Exploring Consistency in Graph Representations: from Graph Kernels to Graph Neural Networks

Xuyuan Liu[1] **Yinghao Cai**[1] **Qihui Yang**[2] **Yujun Yan**[1]
[1]Dartmouth College
[2] University of California San Diego
`{xuyuan.liu.gr, yinghao.cai, yujun.yan}@dartmouth.edu`
`qiy009@ucsd.edu`

Abstract

Graph Neural Networks (GNNs) have emerged as a dominant approach in graph representation learning, yet they often struggle to capture consistent similarity relationships among graphs. While graph kernel methods such as the Weisfeiler-Lehman subtree (WL-subtree) and Weisfeiler-Lehman optimal assignment (WLOA) kernels are effective in capturing similarity relationships, they rely heavily on predefined kernels and lack sufficient non-linearity for more complex data patterns. Our work aims to bridge the gap between neural network methods and kernel approaches by enabling GNNs to consistently capture relational structures in their learned representations. Given the analogy between the message-passing process of GNNs and WL algorithms, we thoroughly compare and analyze the properties of WL-subtree and WLOA kernels. We find that the similarities captured by WLOA at different iterations are asymptotically consistent, ensuring that similar graphs remain similar in subsequent iterations, thereby leading to superior performance over the WL-subtree kernel. Inspired by these findings, we conjecture that maintaining consistency in the similarities of graph representations across GNN layers is crucial for capturing relational structures and improving graph classification performance. Thus, we propose a loss to enforce the similarity of graph representations to be consistent across different layers. Our empirical analysis verifies our conjecture and shows that our proposed consistency loss can significantly enhance graph classification performance across several GNN backbones on various datasets.

1 Introduction

Graph classification tasks are extensively applied across multiple domains, including chemistry [Liu et al., 2022, Xu et al., 2023], bioinformatics [Yan et al., 2019, Li et al., 2023a,b], and social network analysis [Ying et al., 2018, Wang et al., 2024]. Graph neural networks (GNNs) [Kipf and Welling, 2017, Xu et al., 2019, Velickovic et al., 2018, Huang et al., 2024] have emerged as the predominant approach for performing graph classification, owing to their ability to extract rich representations from various types of graph data. A typical GNN employs the message-passing mechanism [Gilmer et al., 2017], where node features are propagated and aggregated across connected nodes. This process effectively captures local tree structures, enabling the differentiation between various graphs. However, GNNs often struggle to preserve relational structures among graphs, resulting in inconsistent relative similarities across the layers. As shown in Figure 1, graphs with higher relative similarity in one layer may exhibit reduced similarity in the subsequent layer. This phenomenon arises from the limitations of cross-entropy loss, which fails to preserve relational structures, as it forces graphs within the same class into identical representations.

We release our code at `https://github.com/GraphmindDartmouth/Graph-consistency`

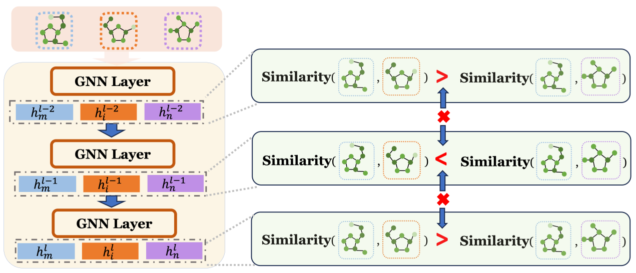


Figure 1: Cosine similarity of three molecules from the NCI1 dataset, evaluated using graph representations from three consecutive GIN layers. Common GNN models fail to preserve relational structures across the layers.

