Graph Contrastive Learning via Weisfeiler-Leman Dual-View Sampling

Anonymous authors
Paper under double-blind review

Abstract

Graph contrastive learning (GCL) approaches have gained momentum over the past few years. By augmenting the original graph data, the common GCL pipeline learns from such multiple contrastive graph views in a self-supervised manner, tackling critical issues in the literature, such as node label scarcity. To obtain contrastive views, most GCL techniques heavily rely on feature-space similarity measures. We consider this as a limiting factor in GCL, since it implies that node features are (in general) informative and closely aligned with the graph topology, an assumption that does not hold, for instance, in the case of heterophilic graphs. In this work, we propose to address the problem by coupling the usual feature-space similarities with structure-based measures, which we propose to implement through the Weisfeiler-Leman (WL) family of algorithms. Our framework, dubbed WLGCL, introduces a dual-view sampling strategy that works on features- and WL-level to construct more reliable contrastive pairs. WLGCL integrates a multi-positive and hard-negative contrastive loss to ensure the alignment-uniformity trade-off without modifying the design of the graph encoder. Extensive experiments on six benchmark datasets and against seven state-of-the-art baselines demonstrate the efficacy of WLGCL, where additional empirical evaluations justify the adoption of our architectural choices for the model.

1 Introduction

Self-supervised learning (SSL) (Gui et al., 2024) has emerged as a powerful paradigm for learning generalizable representations without human-annotated labels. By designing pretext tasks that exploit the inherent structure of data, SSL enables models to extract semantic patterns that transfer effectively to downstream tasks (Tomasev et al., 2022). Among various SSL approaches, contrastive learning (CL) has become one of the most influential frameworks. CL learns discriminative embeddings by encouraging representations of semantically related samples to be similar while enforcing separation between unrelated ones. Originally developed in computer vision with methods such as SimCLR (Chen et al., 2020) and MoCo (He et al., 2020), CL has since been extended to images (Radford et al., 2021), text (Gao et al., 2021), and audio (Baevski et al., 2020), demonstrating that well-designed positive-negative relational constraints can drive robust feature learning across diverse modalities.

Motivated by this success, recent years have witnessed rapid progress in graph contrastive learning (GCL) (Velickovic et al., 2019; You et al., 2020; Zhu et al., 2020), which adapts contrastive principles to relational and non-Euclidean data. GCL methods typically use paired graph augmentations or cross-view transformations to enforce invariance, encouraging the model to capture both topological structure and attribute information. A wide spectrum of design choices (*e.g.*, view generation, contrastive objectives, sampling strategies, and structural granularity) has led to a variety of methods, including GRACE (Zhu et al., 2020) and BGRL (Thakoor et al., 2022), as well as spectral and subgraph-based approaches such as PolyGCL (Chen et al., 2024) and FOSSIL (Sangare et al., 2025). These techniques have achieved strong performance in node classification, clustering, and link prediction, underscoring the versatility of contrastive objectives in graph representation learning.

Despite these advancements, most GCL frameworks still rely heavily on feature-space similarity to construct positive and negative samples. Whether the sampling is explicit or implicit, similarity is determined almost exclusively by distances in the learned embedding space. This assumption works well when node features are informative and strongly aligned with graph topology, as in homophilic graphs. However, it often breaks down in heterophilic or structurally complex settings, where nodes sharing similar structural roles may look entirely different in feature space (Liu et al., 2023). Consequently, feature-based sampling may generate (i) false positives, *i.e.,* nodes that appear close in embeddings but are structurally unrelated; and (ii) weak negatives, *i.e.*, structurally similar nodes that are mistakenly pushed apart. These issues undermine contrastive alignment, distort the embedding geometry, and reduce robustness under distribution shifts. Meanwhile, the Weisfeiler-Leman (WL) family of algorithms offers a principled and expressive way to capture multi-scale structural patterns (Weisfeiler & Leman, 1968). WL refinements, widely used in graph kernels and GNN expressivity analysis, provide structure-aware node similarity that is invariant to permutations and independent of feature noise (Morris et al., 2019). However, despite their strong structural discrimination capabilities, WL-based similarities remain largely underutilized in contrastive learning, which continues to rely predominantly on feature-driven heuristics.

To bridge this gap, we propose WLGCL, a dual-view sampling framework that integrates feature-space similarity with WL-based structural similarity to construct more reliable contrastive pairs. Rather than assuming that embeddings implicitly encode structure, we decouple the two notions of similarity and treat their intersection as trustworthy positives, while using the mismatched cases as naturally informative hard negatives. More fundamentally, our approach reframes contrastive sampling as a consistency problem across views, where agreement between feature-based and structural similarities serves as a signal of semantic reliability. This perspective provides a principled alternative to conventional feature-driven sampling and clarifies the role of structure in contrastive learning. Building upon this sampling strategy, we further introduce a multi-positive contrastive objective that enhances the alignment-uniformity trade-off without altering the encoder design. Across six benchmark datasets spanning homophilic and heterophilic settings, WLGCL consistently outperforms state-of-the-art baselines while maintaining competitive training efficiency.

2 Related Work

This work is related to two lines of research: graph contrastive learning (GCL) and the Weisfeiler-Leman (WL) algorithm. GCL provides a self-supervised framework for learning graph representations by contrasting multiple views, while the WL algorithm offers a principled way to capture graph structural equivalence. Our method connects these two directions by incorporating WL-based structural similarity into contrastive pair construction. We review GCL first, followed by the WL algorithm.

2.1 Self-Supervised and Contrastive Learning on Graphs

Contrastive learning (CL) became popular for the first time in computer vision (Radford et al., 2021), where methods such as SimCLR (Chen et al., 2020) and MoCo (He et al., 2020) demonstrated that meaningful representations can be learned without manual labels by contrasting positive and negative pairs. This paradigm has since been successfully extended to other modalities, such as text (Gao et al., 2021), and audio (Baevski et al., 2020), and more recently has been applied to graph-structured data, giving rise to graph contrastive learning (GCL). The common self-supervised GCL framework learns node- or graph-level representations by contrasting multiple correlated views of a graph (Velickovic et al., 2019; You et al., 2020; Zhu et al., 2020). By enforcing invariance across views while maintaining embedding uniformity, GCL has shown strong performance on downstream tasks, including node classification, clustering, and link prediction (Liu et al., 2023). The contrastive objectives have further been refined through improved sampling and optimization strategies, such as hard-negative mining in MoCHi (Kalantidis et al., 2020). As we will show, we extend these works to a structure-aware, multi-positive setting on graphs.

A typical GCL pipeline consists of three components: a graph encoder, a view generation module, and a contrastive objective. The encoder is usually implemented using message-passing GNNs such as GCN (Kipf & Welling, 2017), GAT (Velickovic et al., 2018), or GIN (Xu et al., 2019). The view generator constructs multiple correlated graph representations via stochastic perturbations, including feature masking, edge or

node dropping, subgraph sampling, diffusion-based propagation, or spectral filtering. The contrastive objective then encourages representations of the same instance across views to be similar, while separating representations of different instances, commonly through InfoNCE-style losses (van den Oord et al., 2018) or mutual information maximization.

Early GCL methods primarily focused on node-to-node or graph-to-graph contrast based on augmented views. GRACE (Zhu et al., 2020), for instance, relies on stochastic feature and edge perturbations to generate paired views, while DGI (Velickovic et al., 2019) maximizes mutual information between local node embeddings and a global summary representation. However, such approaches are often sensitive to augmentation design and negative-sample bias. To improve training stability, augmentation-free or negative-free methods have been proposed, moving beyond purely contrastive objectives. MUSE (Yuan et al., 2023) combines contrastive alignment with semantic masking and reconstruction to better handle heterophilic graphs. Beyond contrastive techniques, BGRL (Thakoor et al., 2022) adopts a bootstrap self-distillation strategy with a momentum-updated target encoder.

