unlike baselines, GCBMs do not require complex training processes for subgraph prediction, leading to stable AUC scores with no standard deviation. Its key concept extraction and subgraph mapping are achieved solely by adjusting the hyperparameter M, greatly reducing the computational cost. The parameter sensitivity analysis of M is provided in Appendix H. Moreover, we evaluate the explanation faithfulness of GCBMs in Appendix G.

5.2.2 Graph Classification Performance

To verify the generalization ability of GCBMs on graph classification, we report the classification accuracy of the two GCBM versions on 9 datasets of various domains in Table 2, and present GCBM-E's AUC scores on 3 large-scale class-imbalanced datasets in Table 3.

Table 3: AUC scores (mean%±standard deviation%) of 10-fold cross-validation of each method on class-imbalanced benchmark datasets (only results of methods that can complete the experiments in 24 hours are reported).

Dataset	PC-3	MCF-7	OGBG-MOLHIV
GNNExplainer	84.1±1.6	79.7±1.5	78.1±1.4
ConfExplainer	83.7±1.5	79.2±1.4	77.3±1.4
GIN	84.6±1.4	80.6±1.5	77.8±1.3
GRDL	85.1±1.6	81.4±1.3	79.8±1.0
GCBM-E	88.8±1.7	86.2±1.3	81.0±1.1

Table 2 shows that GCBM-E achieves top-2 performance on 8 out of 9 datasets and obtains the highest accuracy on 6 of them. Compared with GCBM-E, GCBM uses simple one-hot vectors as graph concept embeddings to generate concept labels. Its average accuracy decreases by 0.9%. This is because one-hot vectors fail to capture contextual differences of the same WL-subtree across different graphs, and ignore correlations and co-occurrences between concepts, leading to degraded quality of concept labels. Additionally, their high sparsity hinders effective graph embedding learning. Note that since post-hoc explainers cannot make forward predictions independently, the classification accuracy reported for all post-hoc explainers is

Table 2: Classification accuracy (mean%±standard deviation%) of GCBMs and baseline models on benchmark datasets.

	Dataset	MUTAG	PTC-MR	BZR	COX2	DHFR	IMDB-B	IMDB-M	COLLAB	DBLP_v1
Group 1	GNNExplainer	66.5±2.3	67.1±6.2	81.1±10.4	80.7±6.6	69.8±7.2	70.4±4.2	45.0±4.3	77.6±5.0	82.8±0.4
	PGExplainer	78.6±11.4	67.0±5.5	82.9±2.7	72.2±2.7	69.0±6.4	62.2±4.0	40.0±6.5	75.7±4.5	81.2±0.5
	ProtGNN	77.6±4.7	68.2±5.8	82.5±2.5	79.2±3.1	70.8±2.3	58.3±7.3	36.0±5.7	69.3±8.6	80.0±0.6
	VGIB	72.1±21.6	66.5±6.1	79.8±3.9	80.0±4.2	51.7±10.5	50.8±3.7	33.9±3.1	72.3±4.8	80.5±0.8
	PGIB	76.6±6.8	66.9±5.4	83.7±5.8	75.8±2.3	77.4±4.1	57.0±2.9	37.0±3.9	73.2±4.9	81.3±0.7
	ConfExplainer	82.5±6.7	66.8±6.1	83.1±5.2	75.5±3.8	71.8±8.2	65.6±5.4	44.9±4.5	77.5±5.2	82.7±0.6
	GraphTrail	87.4±5.0	69.3±5.9	86.7±4.7	83.3±4.9	77.2±3.6	72.9±4.9	46.5±2.1	67.3±4.2	79.2±0.5
	LogiX	82.6±7.1	66.5±4.7	83.1±4.8	81.4±4.5	75.2±6.1	63.5±4.1	48.2±3.1	79.8±3.8	80.8±0.6
	LogicXGNN	81.1±4.2	67.2±3.9	83.4±4.2	82.0±3.8	76.3±4.5	64.2±4.1	49.9±2.8	76.2±4.4	81.4±0.5
	LogiX-GIN	82.6±5.6	68.0±3.3	84.7±3.6	82.1±3.6	75.8±3.4	64.7±3.9	51.8±2.5	76.5±2.5	81.2±0.4
	DC-GNN	80.0±2.0	67.6±4.0	84.6±4.2	80.8±4.1	76.7±4.3	64.6±4.0	49.4±3.0	77.3±4.7	80.9±0.7
Group 2	WL	87.4±5.4	56.0±3.9	81.3±0.6	81.2±1.1	82.4±0.9	67.5±4.0	48.4±4.2	78.5±1.7	77.6±0.6
	WWL	86.3±7.9	52.6±6.8	87.6±0.6	82.7±3.0	82.7±3.6	71.6±3.8	52.6±3.0	81.4±2.1	81.8±0.6
	FWL	85.7±7.5	59.3±6.5	87.6±5.0	82.0±2.7	82.5±4.0	72.0±4.6	49.5±2.6	78.1±1.2	81.9±0.8
	GAWL	87.3±6.3	61.6±4.6	88.3±3.8	79.9±4.9	83.1±4.0	74.2±4.7	50.7±5.4	81.5±2.0	81.1±0.3
Group 3	OT-GNN	91.6±4.6	68.0±7.5	85.9±3.3	83.5±3.6	84.8±3.1	67.5±3.5	52.1±3.0	80.7±2.9	85.4±0.3
	WEGL	91.0±3.4	66.2±6.9	84.4±4.6	81.8±1.8	82.2±2.9	66.4±2.1	50.3±1.0	79.6±0.5	80.4±0.4
Group 4	GIN	89.4±5.6	64.6±7.0	82.6±3.5	81.6±4.9	82.6±2.6	64.3±3.1	50.9±1.7	79.3±1.7	83.3±0.4
	DROPGIN	90.4±7.0	66.3±8.6	77.8±2.6	82.1±2.7	82.7±3.3	66.3±4.5	51.6±3.2	80.1±2.8	83.0±0.9
	SEP	89.4±6.1	68.5±6.1	86.9±0.8	83.7±1.8	84.1±1.2	74.1±0.6	51.5±0.7	81.3±0.2	85.5±0.3
	GMT	89.9±4.2	70.2±6.2	85.6±0.8	84.2±2.0	83.9±0.9	73.5±0.8	50.7±0.8	80.7±0.5	85.2±0.6
	MINCUTPOOL	90.6±4.6	68.3±4.4	87.2±1.0	83.5±2.7	82.2±0.9	72.7±0.8	51.0±0.7	80.9±0.3	84.9±0.5
	ASAP	87.4±5.7	64.6±6.8	85.3±1.3	81.1±3.1	83.3±1.6	72.8±0.5	50.8±0.8	78.6±0.5	82.6±0.6
	WITTOPOPOOL	89.4±5.4	64.6±6.8	87.8±2.4	83.7±3.6	84.5±2.8	72.6±1.8	52.9±0.8	80.1±1.6	85.6±0.8
	GRDL	92.1±5.9	68.3±5.4	92.0±1.1	85.9±2.2	85.1±2.3	74.8±2.0	52.9±1.8	79.8±0.9	85.3±0.5
Group 5	GRAPHORMER	89.6±6.2	71.4±5.2	85.3±2.3	84.0±2.9	83.7±2.1	70.3±0.9	48.9±2.0	80.3±1.3	84.8±0.6
	SAT	92.6±4.3	68.3±4.9	91.7±2.1	84.1±2.6	84.4±3.4	70.0±1.3	47.3±3.2	80.6±0.6	85.9±0.3
Ours	GCBM	93.5±4.0	68.5±5.8	89.7±4.2	84.3±3.5	85.7±3.6	74.2±4.1	52.5±3.1	82.4±1.6	86.3±0.7
	GCBM-E	93.6±4.0	70.6±5.4	90.4±2.8	85.9±3.2	86.1±3.2	75.4±4.3	53.1±2.4	82.8±1.4	87.6±0.5

taken from their default backbones (i.e., multi-layer GNNs as specified in their original papers and official implementations), to objectively reflect the practical performance of the "prediction model and post-hoc explanation" pipeline. Table 3 shows that on 3 large-scale class-imbalanced graphs, GCBM-E significantly outperforms all baseline methods. In summary, GCBMs achieve excellent performance on multiple datasets, especially in class-imbalanced scenarios.