Graph kernel methods, on the other hand, are designed to capture similarities between graphs and utilize these similarities for classification tasks. For instance, subgraph-pattern approaches [Shervashidze et al., 2009, Kriege et al., 2020] compare graphs by counting the occurrences of fixed-size subgraph motifs. Other methods compare sequences of vertices or edges encountered during graph traversals [Borgwardt and Kriegel, 2005, Kashima et al., 2003, Zhang et al., 2018]. Among all graph kernels, two notable ones are the Weisfeiler-Lehman subtree (WL-subtree) kernel [Shervashidze et al., 2011] and the Weisfeiler-Lehman optimal assignment (WLOA) kernel [Kriege et al., 2016]. They are found to have comparable performance to simple GNNs [Nikolentzos et al., 2021]. The WL-subtree kernel iteratively relabels graphs using the Weisfeiler-Lehman algorithm [Weisfeiler and Lehman, 1968] and constructs a kernel based on the number of occurrences of each label. The WLOA kernel uses the same relabeling scheme but computes a matching between substructures to reveal structural correspondences between the graphs.

While effective in capturing relative graph similarity, kernel methods rely on predefined kernels and exhibit insufficient non-linearities, limiting their ability to capture complex patterns in high-dimensional data. Additionally, kernel methods are computationally costly, making them unsuitable for handling large datasets and consequently limiting their overall applicability.

In this work, we aim to bridge the gap between kernel methods and GNN models. Given the iterative nature of GNNs, we study a class of kernels which are induced from graph representations obtained through an iterative process and name them iterative graph kernels (IGK). Within this framework, we define the **consistency** property, which ensures that similar graphs remain similar in subsequent iterations. Our analysis demonstrates that kernels with this property ensure better generalization ability, leading to improved classification performance.Furthermore, we find that this property sheds light on why the WLOA kernel outperforms the WL-subtree kernel. The WLOA kernel asymptotically demonstrates consistency as the iteration goes to infinity, whereas the WL-subtree kernel does not exhibit this behavior. Inspired by these findings and the analogy between message-passing GNNs and the WL-subtree kernel [Xu et al., 2019], we hypothesize that this principle is also applicable to GNNs. To explore this, we introduce a novel loss function designed to align the ranking of graph similarities across GNN layers. The aim is to ensure that the relational structures of the graphs are preserved and consistently reflected throughout the representation space of these layers. We validate this hypothesis by applying our loss function to different GNN backbones across various graph datasets. Extensive experiments demonstrate that our proposed model-agnostic consistency loss improves graph classification performance comprehensively.

In summary, the main contributions of this work are as follows:

- **Novel perspective:** We present a novel perspective on understanding the graph classification performance of GNNs by analyzing the similarity relationships captured by different layers.
- **New insights:** We are the first to introduce and formalize the consistency principle within both kernel-based and GNN methods for graph classification tasks. Additionally, we provide theoretical proofs explaining how this principle enhances the performance.
- **Simple yet effective method:** Empirical results demonstrate that the proposed consistency loss universally enhances performance across a wide range of base models and datasets.

2 Preliminaries

In this section, we begin by introducing the notations and definitions used throughout the paper. Next, we provide an introduction to the fundamentals of Weisfeiler-Lehman isomorphism test, GNNs and graph kernels.

2.1 Notations and Definitions

Let $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{X})$ be an undirected and unweighted graph with N nodes, where $\mathcal{V}$ denotes the node set, $\mathcal{E}$ denotes the edge set, and $\mathbf{X}$ denotes the feature matrix, where each row represents the features of a corresponding node. The neighborhood of a node v is defined as the set of all nodes that connected to v: $\mathcal{N}(v) = \{u|(v, u) \in \mathcal{E}\}$. In a graph dataset, each graph $\mathcal{G}_i$ is associated with a label $\mathcal{Y}_i$, which is sampled from a label set $\mathcal{L}$. In this paper, we focus on the graph classification task, where a model ϕ is trained to map each graph to its label.

2.2 Weisfeiler-Lehman Isomorphism Test

We first introduce the Weisfeiler-Lehman isomorphism test [Weisfeiler and Lehman, 1968], which can be used to distinguish different graphs and is closely related to the message-passing process of GNNs [Xu et al., 2019]. The WL algorithm operates by iteratively relabeling the colors of vertices in the graph. Initially, all vertices are assigned the same color C_0. In iteration i, the color of a vertex v is updated based on its current color $C_{v,i-1}$ and the colors of its neighbors $\left\{C_{u\in\mathcal{N}(v),i-1}\right\}$. The update is given as follows:

$$C_{v,i} = f_c^i\left(\left\{C_{v,i-1}, \left\{C_{u\in\mathcal{N}(v),i-1}\right\}\right\}\right)$$

where f_c^i is an injective coloring function that maps the multisets to different colors at iteration i.This process continues for a predefined number of iterations or until the coloring stabilizes (i.e., the colors no longer change).