More recent approaches explore richer structural and spectral perspectives in GCL. PolyGCL (Chen et al., 2024) introduces learnable polynomial filters to generate frequency-aware views, while EPAGCL (Xu et al., 2025) provides theoretical analysis linking contrastive objectives with spectral smoothing effects. SpeGCL (Shou et al., 2025) further departs from explicit positive-pair construction by enforcing spectral consistency without predefined positives. In parallel, AutoGCL (Yin et al., 2022) automates view generation through learnable augmentation policies, while subgraph-level methods such as FOSSIL (Sangare et al., 2025) leverage optimal transport to align higher-order structural patterns across sampled subgraphs.

Despite these advances, most existing GCL methods still define contrastive pairs predominantly based on feature-space similarity, implicitly assuming that learned embeddings faithfully encode structural roles. This assumption often fails on graphs where feature similarity and structural similarity diverge, motivating the need for more reliable, structure-aware sampling strategies, which we address in this work.

2.2 Weisfeiler-Leman Algorithm

The Weisfeiler-Leman (WL) algorithm is a classical heuristic for graph isomorphism testing that iteratively refines node representations based on local neighborhood structure (Weisfeiler & Leman, 1968). Starting from an initial node labeling, WL repeatedly aggregates the labels of neighboring nodes to produce increasingly fine-grained partitions of the vertex set. Through this refinement process, nodes that share similar structural roles tend to receive identical colors, while structurally distinct nodes are progressively separated. Although the WL algorithm cannot distinguish all non-isomorphic graphs (for instance, certain regular graphs remain indistinguishable under standard WL refinement (Kriege et al., 2018)), it is highly effective for a broad class of graphs and has been widely adopted as a principled notion of structural similarity (Babai & Kucera, 1979). Its ability to capture multi-scale topological patterns makes it particularly useful for comparing nodes and graphs in a permutation-invariant manner.

WL-based refinements have played a central role in graph learning (Morris et al., 2023). Early work, such as the WL-subtree kernel, demonstrated strong performance in graph classification by explicitly exploiting the refinement hierarchy (Shervashidze et al., 2011). More recently, it has been shown that message passing graph neural networks (GNNs) follow a computation pattern analogous to the WL algorithm, iteratively updating node embeddings through neighborhood aggregation (Xu et al., 2019; Morris et al., 2019). As a result, the expressive power of standard GNNs is upper-bounded by the discriminative ability of the WL test.

Extensions of the WL algorithm to higher dimensions further enhance its expressive capacity by refining tuples of nodes rather than individual vertices (Cai et al., 1992). These higher-order WL variants have inspired the design of more powerful graph neural architectures and are commonly used as theoretical tools for analyzing GNN expressiveness (Morris et al., 2020). Overall, the WL family of algorithms provides a principled and structure-aware foundation for graph representation learning, motivating its still-unexplored integration into modern self-supervised and contrastive learning frameworks.

3 Proposed Methodology

In graph contrastive learning (GCL), positive and negative samples are typically constructed based on feature similarity in the embedding space. This implicitly assumes that learned representations faithfully capture the underlying graph structure, an assumption that may not hold in many real-world graphs where feature similarity and structural roles are misaligned. As a result, feature-based sampling often introduces false positives, nodes that appear close in the embedding space yet are structurally unrelated, as well as weak negatives, since randomly or queue-based sampled negatives may include nodes that are structurally or functionally similar to the anchor. This motivates the need for a more robust sampling strategy that explicitly incorporates structural information alongside feature-based similarity.

To this end, we propose WLGCL, a dual-view sampling framework that integrates feature similarity and WL-based structural similarity for contrastive pair construction. Our approach is grounded in the expressivity of the WL algorithm. It generates a hierarchical WL tree capturing multi-hop structural patterns for each node and serves as the structural backbone of our framework. Based on this, WLGCL proceeds in two stages: (i) dual-view candidate retrieval, and (ii) intersection-based positive extension with hard negative mining. In stage (i), for each anchor node, we independently retrieve candidate neighbors from the feature embedding space and the WL structural space, forming two complementary views of similarity. In stage (ii), we take the intersection of these two sets as reliable positives, while nodes appearing in only one view are treated as informative hard negatives. This design enables the model to retain semantically consistent positives while explicitly identifying samples that are misleading from either perspective, overcoming limitations of purely feature-based or random sampling strategies. Finally, the learned representations are optimized using a contrastive loss, where our sampling module acts as a plug-and-play component without modifying the encoder architecture or objective function.

In the following, we deepen into each component of the framework. Before describing them, we also provide some background notions and notation regarding graph learning methods. A visual representation of the complete WLGCL pipeline is provided in Figure 1.

3.1 Background on Graph Learning

Without loss of generality, in this work, we consider undirected graphs with node features. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph, where $v \in \mathcal{V}$ and $(v, u) \in \mathcal{E}$ represent a node and an edge within the graph, respectively. We indicate with $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times F}$ the vector of original node features, where $\mathbf{x}_v \in \mathbb{R}^F$ stands for the features of node $v \in \mathcal{V}$. Then, we use $\mathcal{P}_v$ to represent the set of one-hop neighbors of node v, such that $\mathcal{P}_v = \{p \in \mathcal{V} \colon (v, p) \in \mathcal{E}\}$. Following the typical graph contrastive learning (GCL) terminology, $\mathcal{P}_v$ is also defined as the set of *positive* nodes for the anchor node v, as opposed to its *negative* nodes $\mathcal{N}_v = \{n \in \mathcal{V} \colon (v, n) \notin \mathcal{E}\}$. Apart from these general definitions, in the following sections, we will introduce specific cases of positive and negatives samples for node v. In common graph learning tasks (such as node classification or link prediction), we seek to learn an encoding function over $\mathcal{G}$ to perform the downstream task (such as classifying node labels or predicting the existence of an edge between two nodes). For instance, message passing graph neural networks (GNNs) constitute powerful encoders that map $\mathcal{G}$ to latent node features $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}| \times D}$ such that $\mathbf{H} = \mathrm{GNN}(\mathcal{G}, \mathbf{X})$, where $\mathbf{h}_v \in \mathbb{R}^D$ is the node embedding updated after the message passing with the GNN.

3.2 WL Tree Refinement Process

We employ the Weisfeiler-Leman (WL) refinement procedure to derive structure-aware node similarities used in our dual-view contrastive sampling framework. For each node $v \in \mathcal{V}$, we initialize it with a color $C_v^{(0)}$, which can be derived from node attributes or assigned uniformly when no features are available. The WL algorithm iteratively refines node colors by aggregating the multiset of colors of neighboring nodes.