Detailed analysis of running time and computational complexity is presented in Appendix E. The results show that both GCBM and GCBM-E achieve a good balance between computational cost and classification accuracy.

5.2.3 Ablation Studies

To validate the effectiveness of the key components in GCBMs, we construct two variants of GCBM (GCBM-CP and GCBM-CC) and eight variants of GCBM-E (the rest) by individually replacing or removing each component, and compare their performance across 4 datasets, as shown in Table 4.

Table 4: Classification accuracy (mean%±standard deviation%) of GCBM-E and its variants on benchmark datasets.

Dataset	MUTAG	COX2	DHFR	IMDB-M
GCBM-CP	89.2±4.1	82.3±3.1	84.2±3.3	50.9±3.0
GCBM-CC	90.7±4.5	82.9±3.6	84.6±3.0	50.4±3.5
GCBM-$\mathcal{T}$	92.5±4.3	85.0±2.5	85.7±3.3	53.0±3.3
GCBM-PT	93.1±3.6	85.7±3.1	85.8±3.3	52.9±2.5
GCBM-GCN	87.7±6.2	85.0±3.4	84.2±2.4	52.2±3.4
GCBM-Qwen	92.4±3.5	85.5±3.6	85.8±3.4	53.3±2.7
GCBM-w/o PE	92.8±4.2	84.2±2.9	85.4±2.1	51.1±2.9
GCBM-w/o L	92.5±3.8	84.7±2.4	85.6±3.4	52.4±3.0
GCBM-w/o $\mathcal{L}_c$	90.1±4.0	84.5±3.3	84.9±3.1	51.8±3.5
GCBM-w/o R	91.4±4.5	85.2±3.2	85.5±3.7	52.7±2.6
GCBM	93.5±4.0	84.3±3.5	85.7±3.6	52.5±3.1
GCBM-E	93.6±4.0	85.9±3.2	86.1±3.2	53.1±2.4

GCBM-CP adopts concept presences to construct $\mathbf{x}_{G_i}^{(k)}$, where the j-th entry of $\mathbf{x}_{G_i}^{(k)}$ is set to 1 if concept $T_j^{(k)}$ exists in graph G_i, and 0 otherwise. GCBM-CC uses concept counts to formulate $\mathbf{x}_{G_i}^{(k)}$, where the j-th entry is the occurrence count of concept $T_j^{(k)}$ in G_i. The original GCBM constructs $\mathbf{x}_{G_i}^{(k)}$ based on concept frequencies and consistently achieves the best performance across all datasets. This result demonstrates that normalized concept frequencies provide smooth and scale-invariant supervision signals for GNN concept learning.

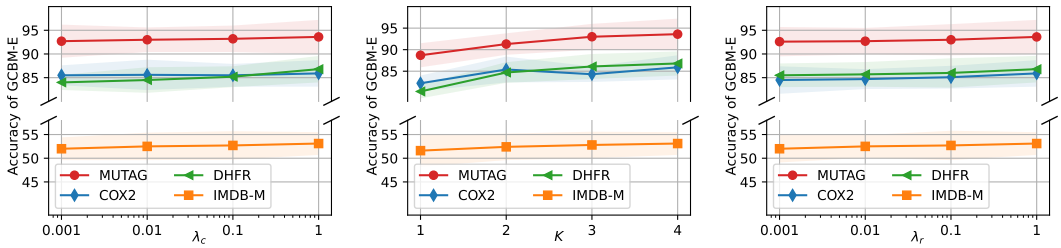

Figure 7: Parameter sensitivity analysis of GCBM-E: λ_c (graph concept loss weight), K (the maximum height of the WL-subtrees), and λ_r (sparse regularization weight).

GCBM-$\mathcal{T}$ replaces the highly discriminative concept set $\mathcal{T}^{(k)}_{pruned}$ with M randomly selected concepts. Its accuracy decreases across all datasets, since randomly chosen concepts yield low information gain and provide limited discriminative power. GCBM-PT replaces WL-subtrees with path-tree structures [29]. Its performance is worse than GCBM-E on all the datasets. GCBM-GCN and GCBM-Qwen replace the backbone GIN with GCN [23] and the graph concept embedding model Transformer with the pre-trained large language model Qwen-1.8B [7] after fine-tuning, respectively. The performance declines on most datasets. However, GCBM-Qwen outperforms GCBM-E slightly on IMDB-M. GCBM-w/o PE removes positional encodings during the LM pre-training phase and treats "graph sentences" as entirely unordered bags of words, while GCBM-w/o L eliminates the lexicographical ordering of "graph words" within each "graph sentence". Both variants yield performance degradation. This is because sorting the WL-subtree encodings of concepts lexicographically and leveraging the positional encodings of the LM enables topologically similar concepts to cluster at adjacent sequence positions, and ordered topological context modeling allows the LM to extract richer structural information. GCBM-w/o $\mathcal{L}_c$ and GCBM-w/o R remove the concept loss and sparsity regularization from GCBM-E, respectively. The former impairs the alignment between graph embeddings and ground-truth concept labels, weakening the supervision of the concept bottleneck. The latter relaxes the constraint on the concept bottleneck layer, resulting in redundant concept weights. Both variants lead to significant clear accuracy drops.

In summary, all key components of GCBMs contribute significantly to its performance. Furthermore, to verify the generality of the GCBM framework, we conduct additional ablation experiments by replacing different backbones, graph substructures, and LMs. Details in Appendix I.

5.2.4 Sensitivity Analysis

In this section, we evaluate the sensitivity of GCBM-E's performance to its parameters λ_c (graph concept loss weight), K (the maximum height of the WL-subtrees), and λ_r (sparse regularization weight) on the same 4 datasets as the above section. Figure 7 demonstrates the performance changes of GCBM-E when adjusting its key parameters. Specifically, decreasing λ_c or λ_r reduces the model's accuracy, while decreasing K leads to a more significant performance drop. Compared to weakening the alignment between graph representations and concept labels (reducing λ_c) or relaxing the constraints of sparse regularization (reducing λ_r), pruning the concept vocabulary (decreasing K) fundamentally undermines the quality of the concept bottleneck, thus causing more severe performance impairment. Overall, reducing λ_c, K, or λ_r consistently degrades GCBM-E's performance across all datasets. A detailed analysis of the model's sensitivity to K and extended experimental evaluations are provided in Appendix J.

5.2.5 Intervention Experiments

To verify the intervenability of the concept layer in GCBMs, we adopt different intervention strategies for GCBM and GCBM-E, and conduct experiments on their misclassified graphs in the DHFR dataset. Specifically, GCBM's concept score for graph G_i on concept T_j relies on G_i's frequency-based embedding and the one-hot vector of T_j. For misclassified graphs, we directly edit their predicted concept vectors $\hat{\mathbf{c}}_i$ [24] and feed the modified concept vectors into the classification layer to re-predict the class, thus correcting the misclassified graphs. For GCBM-E, concept scores are computed via the similarity of Transformer-learned embeddings, and the associations between concepts and classes are measured by the concept bottleneck layer weights. For graph G_i satisfying $\hat{\mathbf{y}}_i \neq \mathbf{y}_i$ and $\text{sim}(\hat{\mathbf{c}}_i, \mathbf{c}_i) \geq \tau_c$ (i.e., valid concept activation but incorrect classification [33]), where τ_c denotes the

Table 5: Intervention results of GCBM and GCBM-E on the DHFR dataset. For simplicity, W_F denotes the concept bottleneck layer weight matrix in intervention experiments, and $W_{F[\text{cls,cpt}]}$ is the weight of the target concept (cpt) corresponding to the class (cls). NMR indicates no modification required. Accuracy represents the classification accuracy (mean%) before and after intervention.