2.3 Graph Neural Network

Most GNNs adopt the message-passing framework [Gilmer et al., 2017], which can be viewed as a derivative of the Weisfeiler-Lehman coloring mechanism. Specifically, let $\boldsymbol{h}_v^{(k-1)}$ represent the feature vector of node v at the $(k-1)$-th iteration. A GNN computes the new feature for v by aggregating the representations of itself and its neighboring nodes $u \in \mathcal{N}(v)$ as follows:

$$\boldsymbol{h}_v^{(k)} = \texttt{UPDATE}^{(k)}\left(\boldsymbol{h}_v^{(k-1)}, \boldsymbol{m}_v^{(k)}\right), \text{ where } \boldsymbol{m}_v^{(k)} = \texttt{AGGR}^{(k)}\left(\left\{\boldsymbol{h}_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$$

The initial node representations $\boldsymbol{h}_v^{(0)}$ are set to the raw node features $\mathbf{X}_v$. At the k-th iteration, the aggregation function $\texttt{AGGR}^{(k)}(\cdot)$ computes the messages $\boldsymbol{m}_v^{(k)}$ received from neighboring nodes. Subsequently, the update function $\texttt{UPDATE}^{(k)}(\cdot)$ computes a new representation for each node by integrating the neighborhood messages $\boldsymbol{m}_v^{(k)}$ with its previous embedding $\boldsymbol{h}_v^{(k-1)}$. After T iterations, the final node representations are combined into a graph representation using a readout function:

$$\boldsymbol{h}_G = \texttt{READOUT}\left(\left\{\boldsymbol{h}_v^{(T)} \mid v \in \mathcal{V}\right\}\right).$$

The readout function, essentially a set function learned by the neural network, commonly employs `AVERAGE` or `MAXPOOL`.

2.4 Graph Kernel

A kernel is a function used to measure the similarity between pairs of objects. For a non-empty set χ and a function $\mathbb{K} : \chi \times \chi \to \mathbb{R}$, the function $\mathbb{K}$ qualifies as a kernel on χ if there exists a Hilbert space $\mathcal{H}_k$ and a feature map function $\phi : \chi \to \mathcal{H}_k$, such that $\mathbb{K}(x, y) = \langle\phi(x), \phi(y)\rangle$ for any $x, y \in \chi$, where $\langle\cdot, \cdot\rangle$ denotes the inner product in $\mathcal{H}_k$. Notably, such a feature map exists if and only if $\mathbb{K}$ is a positive semi-definite function. Let $\mathbb{K}$ be a kernel defined on χ, and let $S = \{x_1, \ldots, x_n\}$ be a finite set of n samples on χ. The Gram matrix for S is defined as $\mathbf{G} \in \mathbb{R}^{n\times n}$, with each element $\mathbf{G}_{ij} = \mathbb{K}(x_i, x_j)$ representing the kernel value between the i-th and j-th data points in S. The Gram matrix is always positive semi-definite.

Graph kernel methods apply the kernel approaches to graph data, typically defined using the $\mathcal{R}$-convolution framework [Haussler et al., 1999, Bause and Kriege, 2022]. Consider two graphs, $\mathcal{G}$ and $\mathcal{G}'$. The key idea is to decompose the graphs into substructures using a predefined feature map function ϕ, and then compute the kernel value by taking the inner product in the feature space: $\mathbb{K}(\mathcal{G}, \mathcal{G}') = \langle\phi(\mathcal{G}), \phi(\mathcal{G}')\rangle$, based on these substructures. Weisfeiler-Lehman (WL) graph kernels stand out as one of the most widely used approaches. These methods employ the Weisfeiler-Lehman coloring scheme to iteratively encode graphs, calculating kernel values based on these colorings. The final kernel values are derived through the aggregation of intermediate results. Next, we introduce the WL-subtree kernel and the WLOA kernel in more detail. Let f^i be the coloring function at the i-th iteration, mapping the colored graph from the previous iteration to a new colored graph. Define ψ^i as the function that captures the cumulative coloring effect up to the i-th iteration: $\psi^i = f^i \circ \cdots \circ f^1$. Specifically, the WL-subtree kernel computes the kernel value by directly using the label histogram as a feature to compute the dot product, which is expressed as:

$$\mathbb{K}^{(h)}_{wl\_subtree}(\mathcal{G}, \mathcal{G}') = \sum_{i=1}^{h} \left\langle \phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G}')) \right\rangle$$

The WLOA kernel applies a histogram intersection kernel to match substructures between different graphs, which is formulated as:

$$\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}') = \sum_{i=1}^{h} \texttt{histmin}\left\{\phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G}'))\right\} \cdot \omega(i)$$

where $\omega(i)$ is a nonnegative, monotonically non-decreasing weight function, and $\omega(i) = 1$ is commonly used in practice [Kriege et al., 2016, Siglidis et al., 2020]. The operator `histmin` denotes the histogram intersection kernel. It is computed by comparing and summing the smallest matching elements between two sets. For example, consider two sets $S_1 : \{\mathfrak{a}, \mathfrak{a}, \mathfrak{b}, \mathfrak{b}, \mathfrak{c}\}$ and $S_2 : \{\mathfrak{a}, \mathfrak{b}, \mathfrak{b}, \mathfrak{c}, \mathfrak{c}\}$. The $\texttt{histmin}(S_1, S_2)$ is calculated by taking the minimum frequency of each distinct element across the two sets, yielding $\min(2, 1) + \min(2, 2) + \min(1, 2) = 4$. To ensure that the kernel values fall in the range of 0 to 1, normalization is often applied. The normalized kernel value $\tilde{\mathbb{K}}^{(h)}$ is then expressed as follows:

$$\tilde{\mathbb{K}}^{(h)}(\mathcal{G}, \mathcal{G}') = \frac{\mathbb{K}^{(h)}(\mathcal{G}, \mathcal{G}')}{\sqrt{\mathbb{K}^{(h)}(\mathcal{G}, \mathcal{G})}\sqrt{\mathbb{K}^{(h)}(\mathcal{G}', \mathcal{G}')}}$$

3 Consistency Principles

To encode relational structures in GNN learning, we first examine how similarities are represented in graph kernels. In this section, we start by defining a class of graph kernels, i.e., the iterative graph kernels, which encompasses many widely used kernels. Then, we delve into a key property known as the consistency property, which may play an important role in enhancing classification performance. We support this assertion through theoretical analysis, elucidating how different kernels adhere to or deviate from this property, thereby explaining their performance differences.

3.1 Iterative Graph Kernels

In this paper, we are interested in a set of kernels defined as follows:

Definition 3.1 *Given a colored graph set χ, a feature map function $\phi : \chi \to \mathcal{H}_k$ (where $\mathcal{H}_k$ is a Hilbert space), and a set of coloring functions $\mathcal{F}_c = \{f^0, f^1, \ldots, f^i\}$ on χ (with $f^i : \chi \to \chi$), we define the set of* ***iterative graph kernels (IGK)*** *as:*

$$\mathbb{K}_{\mathcal{F}_c,\phi}(x, y, i) = \langle \phi(f^i \circ \cdots f^1(x)), \phi(f^i \circ \cdots f^1(y)) \rangle = \langle \psi^i(x), \psi^i(y) \rangle$$

where $x, y \in \chi$ and $\psi^i(\cdot)$ represents a composite function given by: $\psi^i = f^i \circ \cdots \circ f^1$, Then the normalized kernel is given by:

$$\tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x, y, i) = \frac{\mathbb{K}_{\mathcal{F}_c,\phi}(x, y, i)}{\sqrt{\mathbb{K}_{\mathcal{F}_c,\phi}(x, x, i)}\sqrt{\mathbb{K}_{\mathcal{F}_c,\phi}(y, y, i)}}$$