Formally, the coloring at iteration t partitions the set of nodes $\mathcal{V}$ into color classes, in the sense that nodes assigned the same color belong to the same class. The new color of a node v at iteration t is given by:

$$C_v^{(t)} = (C_v^{(t-1)}, \{\!\{C_u^{(t-1)} \colon u \in \mathcal{P}_v\}\!\}) \quad t \in \{1, 2, \ldots, T\}. \tag{1}$$

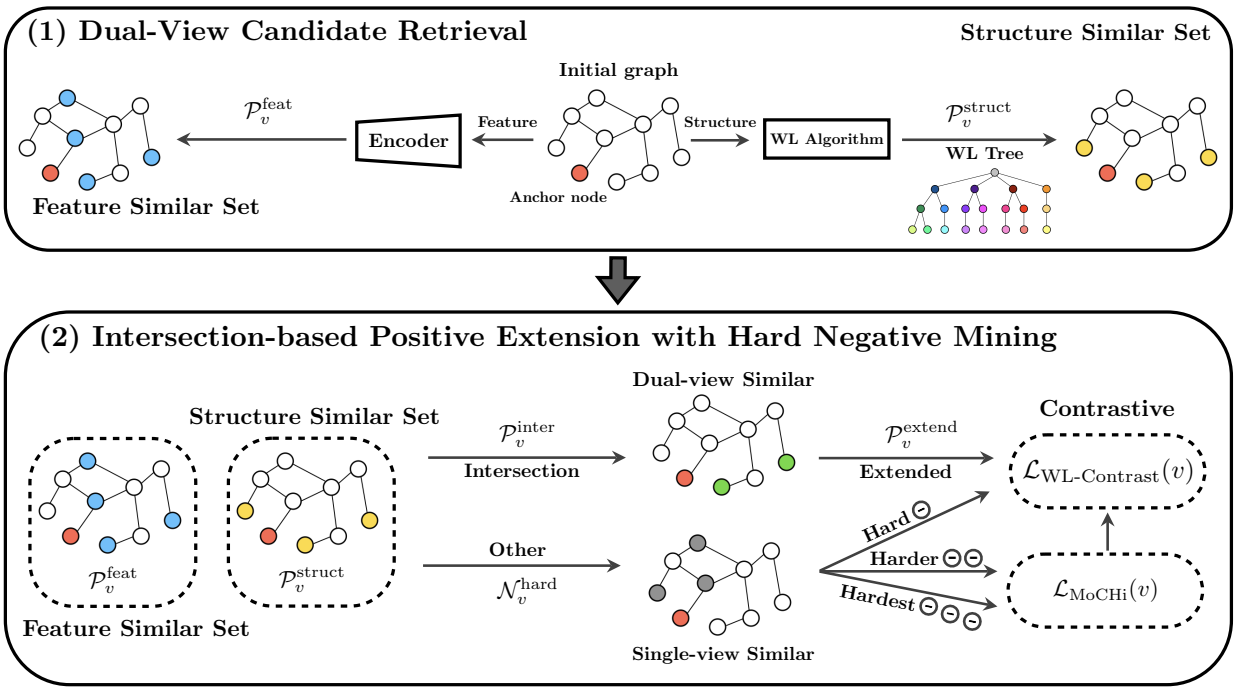

Figure 1: Overview of WLGCL, consisting of two main stages. In the first stage (above), WLGCL retrieves candidate neighbors from the feature embedding and WL structural spaces (left and right, respectively). Then, in the second stage (below), WLGCL computes the intersection of the feature and structure similar sets to obtain a reliable set of positive nodes, taking nodes appearing in only one of the two sets as hard negatives; all such node sets are used to optimize the model's contrastive loss function, involving also a MoCHi-based (Kalantidis et al., 2020) loss component requiring additional levels of harder and hardest negative node samples.

Hence, the new color of v depends on its previous color and on the multiset of the colors of its neighbors in the previous iteration. It is then clear that in iteration t, two vertices v and u receive different colors if they already had different colors in iteration $t-1$ or if the multisets of their neighbors' colors at iteration $t-1$ differ. Since the implication $C_v^{(t-1)} \neq C_u^{(t-1)} \Rightarrow C_v^{(t)} \neq C_u^{(t)}$ holds for all $v, u \in \mathcal{V}$ and any $t \in \mathbb{N}$, the color classes produced in iteration t are at least as fine as those produced in iteration $t-1$. That is, if two nodes belonged to different color classes in iteration $t-1$ of the algorithm, they will surely belong to different classes in the next iteration. Conversely, if they belonged to the same color class in iteration $t-1$, they may or may not be assigned to the same class in iteration t. Hence, the sequence of colorings produced across WL iterations induces a family of nested subsets, which can naturally be represented by a hierarchy.

Figure 2 illustrates a graph together with the colors produced by three iterations of the WL algorithm (from top to bottom). It also shows the hierarchy (WL tree) induced by the refinement process. After three iterations, there are eight color classes, each containing either a single node or two nodes of the input graph. Nodes that remained in the same color class across multiple WL iterations are more structurally similar than nodes that never shared a color class or remained in the same color class across less WL iterations.

3.3 Dual-View Candidate Retrieval

Given an anchor node v, the first stage of our framework independently retrieves two candidate sets of semantically related nodes: one based on feature similarity in the embedding space and the other based on structural similarity in the original graph topology. This separation is crucial, as the former reflects what

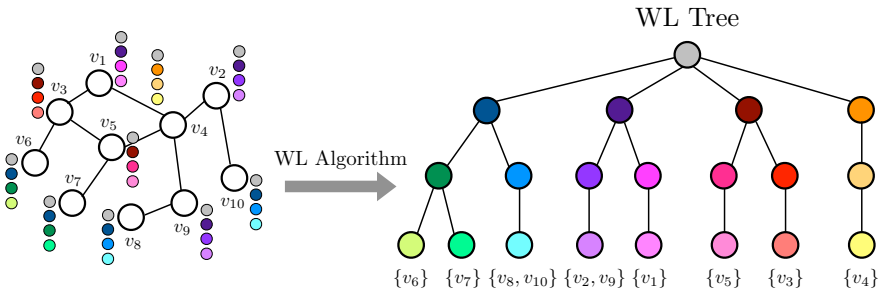

Figure 2: WL algorithm example: 1) On the left side is a graph $\mathcal{G}$ with uniform initial colors and refined colors (ordered from top to bottom) with iteration $t = 3$; 2) On the right side is the corresponding WL tree of the graph.

the encoder currently assumes to be similar, while the latter reflects intrinsic structural patterns that are independent of representation learning.

Feature-based similarity selection. First, we encode the input graph $\mathcal{G}$ along its node features through a GNN encoder to obtain node embeddings $\mathbf{H} = \text{GNN}(\mathcal{G}, \mathbf{X})$. While we maintain that our framework is technically encoder-agnostic and could be seamlessly combined with other GNN backbones, we adopt WLHN as the default encoder in our implementation due to its strong capability in capturing higher-order structural patterns and preserving multi-scale neighborhood information.

For each anchor node (marked **red** in Figure 1), we compute the cosine similarity to all other nodes:

$$S_{\text{feat}}(v, u) = \cos(\mathbf{h}_v, \mathbf{h}_u) = \frac{\mathbf{h}_v^\top \mathbf{h}_u}{\|\mathbf{h}_v\| \cdot \|\mathbf{h}_u\|}. \tag{2}$$

On such a basis, we establish a pre-defined threshold θ_{feat} according to which nodes with feature-based similarity S_{feat} above it will belong to the feature-consistent candidate set (marked **blue** in Figure 1):

$$\mathcal{P}_v^{\text{feat}} = \{u \colon S_{\text{feat}}(v, u) > \theta_{\text{feat}}\}, \quad \forall v \in \mathcal{V}. \tag{3}$$

In the equation above, $\mathcal{P}_v^{\text{feat}}$ indicates the set of nodes whose GNN encoded features are similar under a certain threshold θ_{feat}. Setting a suitable threshold value allows a finer selection of the most similar node embeddings to the anchor one.