Model	Intervention Target	$W_{F[\text{cls 0,cpt j}]}$	$W_{F[\text{cls 1,cpt j}]}$	Impact	Accuracy
GCBM	$\hat{\mathbf{c}}_i$ of misclassified samples	NMR	NMR	+12 corrections	85.7 → 87.3
GCBM-E	$W_{F[\text{cls 0,cpt 20}]}$, $W_{F[\text{cls 1,cpt 20}]}$	0.384 → -0.037	-0.011 → 0.410	+11 corrections; +6 new errors	86.1 → 86.8
GCBM-E	$W_{F[\text{cls 0,cpt 42}]}$, $W_{F[\text{cls 1,cpt 42}]}$	0.188 → -0.247	0.094 → 0.528	+8 corrections; +5 new errors	86.1 → 86.5
GCBM-E	$W_{F[\text{cls 0,cpt 20/42}]}$, $W_{F[\text{cls 1,cpt 20/42}]}$	0.384/0.188 → -0.037/-0.247	-0.011/0.094 → 0.410/0.528	+15 corrections; +8 new errors	86.1 → 87.0

similarity threshold for valid concept activation, we design three intervention schemes: modifying the concept layer weights of discriminative concept 20 alone, modifying those of concept 42 alone, and performing joint intervention on both concepts explore the synergistic effect of key concepts. Detailed modification rules are provided in Appendix K, and intervention results are shown in Table 5.

The results show that GCBM's classification accuracy increases from 85.7% to 87.3%, with the correction of misclassified graphs achieving a net improvement. For GCBM-E, all three intervention schemes enhance overall performance, and the joint intervention exhibits the most significant effect: it corrects 15 misclassified graphs (outperforming the sum of individual interventions on each concept) with only 8 new misclassifications, boosting accuracy from 86.1% to 87.0%. This superior performance validates the combinatorial reasoning of GCBMs. Key concepts do not act independently but synergistically contribute to the model's decision-making. These intervention experiments confirm GCBMs' advantages in interpretability and decision controllability, enabling targeted optimization of prediction results through intervention on core concept combinations.

6 Conclusion

This paper proposes a globally and intrinsically interpretable paradigm for GNNs called Graph Concept Bottleneck Models (GCBMs). As a generic framework, the explicit graph concept bottleneck of GCBMs enforces the combinatorial reasoning of GNNs' predictions to fit the soft logical rule over graph concepts, achieving both excellent interpretability and discriminative power. Human evaluation of molecular graphs confirms that the concepts extracted by GCBMs are highly aligned with domain knowledge, while quantitative assessment demonstrates that these concepts accurately correspond to interpretable subgraphs in graphs. Experiments on multiple datasets verify the superior predictive performance of GCBMs, with notable advantages on large-scale class-imbalanced datasets. Moreover, intervention experiments validate the controllability of GCBMs' decision-making process. In the future, we plan to expand graph concept types using large language models and explore more advanced backbones to further enhance the flexibility and applicability of GCBMs.

Acknowledgments

We thank the anonymous reviewers for their valuable and constructive comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62576253, in part by the National Key R&D Program of China under Grant 2025YFE0202900, in part by Science and Technology Development Fund (FDCT-MOST), Macao SAR under Grant 0007/2025/AMJ, in part by AI for Science Program, Shanghai Municipal Commission of Economy and Informatization under Grant 2025-GLZ-RGZN-BTBX-02016, and in part by the Fundamental Research Funds for the Central Universities.

References

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[2] Emily Alsentzer, Samuel Finlayson, Michelle Li, and Marinka Zitnik. Subgraph neural networks. *Advances in Neural Information Processing Systems*, 33:8017–8029, 2020.

[3] Burouj Armgaan, Manthan Dalmia, Sourav Medya, and Sayan Ranu. Graphtrail: Translating gnn predictions into human-interpretable logical rules. *Advances in Neural Information Processing Systems*, 37:123443–123470, 2024.

[4] Steve Azzolin, Antonio Longa, Pietro Barbiero, Pietro Lio, and Andrea Passerini. Global explainability of gnns via logic combination of learned concepts. In *The Eleventh International Conference on Learning Representations*.

[5] Steve Azzolin, Sagar Malhotra, Andrea Passerini, and Stefano Teso. Beyond topological self-explainable gnns: A formal explainability perspective. *arXiv preprint arXiv:2502.02719*, 2025.

[6] Jinheon Baek, Minki Kang, and Sung Ju Hwang. Accurate learning of graph representations with graph multiset pooling. *arXiv preprint arXiv:2102.11533*, 2021.

[7] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[8] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *International conference on machine learning*, pages 874–883. PMLR, 2020.

[9] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-yan Liu, and Liwei Wang. Graphnorm: A principled approach to accelerating graph neural network training. In *International Conference on Machine Learning*, pages 1204–1215. PMLR, 2021.

[10] Benson Chen, Gary Bécigneul, Octavian-Eugen Ganea, Regina Barzilay, and Tommi Jaakkola. Optimal transport graph neural networks. *arXiv preprint arXiv:2006.04804*, 2020.

[11] Dexiong Chen, Leslie O'Bray, and Karsten Borgwardt. Structure-aware transformer for graph representation learning. In *International conference on machine learning*, pages 3469–3489. PMLR, 2022.

[12] Yuzhou Chen and Yulia R Gel. Topological pooling on graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7096–7103, 2023.

[13] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 257–266, 2019.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[15] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

[16] Chuqin Geng, Ziyu Zhao, Zhaoyue Wang, Haolin Ye, Yuhe Jiang, and Xujie Si. Logicxgnn: Grounded logical rules for explaining graph neural networks. In *International Conference on Learning Representations*, volume 2026, pages 4381–4412, 2026.

[17] Dobrik Georgiev, Pietro Barbiero, Dmitry Kazhdan, Petar Veličković, and Pietro Liò. Algorithmic concept-based explainable reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6685–6693, 2022.

[18] Nikolentzos Giannis and Vazirgiannis Michalis. Graph alignment kernels using weisfeiler and leman hierarchies. In *The 26th International Conference on Artificial Intelligence and Statistics*, 2023.

[19] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[20] Marie Hoarau, Patpanat Sermmai, Thaveechai Varatthan, Ratthiya Thiabma, Tararat Jantra, Roonglawan Rattanajak, Danoo Vitsupakorn, Jarunee Vanichtanankul, Siriporn Saepua, Yongyuth Yuthavong, et al. Discovery of rigid biphenyl plasmodium falciparum dhfr inhibitors using a fragment linking strategy. *RSC Medicinal Chemistry*, 14(9):1755–1766, 2023.

[21] John T Kent. Information gain and a general measure of correlation. *Biometrika*, 70(1):163–173, 1983.

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[23] Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[24] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR, 2020.

[25] Soheil Kolouri, Navid Naderializadeh, Gustavo K Rohde, and Heiko Hoffmann. Wasserstein embedding for graph learning. *arXiv preprint arXiv:2006.09430*, 2020.

[26] Andrei Leman and Boris Weisfeiler. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya*, 2(9):12–16, 1968.

[27] Josh Magnus Ludan, Qing Lyu, Yue Yang, Liam Dugan, Mark Yatskar, and Chris Callison-Burch. Interpretable-by-design text classification with iteratively generated concept bottleneck. *arXiv preprint arXiv:2310.19660*, 2023.

[28] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

[29] Gaspard Michel, Giannis Nikolentzos, Johannes F Lutzeyer, and Michalis Vazirgiannis. Path neural networks: Expressive and accurate graph neural networks. In *International Conference on Machine Learning*, pages 24737–24755. PMLR, 2023.

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[31] Marvin Minsky and Seymour Papert. An introduction to computational geometry. *Cambridge tiass., HIT*, 479(480):104, 1969.

[32] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020.

[33] Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. *arXiv preprint arXiv:2304.06129*, 2023.

[34] Pál András Papp, Karolis Martinkus, Lukas Faber, and Roger Wattenhofer. Dropgnn: Random dropouts increase the expressiveness of graph neural networks. *Advances in Neural Information Processing Systems*, 34:21997–22009, 2021.

[35] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[36] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 701–710. ACM, 2014.

[37] Katharina Possart, Fabian C Herrmann, Joachim Jose, and Thomas J Schmidt. In silico and in vitro search for dual inhibitors of the trypanosoma brucei and leishmania major pteridine reductase 1 and dihydrofolate reductase. *Molecules*, 28(22):7526, 2023.

[38] Alessio Ragno, Marc Plantevit, and Céline Robardet. Faithful explanations for graph classification using logic. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 113–129. Springer, 2025.

[39] Alessio Ragno, Marc Plantevit, and Céline Robardet. On logic-based self-explainable graph neural networks. *Advances in Neural Information Processing Systems*, 38:116138–116166, 2026.