Based on this definition, we can see that graph kernels utilizing the Weisfeiler-Lehman framework, including the WL-subtree kernel [Shervashidze et al., 2011], WLOA [Kriege et al., 2016], and the Wasserstein Weisfeiler-Lehman kernel [Togninalli et al., 2019], should be classified as iterative graph kernels. Conversely, the subgraph-pattern approaches, such as the graphlet kernel [Shervashidze et al., 2009] and the shortest-path kernel [Borgwardt and Kriegel, 2005], do not fall into this category.

3.2 Key Properties: Monotonic Decrease & Order Consistency

To effectively capture relational structures, we design the IGKs to progressively differentiate between graphs. With each iteration, additional structural features are considered, enabling the distinction of graphs that may have been indistinguishable in earlier iterations. This implies two properties: (1) the kernel values monotonically decrease with larger iterations, as the similarity between two graphs decreases with the consideration of more features; and (2) the similarity rankings across different iterations should remain consistent, meaning that graphs deemed dissimilar in early iterations should not be considered similar in later iterations. We then provide formal definitions for these two properties and prove how they can improve generalization in the binary classification task.

Definition 3.2 (Monotonic Decrease) *The normalized iterative graph kernels $\tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,y,i)$ are said to be monotonically decreasing if and only if:*

$$\tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,y,i) \geq \tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,y,i+1) \quad \forall x,y \in \chi$$

Definition 3.3 (Order Consistency) *The normalized iterative graph kernels $\tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,y,i)$ are said to preserve order consistency if the similarity ranking remains consistent across different iterations for any pair of graphs, which is defined as:*

$$\tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,y,i) > \tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,z,i) \Rightarrow \tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,y,i+1) \geq \tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,z,i+1) \quad \forall x,y,z \in \chi$$

Next, we show that these two properties can lead to a monotonically decreasing upper bound on the test error in the binary classification task, which suggests improved performance.

Consider a binary graph classification task and assume that the graph representations obtained at any iteration have a uniform norm. This can be achieved by simply normalizing the graph representations at the end of each iteration. That is, for any graph x: $\|\phi(\psi^i(x))\| = 1$. Then, for any IGK that is monotonically decreasing and preserves the order consistency, the following theorem holds:

Theorem 3.4 *Let $\tilde{\mathbb{K}}_{\mathcal{F}_c,\phi}(x,y,i)$ be a normalized iterative graph kernel. Its generalization ability on the test set is provably strong when it decreases monotonically and preserves order consistency.*

Proof sketch. We consider a binary classification task where supervision is provided through labels and similarity orders. We demonstrate that the presence of these two properties enhances the generalizability of the learned iterative kernel. The proof is structured in two steps:

(1) We first show that iterative kernels can learn the correct ranking given sufficient data during training, using the PAC-learning framework.

(2) Next, we demonstrate that, once the correct ranking is learned, an iterative kernel learned with the properties of monotonic decrease and order consistency guarantees that the error rate of the graphs in the test set decreases as the number of iterations increases.

Details of the proof can be found in Appendix A.

3.3 Theoretical Verification with WL-based Kernels

As discussed in Section 3.1, WL-based kernels can be categorized as iterative graph kernels, as they are generated by the coloring functions in an iterative refinement process. Consequently, a natural question arises regarding how various kernels adhere to these properties and whether their adherence reflects their actual performance. We thus investigate two popular WL-based kernels: the WL-subtree kernel [Shervashidze et al., 2011] and the WLOA kernel [Kriege et al., 2016].