Structure-based similarity selection. At the same time, we measure the structural similarity using a Weisfeiler-Leman (WL) refinement process. Given a graph $\mathcal{G}$, we perform T iterations of WL refinement and construct a WL tree, where each node $v \in \mathcal{V}$ is mapped to a unique leaf node after T iterations. Each leaf corresponds to the final refined WL label that encodes the rooted T-hop neighborhood structure of the node.

We define the WL-tree distance between two nodes v and u as the shortest path length between their corresponding leaf nodes in the WL tree:

$$S_{\text{struct}}(v, u) = \text{sp}(C_v^{(T)}, C_u^{(T)}), \tag{4}$$

where $\text{sp}(\cdot)$ is the function computing the shortest path distance between the two leaf nodes v and u within the T-iterations WL-tree. A smaller distance indicates higher structural similarity, as the two nodes share more common refinement history, while a larger distance implies greater structural dissimilarity.

Based on this distance, and similarly to the feature-based similarity case, we retain the most similar candidate nodes according to the structural similarity (marked in **yellow** in Figure 1) as:

$$\mathcal{P}_v^{\text{struct}} = \{u \colon S_{\text{struct}}(v, u) < \theta_{\text{struct}}\} \quad \forall v \in \mathcal{V}, \tag{5}$$

where $\theta_{\text{struct}} > 0$ is a predefined threshold controlling the maximum admissable WL-tree distance. In practice, θ_{struct} is bounded by the maximum depth of the WL tree (twice of the maximum depth) and is selected as a hyperparameter based on validation performance. Since this criterion depends solely on the graph topology encoded by WL refinement, the resulting candidate set is encoder-agnostic and robust to feature noise and representation oversmoothing.

3.4 Intersection-Based Positive Extension and Hard Negative Mining

In standard GCL, each anchor node in one augmentation view has a single positive, namely, its counterpart in the other view that corresponds to the same underlying node. Negatives are drawn from all other nodes. In contrast, WLGCL extends the positive set by exploiting the dual-view consistency discovered during the candidate retrieval described above.

Given the feature-consistent candidate set $\mathcal{P}_v^{\text{feat}}$ of (3) and the structure-consistent candidate set $\mathcal{P}_v^{\text{struct}}$ of (5), we first compute their intersection:

$$\mathcal{P}_v^{\text{inter}} = \mathcal{P}_v^{\text{feat}} \cap \mathcal{P}_v^{\text{struct}} \quad \forall v \in \mathcal{V}, \tag{6}$$

where $\mathcal{P}_v^{\text{inter}}$ contains nodes simultaneously close to the anchor in the embedding space and in the WL-structural space (marked **green** ● in Figure 1). Then, we combine the intersection with the original positive set to form an extended positive set for anchor as:

$$\mathcal{P}_v^{\text{extend}} = \mathcal{P}_v \cup \mathcal{P}_v^{\text{inter}} \quad \forall v \in \mathcal{V}. \tag{7}$$

This converts the single-positive regime into a multi-positive regime, while keeping all positives cross-view to avoid trivial same-view matching.

Nodes that satisfy only one of the two criteria for similarity based on features or structure are designated as hard negatives:

$$\mathcal{N}_v^{\text{hard}} = (\mathcal{P}_v^{\text{feat}} \cup \mathcal{P}_v^{\text{struct}}) \setminus \mathcal{P}_v^{\text{inter}} \quad \forall v \in \mathcal{V}. \tag{8}$$

These nodes (marked **gray** ● in Figure 1) are semantically harder than random negatives and naturally reflect misleading similarities, which makes them especially valuable for contrastive learning.

Finally, the extended positive set $\mathcal{P}_v^{\text{extend}}$ and the hard negative set $\mathcal{N}_v^{\text{hard}}$ are fed into our contrastive loss (described in the next section), where the extended sample space strengthens both the alignment and the uniformity properties of the learned representation.

3.5 Contrastive Objective with WL-Contrast

A crucial challenge in contrastive learning is the construction of informative negative samples. In InfoNCE-style (van den Oord et al., 2018) objectives, easy negatives contribute little to the gradient signal, whereas overly difficult ones may lead to training collapse. Prior work has explored enriching the negative set with harder samples, for instance, by synthesizing interpolated representations in the embedding space as in MoCHi (Kalantidis et al., 2020). In particular, given an anchor node v, a subset of hard negatives is first identified, and additional harder and hardest negatives are generated by linearly interpolating pairs of embeddings (e.g., hard-hard and anchor-hard pairs), yielding progressively more challenging samples.

In this work, we introduce WL-Contrast, a structure-aware contrastive objective tailored to dual-view graph representations. Our formulation departs from existing approaches in two key aspects. First, we move beyond the single-positive assumption by defining a multi-positive set $\mathcal{P}_v^{\text{extend}}$ (Eq. (7)), capturing nodes that are jointly consistent across both feature and structural views. Second, we propose a semantically grounded notion of hard negatives $\mathcal{N}_v^{\text{hard}}$ (Eq. (8)), defined as nodes that are similar in one view (feature or structure) but dissimilar in the other. Inspired by MoCHi, we construct progressively more challenging negatives via interpolation. Specifically, we generate additional sets of harder negatives, denoted as $\mathcal{N}_v^{\text{harder}}[k]$ and $\mathcal{N}_v^{\text{hardest}}[k']$, where k and k' indicate the selected number of harder and hardest negatives, respectively. These sets extend the initial hard negative pool and further strengthen the contrastive signal. Our proposed

version of the WL-Contrast loss function becomes:

$$\mathcal{L}_{\text{WL-Contrast}}(v) = \frac{1}{|\mathcal{P}_v^{\text{extend}}|} \sum_{p \in \mathcal{P}_v^{\text{extend}}} -\log \frac{\exp(\cos(\mathbf{h}_v, \mathbf{h}_p)/\tau)}{\exp(\cos(\mathbf{h}_v, \mathbf{h}_p)/\tau) + \sum_{n \in \mathcal{N}_v^{\text{hard}}} \exp(\cos(\mathbf{h}_v, \mathbf{h}_n)/\tau) + \mathcal{L}_{\text{MoCHi}}(v)}, \tag{9}$$

where $\exp(\cdot)$ is the exponential function, τ is a temperature parameter, and $\mathcal{L}_{\text{MoCHi}}(v)$ is calculated as:

$$\mathcal{L}_{\text{MoCHi}}(v) = \sum_{n \in \mathcal{N}_v^{\text{harder}}[k]} \exp(\cos(\mathbf{h}_v, \mathbf{h}_n)/\tau) + \sum_{n \in \mathcal{N}_v^{\text{hardest}}[k']} \exp(\cos(\mathbf{h}_v, \mathbf{h}_n)/\tau). \tag{10}$$

By defining hard negatives via cross-view inconsistency, our approach yields more informative and structure-aware negative samples, reducing sampling bias and improving contrastive discrimination. Together with the multi-positive formulation, this leads to a stronger and more stable self-supervised representation.

For a technical overview of our proposed WLGCL method, we report all steps involved in the pipeline in Algorithm 1.

3.6 Theoretical Insight and Complexity Analysis

From a geometric viewpoint, our dual-view sampling mechanism improves the fundamental trade-off between alignment and uniformity in contrastive learning (Wang & Isola, 2020). Conventional GCL models often emphasize alignment by pulling together nodes that are close in the learned embedding space, but these pairs may not be structurally consistent, leading to over-clustered embeddings and a loss of generalization. In contrast, our intersection-based positive selection enforces cross-view consistency between feature and structure, ensuring that only semantically reliable pairs contribute to alignment. At the same time, nodes that are similar in one view but dissimilar in the other are treated as hard negatives, which enhances uniformity by preserving structural diversity in the representation space. This balanced mechanism mitigates over-smoothing and false-positive alignment, resulting in embeddings that are both discriminative and topology-aware.