[40] Ekagra Ranjan, Soumya Sanyal, and Partha Talukdar. Asap: Adaptive structure aware pooling for learning hierarchical graph representations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5470–5477, 2020.

[41] Jiahua Rao, Hanjing Lin, Jiancong Xie, Zhen Wang, Shuangjia Zheng, and Yuedong Yang. Incorporating retrieval-based causal learning with information bottlenecks for interpretable molecular graph learning. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 2398–2409, 2025.

[42] Benjamin Sanchez-Lengeling, Jennifer Wei, Brian Lee, Emily Reif, Peter Wang, Wesley Qian, Kevin McCloskey, Lucy Colwell, and Alexander Wiltschko. Evaluating attribution for graph neural networks. *Advances in neural information processing systems*, 33:5898–5910, 2020.

[43] Till Schulz, Pascal Welke, and Stefan Wrobel. Graph filtration kernels. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8196–8203, 2022.

[44] Sangwoo Seo, Sungwon Kim, and Chanyoung Park. Interpretable prototype-based graph information bottleneck. *Advances in Neural Information Processing Systems*, 36:76737–76748, 2023.

[45] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

[46] Bharath Srinivasan and Jeffrey Skolnick. Insights into the slow-onset tight-binding inhibition of escherichia coli dihydrofolate reductase: detailed mechanistic characterization of pyrrolo [3, 2-f] quinazoline-1, 3-diamine and its derivatives as novel tight-binding inhibitors. *The FEBS journal*, 282(10):1922–1938, 2015.

[47] Divyansh Srivastava, Ge Yan, and Lily Weng. Vlg-cbm: Training concept bottleneck models with vision-language guidance. *Advances in Neural Information Processing Systems*, 37:79057–79094, 2024.

[48] Chung-En Sun, Tuomas Oikarinen, Berk Ustun, and Tsui-Wei Weng. Concept bottleneck large language models. *arXiv preprint arXiv:2412.07992*, 2024.

[49] Zhen Tan, Lu Cheng, Song Wang, Bo Yuan, Jundong Li, and Huan Liu. Interpreting pretrained language models via concept bottlenecks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 56–74. Springer, 2024.

[50] Giusy Tassone, Giacomo Landi, Pasquale Linciano, Valeria Francesconi, Michele Tonelli, Lorenzo Tagliazucchi, Maria Paola Costi, Stefano Mangani, and Cecilia Pozzi. Evidence of pyrimethamine and cycloguanil analogues as dual inhibitors of trypanosoma brucei pteridine reductase and dihydrofolate reductase. *Pharmaceuticals*, 14(7):636, 2021.

[51] Matteo Togninalli, Elisabetta Ghisu, Felipe Llinares-López, Bastian Rieck, and Karsten Borgwardt. Wasserstein weisfeiler-lehman graph kernels. *Advances in neural information processing systems*, 32, 2019.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[53] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

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

[55] Zixiao Wang and Jicong Fan. Graph classification via reference distribution learning: theory and practice. *Advances in Neural Information Processing Systems*, 37:137698–137740, 2024.

[56] Junran Wu, Xueyuan Chen, Ke Xu, and Shangzhe Li. Structural entropy guided graph hierarchical pooling. In *International conference on machine learning*, pages 24017–24030. PMLR, 2022.

[57] Yaochen Xie, Sumeet Katariya, Xianfeng Tang, Edward Huang, Nikhil Rao, Karthik Subbian, and Shuiwang Ji. Task-agnostic graph explanations. *Advances in neural information processing systems*, 35:12027–12039, 2022.

[58] Haotian Xu, Tsui-Wei Weng, Lam M Nguyen, and Tengfei Ma. Graph concept bottleneck models. *arXiv preprint arXiv:2508.14255*, 2025.

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

[60] Pinar Yanardag and SVN Vishwanathan. Deep graph kernels. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1365–1374. ACM, 2015.

[61] Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19187–19197, 2023.

[62] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021.

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

[64] Junchi Yu, Jie Cao, and Ran He. Improving subgraph recognition with variational graph information bottleneck. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19396–19405, 2022.

[65] Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Graph information bottleneck for subgraph recognition. In *International Conference on Learning Representations*.

[66] Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Graph information bottleneck for subgraph recognition. *arXiv preprint arXiv:2010.05563*, 2020.

[67] 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*, pages 12241–12252. PMLR, 2021.

[68] Jiaxing Zhang, Xiaoou Liu, Dongsheng Luo, and Hua Wei. Is your explanation reliable: Confidence-aware explanation on graph neural networks. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 3740–3751, 2025.

[69] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[70] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Cheekong Lee. Protgnn: Towards self-explaining graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 9127–9135, 2022.

A Limitations

Although GCBMs show superiority in interpretability and classification, they have some limitations that do not affect our core contributions. GCBM-E yields the best performance but incurs higher computational complexity. However, this is not intractable since many works [13, 9] have tried to address it. In addition, we have provided a solution in Appendix E to further reduce the training time complexity of the Transformer for learning the graph concept embeddings. Furthermore, concept leakage and shortcut learning are pervasive inherent challenges for CBMs [24, 33]. For GCBMs, we do not claim to have entirely eliminated these fundamental challenges; nevertheless, our model architecture introduces dedicated structural constraints and empirical validation mechanisms that substantially mitigate their effects. An in-depth discussion is provided in Appendix L.

B Notation List

The main mathematical notations and definitions are listed as follows.

Notations	**Definition**
$G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$	Undirected labeled graph
$\lvert\mathcal{V}\rvert$	Number of nodes in graph G
$\mathcal{D} = \{G_i, \mathbf{y}_i\}_{i=1}^n$	Graph dataset
n	Number of graphs
Φ	Backbone
$\hat{\mathbf{c}}_i = \Phi(G_i)$	Predicted concept label of G_i
$\mathbf{c}_i$	Ground-truth concept label of G_i
M	Number of selected discriminative concepts
$T_j^{(k)}$	j-th WL-subtree concept of height k
K	Maximum height of WL-subtrees
$\mathcal{T}^{(k)}$	Graph concept vocabulary of height k
$\mathcal{T}_{pruned}^{(k)}$	Graph concept set of height k
$IG(\mathcal{D}, T_j^{(k)})$	Information gain for concept $T_j^{(k)}$
$\mathbf{x}_{G_i}^{(k)}$	Concept frequency vector of height k of G_i
$\mathbf{t}_j^{(k)}$	One-hot encoding of concept $T_j^{(k)}$
$E(\cdot, \cdot)$	Cosine similarity
f	Sparse linear classifier
$\hat{\mathbf{y}}_i = f(\hat{\mathbf{c}}_i)$	Predicted class label of G_i
$\mathcal{L}$	Overall loss
$\mathcal{L}_c$	Concept loss
$\mathcal{L}_y$	Classification loss
R	L_1 sparsity regularization
λ_c, λ_r	Loss trade-off parameters
Θ_1, Θ_2	Parameters of Φ and f
$\mathbf{h}_{T_j^{(k)}}$	Context-aware embedding of concept $T_j^{(k)}$
$\mathbf{h}_{T_{v_{\pi(u)}}^{(k)}}$	Node embedding of height k of node $v_{\pi(u)}$
$\mathbf{h}_{G_i}^{(k)}$	Graph embedding of height k of G_i
$S_i^{(k)}$	Graph sentence of G_i at height k
$S_{\mathcal{T}_{pruned}^{(k)}}$	Graph sentence of $\mathcal{T}_{pruned}^{(k)}$
$\mathcal{C}_{gs}$	Global graph sentence corpus

C Soft Combinatorial Reasoning via Graph Concept Bottleneck

We formally define the soft combinatorial reasoning realized by the concept bottleneck layer in GCBMs. The final prediction for each class is given by:

$$\hat{y}_c = \sigma(\sum_{j=1}^{M} w_{c,j} \hat{E}(\mathbf{h}_G, \mathbf{h}_{T_j}) + b_c), \tag{7}$$

where $\hat{y}_c$ denotes the predicted probability for the c-th class, $\hat{E}(\mathbf{h}_G, \mathbf{h}_{T_j}) \in [0, 1]$ denotes the predicted concept activation score of the j-th graph concept T_j for the input graph G, $\mathbf{h}_G$ denotes the embedding of G, $\mathbf{h}_{T_j}$ denotes the embedding of T_j, $w_{c,j}$ denotes the weight of the concept T_j for the c-th class in the concept bottleneck layer, σ denotes the activation function, and b_c is the corresponding bias term for the class. The sparsity regularization term R drives most weights w to 0, retaining only key combinatorial structures (graph concepts).