Theorem 3.5 *The normalized WL-subtree kernel is neither monotonically decreasing nor does it preserve order consistency.*

Theorem 3.6 *The normalized WLOA kernel is monotonically decreasing and asymptotically preserves order consistency when $\omega(i) = 1$.*

Proof sketch. For Theorem 3.5, we illustrate a counterexample, while for Theorem 3.6, we consider two graph pairs where the similarity condition holds:

$$\tilde{\mathbb{K}}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}') \geq \tilde{\mathbb{K}}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}'')$$

The similarity at the next iteration $h+1$ is scaled by a factor dependent $\omega(i), i = 1, \cdots, h+1$. The unnormalized kernel increases monotonically, though with diminishing increments over iterations. Given this, when $\omega(i) = 1$ and $h \to \infty$, we obtain:

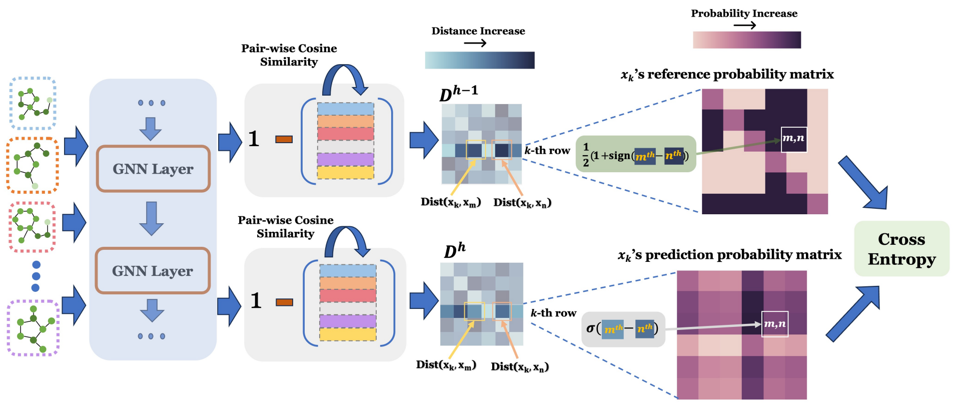
$$\tilde{\mathbb{K}}^{(h+1)}_{WLOA}(\mathcal{G}, \mathcal{G}') \geq \tilde{\mathbb{K}}^{(h+1)}_{WLOA}(\mathcal{G}, \mathcal{G}'')$$

We include the complete proof in Appendix B.1 and B.2.

These findings imply that the WLOA kernel better preserves relational structures compared to the WL-subtree kernel, leading to improved classification performance, as supported by the literature [Kriege et al., 2016].

4 Proposed Strategy

Given the analogy between WL-based kernels and GNNs [Shervashidze et al., 2011, Gilmer et al., 2017], and the observation that GNNs often fail to preserve relational structures, we hypothesize that the consistency principle is also beneficial to GNN models. Thus, we aim to explore how to effectively preserve this principle within the GNN architectures.



Probability Increase
Distance Increase
Pair-wise Cosine Similarity
x_k's reference probability matrix
D^{h-1}
GNN Layer
1 −
k-th row
$\frac{1}{2}(1+\text{sign}(m^{th} - n^{th}))$
m,n
Dist(x_k, x_m)
Dist(x_k, x_n)
Pair-wise Cosine Similarity
D^h
x_k's prediction probability matrix
Cross Entropy
GNN Layer
1 −
k-th row
$\sigma(m^{th} - n^{th})$
m,n
Dist(x_k, x_m)
Dist(x_k, x_n)

Figure 2: Computation of Consistency loss. At each layer, pairwise distance matrix $\mathbf{D}$ is calculated using the normalized representations of graphs in a batch. After randomly selecting a reference graph x_k, the reference probability matrix is computed using the distance matrix from previous layer, where entry (n, m) represents the known probability that the graph x_k is more similar to the graph x_n than to the graph x_m. For the distance matrix of current layer, we compute the predicted probability that x_k is closer to x_n than to x_m and form the prediction probability matrix. Consistency loss is computed as the cross-entropy between the predicted and reference probability matrices

4.1 Consistency Loss

Our objective is to enhance graph representation consistency across GNN layers, which has significant potential to preserve the relational structure in the representation space. If we compare $\phi(\psi^i(G))$ to the graph representations obtained at the i-th layer, preserving the consistency principle is equivalent to preserving the ordering of cosine similarity among the graph representations. However, due to the non-differentiable nature of ranking operations, directly minimizing the ranking differences between consecutive layers is not feasible using gradient-based optimization techniques. Therefore, we aim to optimize pairwise ordering relations instead of the entire ranking list. In this work, our proposed loss employs a probabilistic approach inspired by [Burges et al., 2005]. The entire framework is illustrated in Figure 2.