In terms of computational properties, the proposed framework introduces minimal additional cost and remains fully compatible with standard GCL training. The structure-based similarity derived from the WL refinement is precomputed only once before training and reused throughout all epochs, scales linearly with the number of nodes $\mathcal{V}$ and edges $\mathcal{E}$, which is $\mathcal{O}(\|\mathcal{V}\| + \|\mathcal{E}\|)$. The feature-based similarity, on the other hand, must be updated dynamically at each epoch, since it depends on the evolving encoder representations. However, this operation involves only similarity computation between node embeddings and can be efficiently parallelized within each mini-batch, with the time complexity as $\mathcal{O}(\|\mathcal{V}\| \cdot \log \|\mathcal{V}\|)$ for each epoch. Importantly, our sampling module is model-agnostic and can be seamlessly integrated into any GCL backbone. Overall, the method achieves a favorable balance between theoretical robustness, semantic reliability, and practical scalability, providing an efficient and general plug-in for structure-aware GCL.

4 Experiments

4.1 Experimental Settings

Baselines. We conduct a comprehensive set of experiments to evaluate our proposed model against seven representative GCL baselines: DGI (Velickovic et al., 2019), GRACE (Zhu et al., 2020), BGRL (Thakoor et al., 2022), MUSE (Yuan et al., 2023), EPAGCL (Xu et al., 2025), FOSSIL (Sangare et al., 2025), and PolyGCL (Chen et al., 2024). These methods cover a broad spectrum of contrastive objectives, augmentation strategies, and structure-aware designs, providing a diverse and competitive comparison. Table 1 gives a brief summary and comparison between the baseline models and our proposed method. For all baselines, we use the default hyperparameters provided in their code.

Datasets. Experiments are performed on three homophilic datasets, Cora (Yang et al., 2016), CiteSeer (Yang et al., 2016), and Amazon-Photo (Shchur et al., 2018), as well as three heterophilic datasets:

Algorithm 1 WLGCL overall pipeline.

Input: Graph data $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, node features $\mathbf{X} \in \mathbb{R}^{|V| \times F}$, GNN encoder, iterations number for WL tree T, feature-based threshold θ_{feat}, structure-based threshold θ_{struct}, top-k value k.
Output: Loss function value $\mathcal{L}_{\text{WL-Contrast}}(v)$.
`# WL Tree Refinement Process`
for $t \in \{1, 2, \dots T\}$ **do**
  Compute $C_v^{(t)}$ according to (1).
end for
`# Dual-View Candidate Retrieval`
Given θ_{feat}, compute $\mathcal{P}_v^{\text{feat}}$ according to (2) and (3).
Given θ_{struct} and $C_v^{(T)}$, compute $\mathcal{P}_v^{\text{struct}}$ according to (4) and (5).
`# Intersection-Based Positive Extension and Hard Negative Mining`
Compute $\mathcal{P}_v^{\text{extend}}$ according to (6) and (7).
Compute $\mathcal{N}_v^{\text{hard}}$ according to (8).
Given k, compute $\mathcal{N}_v^{\text{harder}}$ by linear interpolating anchor and negative samples.
Given k', compute $\mathcal{N}_v^{\text{hardest}}$ by linear interpolating two random negative samples.
`# Loss Function Computation`
Calculate $\mathcal{L}_{\text{WL-Contrast}}(v)$ according to (9) and (10).
Return: $\mathcal{L}_{\text{WL-Contrast}}(v)$.

Table 1: High-level comparison of representative graph contrastive learning baselines.

Model	Contrast Type	View Strategy	Negatives	Contrast Scale
DGI (Velickovic et al., 2019)	MI-based	Global vs local	✓	Node / Graph
GRACE (Zhu et al., 2020)	InfoNCE-based	Perturbation-based	✓	Node
BGRL (Thakoor et al., 2022)	Bootstrap	Perturbation-based	✗	Node
MUSE (Yuan et al., 2023)	Hybrid (contrast + recon.)	Feature-based	✓	Node
PolyGCL (Chen et al., 2024)	Spectral contrast	Frequency-domain	✓	Node
EPAGCL (Xu et al., 2025)	Contrastive regularization	Topology-based	✓	Node
FOSSIL (Sangare et al., 2025)	Structural alignment	Subgraph matching	✗	Subgraph
WLGCL	Structure-aware	Dual-view (feature + WL)	✓	Node

Table 2: Statistics of the datasets used.

	Cora	CiteSeer	Amazon-Photo	Actor	Squirrel	Chameleon
Nodes	2,708	3,327	7,650	7,600	5,201	2,277
Edges	10,556	9,104	238,162	30,019	217,073	36,101
Features	1,433	3,703	745	932	2,089	2,325
Classes	7	6	8	5	5	5

Actor (Pei et al., 2020), Squirrel (Rozemberczki et al., 2021), and Chameleon (Rozemberczki et al., 2021). These together enable a thorough assessment of performance under varying levels of structural alignment. The statistics of the datasets we used are demonstrated in Table 2.

Training and test paradigms. For all our experiments, we adopt the WLHN (Nikolentzos et al., 2023) as the GNN backbone encoder architecture, using 4 layers and a hidden dimension of 64 as suggested in their paper. The feature-similarity and structure-similarity thresholds θ_{feat} and θ_{struct}, respectively, are tuned individually for each dataset to account for differences in graph density, feature quality, and homophily level. All models are trained and evaluated under identical conditions. Performance is assessed via downstream node classification accuracy, using a logistic regression classifier trained on the learned node embeddings. Importantly, the train/test masks are used only for this downstream evaluation and play no role in training the embedding model itself. The representation learning phase is entirely self-supervised: the GNN is trained

Table 3: Node classification accuracy (± std) comparison across six benchmark datasets over 5 runs. The best results are highlighted in **bold**, while the second best-performing methods are underlined. The improvement is calculated base on WLGCL and the second best-performing method.

Model	Homophilic			Heterophilic		
	Cora	CiteSeer	Amazon-Photo	Actor	Squirrel	Chameleon
GRACE	$0.671_{\pm 0.025}$	$0.558_{\pm 0.009}$	$0.709_{\pm 0.048}$	$0.258_{\pm 0.014}$	$0.309_{\pm 0.013}$	$0.383_{\pm 0.026}$
DGI	$0.704_{\pm 0.055}$	$0.535_{\pm 0.033}$	$0.628_{\pm 0.044}$	$0.256_{\pm 0.015}$	$0.308_{\pm 0.032}$	$0.397_{\pm 0.067}$
BGRL	$0.706_{\pm 0.010}$	$0.598_{\pm 0.015}$	$0.729_{\pm 0.032}$	$0.240_{\pm 0.050}$	$0.306_{\pm 0.033}$	$0.401_{\pm 0.071}$
MUSE	$0.673_{\pm 0.033}$	$0.603_{\pm 0.023}$	$0.736_{\pm 0.028}$	$0.279_{\pm 0.034}$	$0.294_{\pm 0.045}$	$0.400_{\pm 0.103}$
EPAGCL	$0.750_{\pm 0.020}$	$0.644_{\pm 0.022}$	$0.714_{\pm 0.039}$	$0.272_{\pm 0.007}$	$0.323_{\pm 0.017}$	$0.320_{\pm 0.049}$
FOSSIL	$\underline{0.758}_{\pm 0.006}$	$0.634_{\pm 0.010}$	$\underline{0.761}_{\pm 0.024}$	$\underline{0.343}_{\pm 0.003}$	$\underline{0.344}_{\pm 0.014}$	$\underline{0.432}_{\pm 0.029}$
PolyGCL	$0.751_{\pm 0.011}$	$\underline{0.665}_{\pm 0.018}$	$0.758_{\pm 0.031}$	$0.321_{\pm 0.009}$	$0.336_{\pm 0.010}$	$0.399_{\pm 0.020}$
WLGCL	$\mathbf{0.764}_{\pm 0.013}$	$\mathbf{0.678}_{\pm 0.010}$	$\mathbf{0.774}_{\pm 0.034}$	$\mathbf{0.358}_{\pm 0.014}$	$\mathbf{0.357}_{\pm 0.024}$	$\mathbf{0.452}_{\pm 0.036}$
Improvement	+0.79%	+1.95%	+1.71%	+4.37%	+3.78%	+4.63%

solely to learn node embeddings, without using any label information or predefined data splits. When datasets provide predefined masks, these are adopted as-is for the evaluation stage.