Following the theory of perceptrons [31] and CBMs [24, 33], the weighted sum formulation realizes continuous relaxations of Boolean logic operations:

- **Soft AND**: $w_{c,i} > 0, w_{c,j} > 0, b_c < 0$. The class is activated only when both concept scores $\hat{E}(\mathbf{h}_G, \mathbf{h}_{T_i})$ and $\hat{E}(\mathbf{h}_G, \mathbf{h}_{T_j})$ are sufficiently large.
- **Soft OR**: $w_{c,i} > 0, w_{c,j} > 0, b_c \approx 0$. The class is activated when either concept score $\hat{E}(\mathbf{h}_G, \mathbf{h}_{T_i})$ or $\hat{E}(\mathbf{h}_G, \mathbf{h}_{T_j})$ is sufficiently large.
- **Soft NOT**: $w_{c,j} < 0$. A higher concept score $\hat{E}(\mathbf{h}_G, \mathbf{h}_{T_j})$ suppresses the logit of the class and reduces the prediction confidence.

The graph concept bottleneck layer is a learnable reasoning engine. It explicitly learns weighted logical combinations of concepts, which form the theoretical foundation of the soft combinatorial reasoning in our model.

D The Generation of Context-Aware Embedding-Based Concept Labels

To clarify the design of our context-aware graph concept embedding module, we present the full generation process of graph concept labels in Algorithm 1.

E Running Time and Complexity Analysis

Table 6 reports the preprocessing time of GCBM-E on different datasets, including three components: graph concept extraction, discriminative concept selection, and graph concept label generation. The result shows that the preprocessing time of GCBM-E is minimal on small datasets and remains acceptable on large datasets.

Table 7 reports the training time of GCBM, GCBM-E, and several representative baselines. As shown in Table 7, WWL (a WL-based graph kernel method) achieves the shortest training time, as graph kernel methods do not require constructing and training neural networks. Despite its fast speed, WWL provides relatively low classification accuracy. For fair comparison, all GNN-based and Graph Transformer-based methods use a unified setup of 300 training epochs and a batch size of 64. These results confirm that GCBMs yield superior training efficiency, achieving a good balance between computational cost and predictive performance. The last row shows the running time differences between GCBM-E and GCBM, i.e., the running time for the Transformer to learn the graph concept embeddings. We can observe that learning graph concept embeddings with the Transformer is scalable to large datasets.

GCBM-E has a longer training time than GCBM due to differences in the computational complexities of their graph and concept embedding models. GCBM derives graph and concept vectors directly from frequency statistics without additional training overhead, incurring only $O(n \cdot M \cdot |\mathcal{T}^{(k)}|)$ complexity for concept label calculation, where n is the number of graphs, M is the number of selected concepts, and $|\mathcal{T}^{(k)}|$ is the size of the graph concept vocabulary. In contrast, GCBM-E requires training both GIN as the backbone and a Transformer as the embedding model, leading to $O(n \cdot M \cdot d)$ complexity for concept label calculation plus extra overhead from Transformer training: $O(T \cdot n \cdot (L \cdot S^2 \cdot d + L \cdot S \cdot d^2))$ and $O(T \cdot M \cdot (L \cdot S^2 \cdot d + L \cdot S \cdot d^2))$. Here, T is the Transformer's training epochs, L is its number of network layers, S is the length of the "graph sentences", and d is

Algorithm 1: Generation of Context-Aware Embedding-Based Graph Concept Labels

Input: Graph dataset $\mathcal{D} = \{(G_i, \mathbf{y}_i)\}_{i=1}^{n}$, Max WL-subtree height K, Number of selected concepts M, Pre-trained language model LM

Output: Complemented dataset $\mathcal{D} = \{(G_i, \mathbf{c}_i, \mathbf{y}_i)\}_{i=1}^{n}$

`// Phase 1: Build graph sentence corpus`

Initialize the graph sentence corpus $\mathcal{C}_{gs} = \emptyset$;

for $k = 1$ **to** K **do**

  Initialize the k-level graph concept corpus $\mathcal{C}^{(k)} = \emptyset$;

  Extract the WL-subtree of height k for each node v in each graph G_i and add it to $\mathcal{C}^{(k)}$;

  Form the graph concept vocabulary $\mathcal{T}^{(k)} = \{T_1^{(k)}, \ldots, T_{|\mathcal{T}^{(k)}|}^{(k)}\}$ with the unique WL-subtrees in $\mathcal{C}^{(k)}$;

  Build the frequency matrix $\mathbf{X}^{(k)} \in \mathbb{R}^{n \times |\mathcal{T}^{(k)}|}$ of $\mathcal{D}$, where $\mathbf{X}_{i,j}^{(k)} = \text{Frequency}(G_i, T_j^{(k)})$;

  Calculate information gain $IG(\mathcal{D}, T_j^{(k)})$ for each concept $T_j^{(k)}$ via Eq. (1);

  Select top-M concepts from $\mathcal{T}^{(k)}$ to form the concept set $\mathcal{T}_{pruned}^{(k)} = \{T_1^{(k)}, \ldots, T_M^{(k)}\}$;

  Build "graph sentences" $S_i^{(k)}$ for each G_i and $S_{\mathcal{T}_{pruned}^{(k)}}$ for $\mathcal{T}_{pruned}^{(k)}$ with sorted WL-subtree encodings;

  Form the k-level graph sentence corpus $\mathcal{C}_{gs}^{(k)}$ with $S_i^{(k)}$ and $S_{\mathcal{T}_{pruned}^{(k)}}$;

  $\mathcal{C}_{gs} \leftarrow \mathcal{C}_{gs} \cup \mathcal{C}_{gs}^{(k)}$;

`// Phase 2: Generate graph concept label`

Train LM on $\mathcal{C}_{gs}$;

for $k = 1$ **to** K **do**

  Generate node embeddings $\mathbf{h}_{T_{v_{\pi(u)}}^{(k)}}$ and concept embeddings $\mathbf{h}_{T_j^{(k)}}$ via Eq. (5);

  Aggregate the node embeddings of G_i and generate graph embeddings $\mathbf{h}_{G_i}^{(k)}$ via Eq. (6);

  Calculate concept scores via Eq. (3) and generate k-level concept labels $\mathbf{c}_i^{(k)}$ via Eq. (2);

Generate global graph labels $\mathbf{c}_i = \text{Concat}(\mathbf{c}_i^{(1)}, \ldots, \mathbf{c}_i^{(K)})$;

Complement the dataset $\mathcal{D} = \{(G_i, \mathbf{c}_i, \mathbf{y}_i)\}_{i=1}^{n}$ with $\mathbf{c}_i$;

return $\mathcal{D}$

Table 6: Preprocessing time (in seconds) of each component in GCBM-E .

Dataset	MUTAG	COX2	DHFR	IMDB-M	OGBG-MOLHIV
Graph Concept Extraction	0.28	2.72	6.49	7.21	2810.93
Discriminative Concept Selection	0.21	0.71	1.24	1.19	586.52
Graph Concept Label Generation	7.27	10.53	15.59	22.11	10226.89

Table 7: Training time (in seconds) of GCBMs and representative baselines. The reported training time of all GNNs is the total training time for 300 epochs.

Dataset	MUTAG	COX2	DHFR	IMDB-M	OGBG-MOLHIV
GNNExplainer	304.75	596.58	667.59	1470.42	90813.60
WWL	4.75	44.52	117.29	263.62	10172.49
GRDL	173.44	354.27	524.56	1396.85	73856.83
GRAPHORMER	298.80	559.95	595.82	1447.38	85964.34
GCBM	158.06	332.09	424.62	1152.57	60557.46
GCBM-E	169.05	352.12	513.79	1308.29	72625.71
Transformer embedding	10.99	20.03	89.17	155.72	12068.25

the hidden layer dimension. S^2 and d^2 terms are the key drivers of GCBM-E 's higher complexity. Other components share roughly the same complexity for both models: graph concept extraction is $O(n \cdot K \cdot |\mathcal{V}|)$, where K is WL-subtrees height and $|\mathcal{V}|$ is the number of nodes, discriminative concept selection is approximately $O(n \cdot |\mathcal{T}^{(k)}|)$, and the forward computation complexity of the GIN backbone is $O(n \cdot K \cdot |\mathcal{V}| \cdot d)$.