Let $\mathbf{H}^h \in \mathbb{R}^{n \times d}$ denote the graph embedding matrix for n examples in a batch, each with a d-dimensional feature vector. We first compute the distance matrix $\mathbf{D}^h$ for all the graphs in a batch at the h-th layer. The entries $D_{i,j}^h$ of this matrix represent the distance between the representations of the i-th and j-th graphs, calculated as $D_{i,j}^h = \texttt{Dist}\left(H_{x_i}^h, H_{x_j}^h\right)$. Here, we use the cosine distance, the complement of cosine similarity in positive space, expressed as: $1 - \frac{H_{x_i}^h \cdot H_{x_j}^h}{\left\|H_{x_i}^h\right\| \cdot \left\|H_{x_j}^h\right\|}$. Considering the distance relationship to an arbitrary graph x_k in the batch, the predicted probability $\hat{\mathbb{P}}_{n,m|k}^h$ that x_k is more similar to graph x_n than to graph x_m at layer h is defined as $\hat{\mathbb{P}}_{n,m|k}^h\left(\hat{D}_{k,n}^h < \hat{D}_{k,m}^h\right)$. This probability score, which ranges from 0 to 1, is formulated using the sigmoid function as follows:

$$\hat{\mathbb{P}}_{n,m|k}^h\left(\hat{D}_{k,n}^h < \hat{D}_{k,m}^h\right) = \frac{1}{1 + \exp\left(\hat{D}_{k,m}^h - \hat{D}_{k,n}^h\right)}$$

Given the distance matrix from the previous layer D^{h-1}, the known probability that graph x_k is more similar to graph x_n than to graph x_m, denoted as $\tilde{\mathbb{P}}_{n,m}^{h-1}(\hat{D}_{k,n}^{h-1}, \hat{D}_{k,m}^{h-1})$, can be formulated as follows:

$$\tilde{\mathbb{P}}_{n,m|k}^{h-1} = \frac{1}{2}\left(1 + \text{sign}(\hat{D}_{k,n}^{h-1} - \hat{D}_{k,m}^{h-1})\right)$$

We can then minimize the discrepancy between the predicted and the known probability distributions to enhance representation consistency across the layers. Here,we employ the cross-entropy loss to effectively measure the divergence between these two distributions. Specifically, for a pair (x_n, x_m) centered on x_k at layer h, the cross-entropy loss can be expressed as:

$$\mathcal{L}_{\text{cross-entropy}}\left((x_n, x_m) \mid x_k, h\right) = -\tilde{\mathbb{P}}_{n,m|k}^{h-1} \log \hat{\mathbb{P}}_{n,m|k}^h - \left(1 - \tilde{\mathbb{P}}_{n,m|k}^{h-1}\right) \log\left(1 - \hat{\mathbb{P}}_{n,m|k}^h\right)$$

Then, the total loss function, which quantifies the consistency of pair-wise distance relations for graph x_k at layer h, can be formulated as

$$\mathcal{L}_{\text{consistency}}(k) = \sum_{n,m} \mathcal{L}((x_n, x_m) \mid x_k, h)$$

The overall objective function of the proposed framework can then be formulated as the weighted sum of the original loss and the consistency loss.

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{origin}} + \lambda \sum_i \mathcal{L}_{\text{consistency}}(i)$$

Here, λ is a hyperparameter that controls the strength of the consistency constraint.

5 Experiment

In this section, we examine the effectiveness of the proposed consistency loss for the graph classification task. Specifically, we aim to address the following questions: **Q1:** Does the consistency loss effectively enhance the performance of various GNN backbones in the graph classification task? **Q2:** How does the consistency loss influence the rank correlation of graph similarities across GNN layers? **Q3:** How does the consistency loss influence dataset performance across varying levels of complexity, both in structural intricacy and task difficulty?