For datasets without such masks, we randomly sample 20% of the nodes as the training set and use the remaining 80% for testing, with identical splits across all methods and random seeds. Every experiment was repeated five times with different random seeds and we report the mean value and standard deviation for the performance. Our experiments are conducted on a machine equipped with an NVIDIA RTX A5000 GPU, ensuring consistency and reproducibility across all baselines and datasets.

4.2 Results and Discussion

Table 3 reports the classification accuracy for node classification across six benchmark datasets. Our proposed WLGCL consistently matches or surpasses the strongest baseline on every dataset, demonstrating the effectiveness of jointly enforcing feature-structure consistency in contrastive learning. On the homophilic graphs Cora, CiteSeer, and Amazon-Photo, WLGCL yields clear improvements over the best-performing baselines. On Cora, WLGCL achieves an accuracy of 0.764, outperforming FOSSIL (0.758) by +0.79%. On CiteSeer, WLGCL reaches 0.678, surpassing PolyGCL (0.665) by +1.95%. On Amazon-Photo, the improvement is even more pronounced, outperforming FOSSIL (0.761) by +1.71%. These gains show that even in feature-homophilic settings, incorporating structural consistency leads to better positive/negative sample construction and more discriminative embeddings.

The advantages become even clearer on heterophilic datasets, where feature-based similarity alone is unreliable for sampling due to the mismatch between node features and structural roles. Across three heterogeneous datasets, our model achieved an improvement of approximately 4%. These substantial improvements can be attributed to the fact that, in heterophilic graphs, structurally similar nodes often exhibit dissimilar features, causing feature-only methods to produce misleading positives and weak negatives. By explicitly incorporating WL-based structural similarity, our framework is able to recover such structurally consistent relationships, while the dual-view sampling strategy filters out inconsistent pairs and identifies informative hard negatives. This leads to more reliable contrastive signals and ultimately stronger representation learning when topology carries more semantic information than raw features.

Overall, the results demonstrate that our structure-informed multi-positive sampling framework provides consistent and often significant performance gains over both classical and state-of-the-art GCL baselines.

4.3 Ablation Study

Impact of encoder choice. To analyze the importance of the encoder architecture in our framework, we compare the performance of four different GNN backbones: GCN (Kipf & Welling, 2017), GIN (Xu et al.,

Table 4: Node classification accuracy (± std) of different encoders across six benchmark datasets over 5 runs.

Encoder	Homophilic			Heterophilic		
	Cora	CiteSeer	Amazon-Photo	Actor	Squirrel	Chameleon
GCN	$\underline{0.735}_{\pm 0.010}$	$0.568_{\pm 0.025}$	$0.770_{\pm 0.023}$	$0.306_{\pm 0.014}$	$\underline{0.355}_{\pm 0.024}$	$0.415_{\pm 0.026}$
GIN	$0.711_{\pm 0.021}$	$\underline{0.577}_{\pm 0.029}$	$0.755_{\pm 0.032}$	$0.290_{\pm 0.018}$	$0.337_{\pm 0.033}$	$0.398_{\pm 0.088}$
GAT	$0.725_{\pm 0.010}$	$0.559_{\pm 0.028}$	$\underline{0.772}_{\pm 0.032}$	$\underline{0.321}_{\pm 0.011}$	$0.325_{\pm 0.054}$	$\underline{0.423}_{\pm 0.057}$
WLHN	$\mathbf{0.764}_{\pm 0.013}$	$\mathbf{0.678}_{\pm 0.010}$	$\mathbf{0.774}_{\pm 0.034}$	$\mathbf{0.358}_{\pm 0.006}$	$\mathbf{0.357}_{\pm 0.024}$	$\mathbf{0.435}_{\pm 0.036}$

2019), GAT (Velickovic et al., 2018), and WLHN (Nikolentzos et al., 2023), which is our chosen backbone. All baseline encoders are configured with two layers, hidden dimension 64, and output dimension 32 to ensure a fair comparison, while WLHN follows the parameters described in Section 4.1. These encoders are integrated into our dual-view contrastive learning pipeline without modifying any other component.

Table 4 reports the node classification accuracy across six datasets. The results show that WLHN consistently outperforms the other encoders on all datasets, demonstrating its advantage in capturing structural nuances that are crucial to our feature-structure intersection mechanism. For example, on heterophilic datasets such as Actor, Squirrel, and Chameleon, WLHN achieves accuracy improvements of +11.9%, +9.8%, and +5.3% over the strongest alternative encoder, respectively. Even on homophilic datasets (Cora, CiteSeer, Amazon-Photo), WLHN provides noticeable gains, achieving the highest accuracy in all cases.

These results validate that the choice of encoder plays a critical role in the effectiveness of WLGCL. Because WLHN generates richer structural representations and preserves multi-scale neighborhood information, it aligns naturally with our structure-aware positive and hard negative sampling strategy, leading to superior contrastive representations. Importantly, we also observe that the proposed framework consistently achieves strong performance relative to the baselines on heterophilic datasets across different encoder choices. This suggests that the observed improvements are primarily attributed to the effectiveness of the dual-view contrastive framework itself, rather than being dependent on a specific encoder architecture.

Effect of loss functions design. We further investigate the role of the contrastive objective by comparing three loss function variants under the same framework and a fixed WLHN encoder. The results are summarized in Table 5. The first variant uses the standard InfoNCE loss (van den Oord et al., 2018), where each anchor node is paired with only one single positive (its counterpart in the alternative augmentation) and all remaining nodes serve as ordinary negatives. This baseline provides a reference point for evaluating the effect of positive expansion and hard negative sampling. The second variant replaces InfoNCE with MoCHi (Kalantidis et al., 2020) loss introduced in Section 3.5, which augments the negative set by synthesizing harder negatives through interpolation in the embedding space. Although it improves performance across all datasets compared to InfoNCE, it remains limited by its reliance on feature similarity alone for hard negative construction. Finally, our proposed WL-Contrast loss integrates both multi-positive semantics and dual-view hard negatives. By treating the intersection of feature-similar and structure-similar nodes as additional positives, the loss encourages alignment across both latent and structural spaces. Likewise, nodes that satisfy only one of the two similarity constraints form semantically meaningful hard negatives. As shown in the table, our loss consistently achieves the highest accuracy on all datasets, while also exhibiting the lowest variance across multiple runs. These results highlight the importance of incorporating structural consistency into both the positive and negative components of the contrastive objective.