In general, GCBM achieves higher training efficiency, making it a more practical solution for fast graph classification tasks. In contrast, GCBM-E obtains higher accuracy through a higher-quality concept bottleneck and can be applied to scenarios with stricter requirements for reliability. In addition, we will adopt subgraph sampling in future work to reduce the complexity of Transformer training. Since S^2 and d^2 are the key drivers of complexity, we split large graphs into multiple subgraphs and apply GCBM-E to each subgraph. For the final node embeddings, we aggregate the embeddings of each node across all subgraphs containing it by averaging, which preserves the global information of the original graph while maintaining low complexity.

F Experimental Setup

Datasets We conducted experiments on 9 public graph datasets (covering bioinformatics: MUTAG, PTC-MR, BZR, COX2, DHFR; social networks: IMDB-B, IMDB-M, COLLAB, DBLP_v1) for graph classification tasks across small, medium, and large scales, along with 3 large-scale class-imbalanced datasets PC-3, MCF-7 and OGBG-MOLHIV to test performance in imbalanced scenarios, and 2 specialized interpretation datasets Solubility [15], Benzene [42] for interpretability evaluation. Dataset statistics are available in [32].

Baselines We select 5 groups of representative advanced methods as baselines for a comprehensive evaluation, categorized as follows.

Interpretable GNN models (Group 1): These methods focus on identifying graph substructures that support the model's correct predictions, including GNNExplainer [63], PGExplainer [28], ProtGNN [70], VGIB [64], PGIB [44], ConfExplainer [68], GraphTrail [3], LogiX [38], LogicXGNN [16], LogiX-GIN [39], DC-GNN [5].

WL-based graph kernels (Group 2): These methods measure graph similarity via hand-crafted kernel functions, serving as classical baselines in graph classification, including WL-subtree kernel [45], WWL [51], FWL [43], and GAWL [18].

Optimal transport-based models (Group 3): These methods align graph distribution features using optimal transport theory and perform prominently in similarity measurement tasks, including OT-GNN [10] and WEGL [25].

Mainstream GNN models (Group 4): These methods capture graph structural features through hierarchical aggregation, representing the mainstream of graph representation learning. These models include GIN [59], DropGIN [34], SEP [56], GMT [6], MinCutPool [8], ASAP [40], WIT-TOPOPOOL [12], and GRDL [55].

Graph Transformer models (Group 5): These methods introduce self-attention mechanisms to model long-range dependencies in graphs, demonstrating potential in handling complex graph structures. These models include GRAPHORMER [62] and SAT [11].

Parameter Settings and Hardware Configuration The default parameters for each comparison method are used. For GCBMs:

We adopt the Transformer variant BERT [14] as the LM. Both the number of hidden layers and that of attention heads are set to 4, and the hidden dimension is set to 64. The maximum length of text sequences is set to the length of the longest sequence among the "graph sentence" in the dataset. Input tokens are randomly masked at a 15% rate for self-supervised training. The BERT model is trained for 3 epochs with cross-entropy as the loss function, and the batch size is adjusted according to the dataset size. A detailed discussion of the sequential positional encodings in BERT is provided in Appendix M.

GCBM's backbone GIN consists of 4 layers with a fixed hidden dimension of 64. The readout operation follows the default setting of GIN, using global sum pooling. To enhance the propagation of deep representations, we introduce residual connections for each layer with learnable residual weights.

In addition, we select $M = 64$ highly discriminative concepts for each layer and dynamically adjust the importance of concepts at different levels via learnable hierarchical concept weights. Furthermore, a dropout mechanism is introduced in each layer to prevent overfitting.

During training, concept labels are incorporated to compute concept loss; during testing, no concept labels are required, and the model automatically predicts concept labels and outputs classification results. The total number of training epochs is set to 300, with the Adam [22] optimizer employed. The learning rate scheduling strategy is ReduceLROnPlateau, with a minimum improvement threshold of 0.1 and a minimum learning rate of 10^{-6}. We use the Optuna [1] optimization framework to search for hyperparameters, including dropout rate, initial learning rate, weight decay, learning rate decay factor, patience, and cooldown period. Additionally, a gradient clipping strategy is applied to stabilize training and prevent gradient explosion. Experiments perform 10-fold cross-validation.

Experiments are conducted on a server equipped with an Intel (R) Xeon (R) Gold 6226R processor, 256GB memory, an NVIDIA RTX 3090 GPU, and Ubuntu 18.04.6 LTS as the operating system. The deep learning framework used is PyTorch 1.13.0.

G Quantitative Faithfulness Evaluation of Explanations

Following the ERASER benchmark suite, we quantitatively evaluate the explanation faithfulness of GCBMs using Sufficiency and Comprehensiveness.

G.1 Definition and Extraction of "Top-m Concepts"

To ensure fair comparison, we adapt the concept definitions and extraction protocols according to the representation mechanisms of each explainer family.

Post-hoc Explainers (e.g., GNNExplainer) A concept is defined as a connected subgraph motif with high attribution weight. We rank the learned continuous edge mask weights $\sigma(M_e)$ in descending order, retain the top 50% highest-weighted edges, and zero out the remaining edges. Following standard benchmarking practices to convert continuous edge attributions into discrete motif concepts, we apply connected component decomposition to the retained edge set to extract m disjoint connected subgraphs $\{G_1, G_2, \ldots, G_n\}$. For each component G_i, its importance score is computed as the sum of its internal edge mask weights: $\text{Score}(G_i) = \sum_{e \in G_i} \sigma(M_e)$. We then sort the subgraphs by $\text{Score}(G_i)$ in descending order and select the top-m subgraphs as the Top-m concepts.

Logic-based Self-Explainable GNNs (e.g., LogiX-GIN) LogiX-GIN expresses graph classification decisions as disjunctive normal form logical rules over graded modal logic (GML) literals. A concept is defined as a binary rule literal/motif ($\tilde{a}[i] \in \{0, 1\}$) in the final TELL logic layer, where each activated literal $\tilde{a}[i] = 1$ corresponds to a set of induced neighborhood subgraphs satisfying a specific GML counting and topological condition. Leveraging the non-negative monotonicity constraint ($\mathbf{W}^+ \geq 0$) of the TELL classifier, we rank all rule literals according to the target class's non-negative weight vector $\mathbf{W}^+[i]$ in descending order. The m highest-weighted literals are selected as the Top-m concepts, whose corresponding non-isomorphic induced subgraph patterns are mapped via GML.

GCBMs A concept is an entry T_i from the graph concept set $\mathcal{T}_{\text{pruned}}$, mined via information gain in the preprocessing stage and represented as a WL-subtree topological motif. Top-m concepts are directly taken as the concepts corresponding to the m neurons with the largest absolute weight magnitudes $|\mathbf{W}[i]|$ in the concept bottleneck layer.

G.2 Protocols for Concept Removal and Retention

Post-hoc Explainers For removal, we locate the nodes and edges belonging to the Top-m subgraph motifs in the original graph G_i and mask them out by deleting corresponding edges and zeroing node feature vectors, yielding the masked graph $G_{i \setminus \text{top-}m}$. For retention, we keep only the nodes and edges belonging to the Top-m subgraph motifs and mask all other graph components of G_i, obtaining the subgraph $G_{i,\text{top-}m}$.

Logic-based GNNs For removal, we directly zero out the binary activations of the Top-m rule literals ($\tilde{a}[\text{top-}m] = 0$) in the TELL logic layer of LogiX-GIN. This is equivalent to deactivating these m GML subgraph conditions, and the model outputs the predicted probability $p(y_i \mid \tilde{a}_{\setminus \text{top-}m})$.

Table 8: Quantitative faithfulness evaluation on Solubility and Benzene ($m = 5$).