4.4 Time Efficiency Trade-off

To further analyze the practicality of our method, we evaluate its time efficiency relative to seven representative GCL baselines. Figure 3 reports the trade-off between accuracy and training time per epoch (in seconds) on four datasets. Models such as GRACE, DGI, and BGRL achieve fast training speed due to their lightweight objectives, but their accuracy is considerably lower, especially on heterophilic graphs. In con-

Table 5: Node classification accuracy (± std) of different loss function design on four datasets over 5 runs. EP means extended positives, HN_{feat} means hard feature negatives, $\text{HN}_{\text{struct}}$ means hard structure negatives.

	EP	HN_{feat}	$\text{HN}_{\text{struct}}$	**Cora**	**CiteSeer**	**Actor**	**Squirrel**
InfoNCE	✗	✗	✗	$0.732_{\pm 0.023}$	$0.616_{\pm 0.026}$	$0.318_{\pm 0.043}$	$0.338_{\pm 0.067}$
MoCHi	✗	✓	✗	$0.745_{\pm 0.019}$	$0.643_{\pm 0.022}$	$0.334_{\pm 0.049}$	$0.347_{\pm 0.066}$
WL-Contrast	✓	✓	✓	$\mathbf{0.764}_{\pm 0.013}$	$\mathbf{0.678}_{\pm 0.010}$	$\mathbf{0.358}_{\pm 0.006}$	$\mathbf{0.357}_{\pm 0.024}$

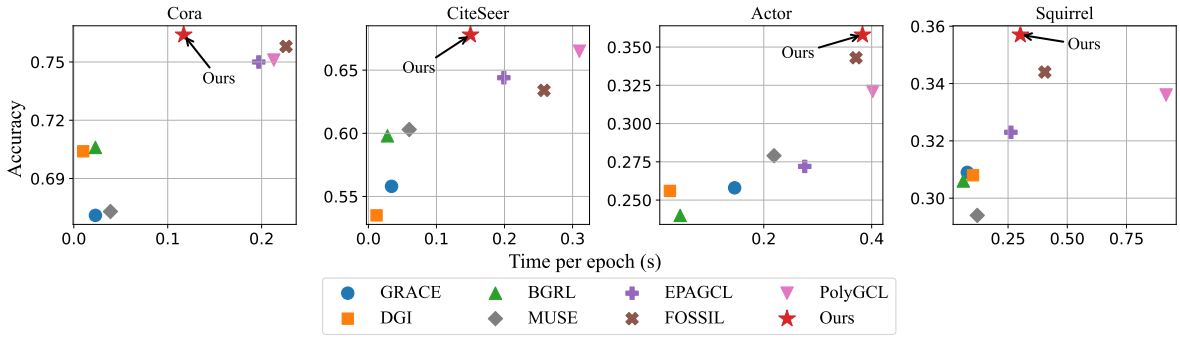

Figure 3: Time–accuracy comparison across four datasets. WLGCL ("Ours") attains higher accuracy with competitive computational cost.

trast, methods that incorporate more complex augmentations or contrastive processing, such as EPAGCL, PolyGCL, FOSSIL, obtain higher accuracy but require substantially longer training time.

Across all datasets, WLGCL achieves a favorable balance between computational cost and predictive performance. Although the dual-view sampling introduces moderate overhead, primarily from computing feature similarity each epoch, our framework remains competitive in runtime while consistently delivering the highest accuracy. Notably, on Cora and CiteSeer, WLGCL lies close to the Pareto frontier: it outperforms EPAGCL and PolyGCL by a large margin in accuracy while taking less time per epoch. The efficiency gain is even more evident on heterophilic datasets such as Actor and Squirrel, where our model achieves the best accuracy with only modest additional cost compared to the strongest baselines.

It is also important to note that the structure-based similarity branch, implemented through WL tree construction, is computed only once during a preprocessing stage. This cost is therefore not included in the per-epoch measurements shown in Figure 3. In practice, this preprocessing step is lightweight: its runtime is roughly equivalent to a single training epoch (0.15s on Cora, 0.21s on CiteSeer, 0.20s on Actor, and 0.15s on Squirrel). Because this cost is paid only once at the beginning of training, its amortized impact on overall efficiency is negligible. Overall, these results suggest that the proposed WLGCL provides significant accuracy improvements without incurring prohibitive computational overhead, making it suitable for real-world graph learning scenarios where both efficiency and performance are crucial.

4.5 Hyper-parameter Sensitivity

Impact of feature similarity space. We also examine the impact of different strategies for computing feature similarity in the dual-view sampling process. Specifically, we compare two variants: (i) Graph-space similarity, where node similarity is computed directly from the raw input features; (ii) Embedding-space similarity, where similarity is measured in the latent space produced by the WLHN encoder. Both variants share the same structure-based similarity branch and hard-negative construction, isolating the effect of the feature-similarity module. As shown in Table 6, using embedding-space similarity consistently leads to higher performance across all datasets. This behavior is expected: raw features may be noisy, sparse, or weakly aligned with structural patterns, making similarity estimates unreliable. In contrast, similarities measured in the learned latent space capture more discriminative and structure-aware representations, resulting in

Table 6: Node classification accuracy (± std) of different feature similarity space over 5 runs.

Feature Space	**Cora**	**CiteSeer**	**Amazon-Photo**	**Actor**	**Squirrel**	**Chameleon**
Graph Space	$0.759_{\pm 0.017}$	$0.670_{\pm 0.021}$	$0.760_{\pm 0.027}$	$0.351_{\pm 0.014}$	$0.349_{\pm 0.012}$	$0439_{\pm 0.028}$
Embedding Space	$\mathbf{0.764}_{\pm 0.013}$	$\mathbf{0.678}_{\pm 0.010}$	$\mathbf{0.774}_{\pm 0.034}$	$\mathbf{0.358}_{\pm 0.014}$	$\mathbf{0.357}_{\pm 0.024}$	$\mathbf{0.452}_{\pm 0.036}$
Improvement	+0.66%	+1.19%	+2.24%	+1.99%	+2.29%	+2.96%

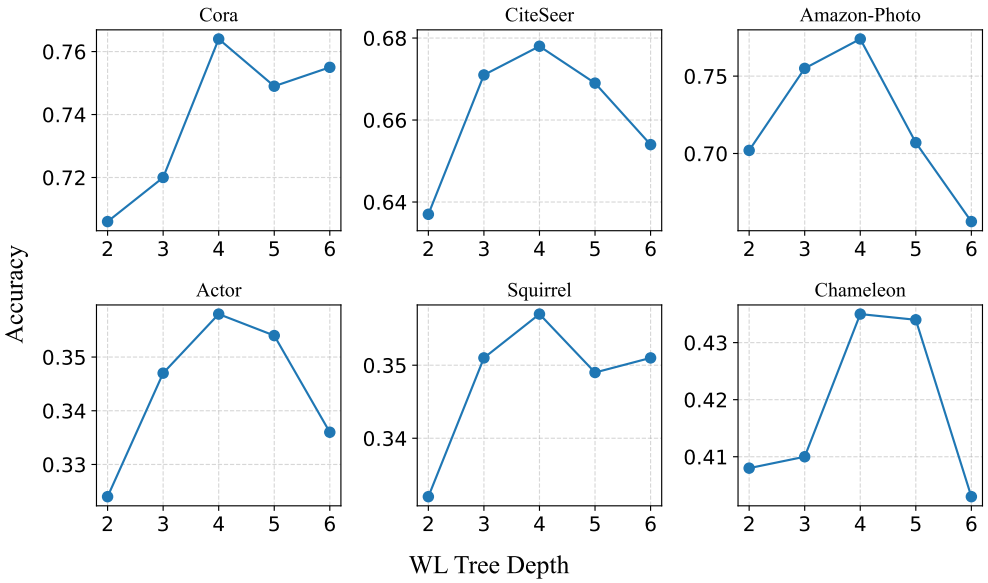

Figure 4: Node classification accuracy under different WL tree depths across six datasets.

more accurate positive selection and more meaningful hard negatives. Overall, these results confirm that embedding-space similarity provides a stronger semantic signal, thereby enhancing the effectiveness of our dual-view contrastive framework.

Effect of WL tree depth. We finally investigate how the depth of the WL tree affects the structure-based similarity branch in our dual-view framework. Figure 4 reports the performance of our model when varying the WL refinement depth from two to six. We observe a consistent pattern across all datasets: accuracy improves as the depth increases from two to around four, after which performance begins to plateau or slightly decline. This trend aligns with the intuition behind WL refinement. Shallow depths capture only very local structural patterns, leading to coarse or noisy structural similarities. As a result, the structure-based similarity becomes less informative, leading to suboptimal intersections with the feature-based positives. Increasing the depth expands the receptive field and allows the WL subtree to encode richer multi-hop dependencies, enabling more meaningful structural matches and better alignment with the embedding space. However, excessively increasing the WL depth introduces a different limitation. As refinement proceeds, WL signatures become increasingly fine-grained and discriminative: nodes that differ at any smaller depth will never become identical at larger depths, and their structural distance can only increase or remain as zero. As a result, deep WL refinement fragments the node space into many highly specific structural equivalence classes, substantially reducing the number of structurally similar node pairs. This sparsification of structural positives limits their intersection with feature-based positives and weakens the contrastive learning signal between the two views. Overall, a WL depth of approximately four provides the best balance between local and global structural information and yields the strongest performance within our dual-view contrastive learning framework.

5 Conclusion and Future Work

In this work, we introduced WLGCL, a dual-view graph contrastive learning framework that explicitly integrates feature similarity and WL-based structural similarity. By disentangling similarity into two complementary perspectives, our method constructs a more discriminative set of positives through feature-structure intersection, and generates semantically meaningful hard negatives through cross-view inconsistency. Combined with the WL-Contrast objective, WLGCL achieves more robust alignment and better uniformity in the latent space. Extensive experiments on both homophilic and heterophilic datasets demonstrate that our method consistently outperforms state-of-the-art GCL models in accuracy while maintaining competitive training efficiency. Further ablation studies and hyper-parameter sensitivity analyses confirm the benefits of dual-view sampling, multi-positive contrastive objectives, embedding-space similarity, and WL tree depth optimization.

Looking forward, several promising directions remain for future work. First, while WLGCL relies on WL refinement for structural similarity, more expressive structural priors such as motif-based patterns or optimal transport distances could be explored to further enrich structure-aware contrastive sampling. Second, the proposed dual-view framework can be extended beyond node-level learning to subgraph- and graph-level representation learning, enabling applications in molecular graphs and other structured domains. Finally, improving the scalability of WL-based similarity computation and establishing stronger theoretical guarantees for structure-aware contrastive objectives remain important directions for future investigation.

References

László Babai and Ludek Kucera. Canonical labelling of graphs in linear average time. In *FOCS*, pp. 39–46. IEEE Computer Society, 1979.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *NeurIPS*, 2020.

Jin-yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Comb.*, 12(4):389–410, 1992.

Jingyu Chen, Runlin Lei, and Zhewei Wei. PolyGCL: Graph constrastive learning via learnable spectral polynomial filters. In *ICLR*, 2024.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pp. 1597–1607, 2020.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In *EMNLP*, pp. 6894–6910, 2021.

Jie Gui, Tuo Chen, Jing Zhang, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. A survey on self-supervised learning: Algorithms, applications, and future trends. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(12):9052–9071, 2024.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pp. 9726–9735, 2020.

Yannis Kalantidis, Mert Bülent Sariyildiz, Noé Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *NeurIPS*, 2020.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Nils M. Kriege, Christopher Morris, Anja Rey, and Christian Sohler. A property testing framework for the theoretical expressivity of graph kernels. In *IJCAI*, pp. 2348–2354, 2018.

Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and Philip S. Yu. Graph self-supervised learning: A survey. *IEEE Trans. Knowl. Data Eng.*, 35(6):5879–5900, 2023.

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, pp. 4602–4609, 2019.

Christopher Morris, Gaurav Rattan, and Petra Mutzel. Weisfeiler and leman go sparse: Towards scalable higher-order graph embeddings. In *NeurIPS*, 2020.

Christopher Morris, Yaron Lipman, Haggai Maron, Bastian Rieck, Nils M. Kriege, Martin Grohe, Matthias Fey, and Karsten M. Borgwardt. Weisfeiler and leman go machine learning: The story so far. *J. Mach. Learn. Res.*, 24:333:1–333:59, 2023.

Giannis Nikolentzos, Michail Chatzianastasis, and Michalis Vazirgiannis. Weisfeiler and Leman go hyperbolic: Learning distance preserving node representations. In *AISTATS*, pp. 1037–1054, 2023.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *ICLR*. OpenReview.net, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, Proceedings of Machine Learning Research, pp. 8748–8763. PMLR, 2021.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *J. Complex Networks*, 9(2), 2021.

Amadou S. Sangare, Nicolas Dunou, Jhony H. Giraldo, and Fragkiskos D. Malliaros. A fused gromov-wasserstein approach to subgraph contrastive learning. *Trans. Mach. Learn. Res.*, 2025, 2025.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *CoRR*, abs/1811.05868, 2018.

Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-lehman graph kernels. *J. Mach. Learn. Res.*, 12:2539–2561, 2011.

Yuntao Shou, Xiangyong Cao, and Deyu Meng. Spegcl: Self-supervised graph spectrum contrastive learning without positive samples. *IEEE Trans. Neural Networks Learn. Syst.*, 36(11):19546–19559, 2025.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L. Dyer, Rémi Munos, Petar Velickovic, and Michal Valko. Large-scale representation learning on graphs via bootstrapping. In *ICLR*. OpenReview.net, 2022.

Nenad Tomasev, Ioana Bica, Brian McWilliams, Lars Buesing, Razvan Pascanu, Charles Blundell, and Jovana Mitrovic. Pushing the limits of self-supervised resnets: Can we outperform supervised learning without labels on imagenet? *CoRR*, abs/2201.05119, 2022.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR (Poster)*. OpenReview.net, 2018.

Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. In *ICLR (Poster)*. OpenReview.net, 2019.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9929–9939. PMLR, 2020.

Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *nti, Series*, 2(9):12–16, 1968.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*. OpenReview.net, 2019.

Yanchen Xu, Siqi Huang, Hongyuan Zhang, and Xuelong Li. Why does dropping edges usually outperform adding edges in graph contrastive learning? In *AAAI*, pp. 21824–21832. AAAI Press, 2025.

Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 40–48. JMLR.org, 2016.

Yihang Yin, Qingzhong Wang, Siyu Huang, Haoyi Xiong, and Xiang Zhang. Autogcl: Automated graph contrastive learning via learnable view generators. In *AAAI*, pp. 8892–8900. AAAI Press, 2022.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *NeurIPS*, 2020.

Mengyi Yuan, Minjie Chen, and Xiang Li. MUSE: multi-view contrastive learning for heterophilic graphs via information reconstruction. In *CIKM*, pp. 3094–3103. ACM, 2023.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *CoRR*, abs/2006.04131, 2020.