Dataset	Solubility		Benzene	
	Suff	Comp	Suff	Comp
GNNExplainer	0.38	0.26	0.31	0.15
PGExplainer	0.29	0.24	0.22	0.20
GraphTrail	0.25	0.38	0.17	0.36
LogiX	0.20	0.52	0.15	0.44
LogiX-GIN	0.17	0.59	0.08	0.54
GCBM-E	0.13	0.61	0.08	0.58

For retention, we keep the activations of Top-m literals ($\tilde{a}[\text{top-}m] = 1$) and set all non-top-$m$ literal activations to zero ($\tilde{a}[\text{non-top-}m] = 0$), then compute $p(y_i \mid \tilde{a}_{\text{top-}m})$.

GCBMs In GCBMs, the forward prediction follows the linear decision process $\hat{\mathbf{y}} = \sigma\left(\sum_{i=1}^{M} \hat{c}_i \cdot \mathbf{W}[i] + b\right)$. For removal, we directly set the weights corresponding to the Top-m concepts in the linear classifier to zero ($\mathbf{W}[\text{top-}m] = 0$). This operation is equivalent to zeroing the activation scores of the corresponding concepts and completely cutting off the information flow of these concepts to the decision function. For retention, we retain the linear weights of the Top-m concepts and set all non-top-m weights to zero ($\mathbf{W}[\text{non-top-}m] = 0$). Note that GCBM adopts L_1 sparsity regularization during training, so the weights of most non-top-m concepts naturally converge close to zero after training.

G.3 Formal Metrics Computation

Let $\mathcal{D}$ denote the dataset and $\mathbf{y}_i$ denote the ground-truth label of graph G_i. The Sufficiency and Comprehensiveness metrics are computed as follows:

- Sufficiency (Suff $\downarrow$): Measures the deviation of predicted probability when only the Top-m concepts are retained:

$$\text{Suff} = \frac{1}{|\mathcal{D}|} \sum_{G_i \in \mathcal{D}} |p(\mathbf{y}_i \mid G_i) - p(\mathbf{y}_i \mid \text{top-}m)| \tag{8}$$

- Comprehensiveness (Comp $\uparrow$): Measures the drop in predicted probability after removing the Top-m concepts:

$$\text{Comp} = \frac{1}{|\mathcal{D}|} \sum_{G_i \in \mathcal{D}} \left(p(\mathbf{y}_i \mid G_i) - p(\mathbf{y}_i \mid \setminus\text{top-}m)\right) \tag{9}$$

Table 8 reports the quantitative faithfulness evaluation results including Sufficiency and Comprehensiveness metrics on the Solubility and Benzene datasets with $m = 5$. The result shows that compared with post-hoc explainers and logic-based self-explainable baselines, GCBM-E achieves the lowest Sufficiency and the highest Comprehensiveness on both datasets. This observation verifies that the concepts extracted by GCBM-E carry decisive predictive information for graph classification and yield superior explanation faithfulness.

H Quantitative Assessment of Interpretability

GCBMs extract key subgraphs in an efficient manner without additional training, which is simply achieved by adjusting the number of selected concepts M. We further conduct parameter sensitivity analysis to validate the rationality of this design. Table 9 reports the AUC scores of GCBMs under different M values.

I Ablation Studies for Generality of GCBMs

To validate the generality of GCBMs across different backbones, graph substructures, and LMs, we perform ablation experiments on 4 datasets. We replace each core component with representative

Table 9: AUC scores (mean%) of GCBM/GCBM-E with different M values.

Dataset	Solubility	Benzene
M=2	91.2	83.6
M=5	81.2	85.9
M=10	72.8	86.1
M=15	68.9	84.0

alternatives while keeping other settings consistent. Specifically, GCBM-SG replaces WL-subtrees with general subgraphs [2]. GCBM-GAT replaces the GIN backbone with the graph attention network GAT [53]. GCBM-W2V and GCBM-G replace the graph concept embedding model Transformer with Word2Vec [30] and GloVe [35], respectively. Table 10 reports the classification accuracy of GCBM-E and these variants.

Table 10: Classification accuracy (mean%±standard deviation%) of GCBM-E and its variants on benchmark datasets.

Dataset	MUTAG	COX2	DHFR	IMDB-M
GCBM-SG	92.8±3.7	85.3±3.0	85.4±3.5	52.9±2.0
GCBM-GAT	92.7±3.0	85.7±3.4	85.9±2.9	52.6±2.1
GCBM-W2V	93.2±3.9	85.6±3.3	86.0±2.8	52.8±2.4
GCBM-G	93.0±4.2	85.7±2.9	86.0±3.7	52.4±2.1
GCBM-E	93.6±4.0	85.9±3.2	86.1±3.2	53.1±2.4

These results demonstrate that GCBMs serve as a generic and flexible framework that can be easily extended to various backbones, graph substructures, and LMs, due to their modular design. First, the backbone acts as a general graph encoder that maps input graphs to concept label vectors, where the concept loss aligns these vectors with ground-truth concept labels. This alignment is independent of the GNN's message-passing mechanism, enabling compatibility with various backbones. Second, graph concepts are uniformly defined as discrete topological substructures. We construct a concept vocabulary from non-isomorphic substructures and represent each graph as a concept token sequence ("graph sentence"). This pipeline is substructure-agnostic and naturally supports various substructure types. Third, different LMs can effectively learn concept embeddings, as all LMs support token sequence inputs ("graph sentence") without task-specific adaptation.

J Extended Sensitivity Analysis on Hyperparameter K

To evaluate the impact of the maximum WL-subtree height K, we conducted extended experiments up to $K = 5$ and $K = 6$. Table 11 reports the classification accuracy of GCBM-E across different K values on representative datasets.

Table 11: Classification accuracy (mean%±standard deviation%) of GCBM-E for different K.

Dataset	MUTAG	DHFR	IMDB-M
$K = 1$	87.8±4.5	80.8±3.2	51.6±2.8
$K = 2$	91.3±4.1	84.8±3.1	52.5±2.6
$K = 3$	93.1±3.8	86.1±3.0	52.8±2.5
$K = 4$ (Default)	93.6±4.0	86.1±3.2	53.1±2.4
$K = 5$	93.6±4.2	86.2±3.0	53.0±2.5
$K = 6$	93.3±4.0	85.8±3.5	52.7±2.2

As shown in Table 11, the performance drops sharply at $K = 1$. This is because $K = 1$ strictly restricts graph concepts to WL-subtrees of height 1, discarding critical multi-hop topological information and severely impairing the model's expressiveness. When $K \geq 3$, the classification accuracy

on all datasets enters a high and stable performance plateau. Increasing K to 5 or 6 brings no further accuracy improvement , and occasionally incurs a slight degradation due to subtree redundancy and mild overfitting.

Setting the default to $K = 4$ consistently achieves optimal or near-optimal predictive accuracy across various datasets (see Table 2 and Table 3), while preserving the interpretability of concepts. Therefore, the hyperparameter K introduces almost no tuning burden in practical applications.

K Detailed Procedures for Intervention Experiments

For graph G_i, $\hat{\mathbf{y}}_i$ and $\mathbf{y}_i$ denote the predicted label and the ground-truth label, respectively. $\hat{\mathbf{c}}_i$ and $\mathbf{c}_i$ represent the predicted concept label and the ground-truth concept label, respectively. W_F is the weight matrix of the concept bottleneck layer, where $W_{F[i,j]}$ denotes the weight of the j-th concept corresponding to the i-th class in W_F, and $\text{sim}(\cdot, \cdot)$ denotes the cosine similarity between two vectors.

Sample Selection Criteria Intervention experiments are conducted exclusively on misclassified graphs satisfying two criteria:

- High consistency between the predicted concept label and the ground-truth concept label: $\text{sim}(\hat{\mathbf{c}}_i, \mathbf{c}_i) \geq \tau_c$, where τ_c is a similarity threshold ensuring valid concept activation;
- The graph belongs to the target error type: $\mathbf{y}_i = \text{cls 1}$ but $\hat{\mathbf{y}}_i = \text{cls 0}$.

The intervention graph set is defined as $S = \{G_i \mid \hat{\mathbf{y}}_i = \text{cls 0}, \mathbf{y}_i = \text{cls 1}, \text{sim}(\hat{\mathbf{c}}_i, \mathbf{c}_i) \geq \tau_c\}$.

Core Parameter Calculation for Intervention For the intervention graph set S, we first compute two key statistics reflecting the overall error properties: the average logit difference $\overline{\Delta a}$ and the average activation value $\overline{f^{(\text{cpt})}}$ of the target concept.

For graph $G_i \in S$, the pre-intervention logit difference is defined as:

$$\Delta a_{G_i} = \text{logit}_{G_i}(\text{cls 0}) - \text{logit}_{G_i}(\text{cls 1}), \tag{10}$$

where $\text{logit}_{G_i}(\text{cls}) = W_{F[\text{cls},:]} \cdot \hat{\mathbf{c}}_i^T + b_F$ and b_F is the bias term of the concept bottleneck layer. The average logit difference of S is:

$$\overline{\Delta a} = \frac{1}{|S|} \sum_{G_i \in S} \Delta a_{G_i}. \tag{11}$$

$\overline{\Delta a} > 0$ indicates that the model's logit value for the incorrect class (cls 0) is higher than that for the correct class (cls 1) on S, and the intervention aims to reverse this gap.

For the target concept (cpt j), the activation value of G_i is $\hat{\mathbf{c}}_i[j]$, corresponding to the j-th entry of $\hat{\mathbf{c}}_i$. The average activation value of the target concept on S is defined as:

$$\overline{f^{(\text{cpt j})}} = \frac{1}{|S|} \sum_{G_i \in S} \hat{\mathbf{c}}_i[j]. \tag{12}$$

This parameter quantifies the average relevance of the target concept to S.

Weight Adjustment Rules The objective of the intervention is to adjust the target concept's weights for the two classes, enhancing its contribution to the correct class while suppressing that to the incorrect class.

The weight adjustment value is calculated as:

$$\Delta w = \frac{\overline{\Delta a} + b}{2 \cdot \overline{f^{(\text{cpt})}}}, \tag{13}$$

where b is the margin aiming to ensure the correct class's average logit stably exceeds the incorrect class's.

Weight Modification Operations For each target concept (cpt j), weight adjustments follow:

- Increase the weight for the correct class (cls 1): $W_{F[\text{cls 1,cpt j}]} \leftarrow W_{F[\text{cls 1,cpt j}]} + \Delta w$;

- Decrease the weight for the incorrect class (cls 0): $W_{F[\text{cls 1,cpt j}]} \leftarrow W_{F[\text{cls 1,cpt j}]} - \Delta w$;
- Retain all other weights unchanged to avoid interfering with the classification logic of unrelated non-target concepts.

Intervention Implementation Details We perform two rounds of interventions on GCBM-E, focusing on key concepts 20 and 42 (verified as key substructures via human evaluation in interpretability experiments). Detailed parameters are summarized in Table 12.

Table 12: Detailed parameters of the intervention experiments.

Concept	$\overline{\Delta a}$	$\overline{f^{(j)}}$	Δw	$W_{F[\text{cls 0,cpt j}]}$	$W_{F[\text{cls 1,cpt j}]}$
cpt 20	0.51	0.8426	0.4213	0.3842 $\rightarrow$ -0.0371	-0.0112 $\rightarrow$ 0.4101
cpt 42	0.32	0.7635	0.3405	0.1876 $\rightarrow$ -0.2466	0.0939 $\rightarrow$ 0.5281

Note that the concept similarity threshold τ_c and the intervention margin b are tunable hyperparameters. Adjusting them can further optimize the intervention effects. The default settings ($\tau_c = 0.6$, $b = 0.2$) were determined via 5-fold cross-validation to balance concept activation validity and intervention stability. After each round of weight modification, the model is re-evaluated on the entire test set to ensure the number of new misclassifications is fewer than that of corrected samples.

L Mitigating Concept Leakage and Shortcut Learning in GCBMs

L.1 Mitigating Concept Leakage via Constraints

Unlike unconstrained latent representations, the concept vector $\hat{\mathbf{c}}_i$ in GCBMs is constrained by concept labels $\mathbf{c}_i$ and directly supervised by the concept loss $\mathcal{L}_c$. Furthermore, with L_1 sparsity regularization, $\hat{\mathbf{c}}_i$ is compressed into a highly sparse bottleneck that retains only a small set of selected discriminative concepts. This constrained bottleneck forces the GNN backbone to route nearly all prediction-relevant information through human-interpretable topological concepts, substantially limiting the channel capacity available for extraneous or leaked signals.

L.2 Verifying Shortcut Reliance via Concept Intervention Experiments

To verify whether the model truly identifies and relies on target concepts instead of exploiting spurious shortcut correlations, we perform concept intervention experiments. As shown in Sec. 5.2.5, manually modifying the activation score or classifier weight of a specific concept shifts the output class logit in a predictable manner and successfully rectifies misclassified samples. If the model only relied on spurious shortcuts, direct interventions on the concept activation score or weights would not produce such consistent corrections across misclassified instances.

M Discussion on the standard sequential positional encodings in BERT

M.1 Target Concept Embeddings are Position-Invariant

The embedding of a concept does not change with its position across different "graph sentences". For each concept $T_j \in \mathcal{T}_{pruned}$, its embedding $\mathbf{h}_{T_j}$ is extracted from the fixed concept sequence $S_{\mathcal{T}_{pruned}}$. Accordingly, $\mathbf{h}_{T_j}$ is a static vector that remains unchanged over the entire dataset. When computing the score between an arbitrary graph G_i and this concept, the graph embedding $\mathbf{h}_{G_i}$ is always matched against this fixed global vector $\mathbf{h}_{T_j}$ via cosine similarity.

M.2 Lexicographic Sorting and Positional Encoding: Capturing Topological Proximity Rather Than Natural Language Word Order

Positional encodings in natural language modeling encode temporal order and grammatical structure, yet they serve a fundamentally different role in our framework. Specifically, we sort concepts deterministically by their WL-subtree encodings to construct the "graph sentence" S_i. This sorting

strategy naturally groups concepts with similar topology into adjacent positions within the sequence. Positional encodings do not represent syntactic hierarchies of natural language. Instead, they allow the self-attention mechanism to measure relative topological distances and co-occurrence patterns among adjacent subgraphs. LMs do not learn the absolute spatial coordinates of concepts, but contextual co-occurrence patterns induced by the sequence ordering.

M.3 Domain Rationality: Context-Awareness and Continuous Soft Supervision

A straightforward example comes from molecular property prediction: the biochemical activity of a functional group (concept) naturally depends on its adjacent groups (context) within a molecule, analogous to how word semantics rely on surrounding context. Constructing graph sentences from sorted concepts enables the graph representation $\mathbf{h}_{G_i}$ to capture global contextual interactions and distinguish which concepts (substructures) share similar chemical environments. Instead of hard binary labels, we compute the continuous cosine similarity between the context-aware graph embedding $\mathbf{h}_{G_i}$ and the invariant concept embedding $\mathbf{h}_{T_j}$ to generate continuous pseudo-labels for quantifying concept alignment. This strictly follows standard established practices in vision-language CBMs [33].

M.4 Role of Transformer: Concept Label Refiner, Not Decision Engine

Crucially, the LM is not involved in inference and prediction. It is used only during offline data preprocessing to produce high-quality continuous concept supervision signals $\mathbf{c}_i$ and is fully detached at inference time. The GNN backbone performs a forward pass to predict concept scores $\hat{\mathbf{c}}_i$, and the concept bottleneck layer yields the final prediction $\hat{\mathbf{y}}_i = f(\hat{\mathbf{c}}_i)$. The entire online prediction pipeline remains intrinsically interpretable and manually intervenable, removing concerns that positional encodings may compromise the interpretability of online decision-making.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have made clear claims about contributions and scopes in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have the seperated limitation section in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the details of our model and training strategy in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide implementation code for reproducing our work in the supplementary materials.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We have introduced experiment settings and details in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the mean and standard deviation in our performance evaluation experiments, and clarify that the error sources are 10-fold cross-validation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have discussed the information about computing resources in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We follow the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: GCBMs not only contribute to academic research but also show broad potential in real-world applications. They help promote the reliable deployment of models in high-stakes fields such as drug discovery.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We do not release any data or model that has high risks for misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cite the original papers for all open-source datasets, models, and other resources.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide documentation with our code.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing or human subjects research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects are involved in this research, so IRB approval or equivalent ethical review is not required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: We only use Transformer/BERT as a standard module for graph concept embedding; LLMs are not part of core innovations and are only used for writing assistance, so no declaration is needed.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.