5.1 Experiment Setup

Dataset We conduct extensive experiments using the TU Dataset [Morris et al., 2020], the Open Graph Benchmark (OGB) [Hu et al., 2020] and Reddit Threads(Reddit-T) dataset [Bause and Kriege, 2022]. The TU Dataset consists of eight graph classification datasets, categorized into three main types: (1) Chem/Bioinformatics datasets, including D&D, NCI1, NCI109, and PROTEINS; (2) Social Network datasets, including IMDB-BINARY,IMDB-MULTI and COLLAB, where a constant feature value of 1 was assigned to all vertices due to the absence of vertex features; and (3) a Computer Vision dataset, COIL-RAG. The OGB datasets include ogbg-molhiv, a molecular property prediction dataset used to determine whether a molecule inhibits HIV replication. The Reddit-T dataset is used for classifying threads from Reddit as either discussions or non-discussions, where users are represented as nodes and replies between them as links.

For the TU dataset and Reddit-T, consistent with prior work [Xinyi and Chen, 2019, Xu et al., 2019], we utilize an 8:1:1 ratio for training, validation, and testing sets. For the OGB datasets, we use the official splits provided. Training stops at the highest validation performance, and test accuracy is taken from the corresponding epoch in each fold. Final results are reported as the mean accuracy (except ogbg-molhiv) and standard deviation over 10 folds. For ogbg-molhiv, we follow the official evaluator and use ROC-AUC as the evaluation metric. Detailed information about these datasets can be found in Appendix C.

Model We use three widely adopted GNN models as baselines: namely, GCN [Kipf and Welling, 2017], GIN [Xu et al., 2019], and GraphSAGE [Hamilton et al., 2017]. We also include two recent GNN models, namely GTransformer [Shi et al., 2021] and GMT [Baek et al., 2021]. To ensure a fair comparison, we maintain the same number of layers and layer sizes for both the base models and the models with our proposed consistency loss, ensuring the sharing of the same network architecture. Detailed information about hyperparameter tuning is provided in Appendix D.

5.2 Effectiveness of Consistency Loss

To answer **Q1**, we present the results for the TU, OGB, and Reddit-T datasets in Table 1. As shown in this table, GNN models with the consistency loss yield significant performance improvements over their base models on different datasets. These findings suggest that the consistency framework is a versatile and robust approach for enhancing the predictive capabilities of GNNs in real-world datasets, irrespective of the base model and dataset domain. Notably, the GIN method demonstrates the most significant improvements, achieving enhancements of up to 4.51% on the D&D dataset, 4.32% on the COLLAB dataset, and 3.70% on the IMDB-B dataset. This improvement can be linked to our empirical observation regarding the weak ability of GIN to preserve consistency across layers. In addition, our method demonstrates satisfactory improvements on datasets with numerous classes (e.g., COIL-RAG) and large-scale datasets (e.g., ogbg-molhiv and Reddit-T), indicating that our approach is both flexible and scalable for handling complex and extensive datasets.

Table 1: Classification performance on the TU, OGB and Reddit-T datasets, with and without the consistency loss. Highlighted cells indicate instances where the base GNN with the consistency loss outperforms the base GNN alone. The reported values are average accuracy for TU datasets and ROC-AUC for the ogbg-molhiv dataset, including their standard deviations.

	NCI1	NCI109	PROTEINS	D&D	IMDB-B	IMDB-M	COLLAB	COIL-RAG	OGB-HIV	REDDIT-T
#Graphs	4110	4127	1113	1178	1000	1500	5000	3900	41127	203088
Avg. #nodes	29.87	29.68	39.06	284.32	19.77	13.00	74.49	3.01	25.50	23.93
GCN	73.96$\pm$ 2.37	74.04$\pm$ 3.09	73.24$\pm$ 6.93	74.92$\pm$ 2.66	75.40$\pm$ 2.97	55.07$\pm$1.24	81.72$\pm$0.84	91.72$\pm$1.65	72.86$\pm$1.90	76.00$\pm$0.44
$+\mathcal{L}_{\text{consistency}}$	75.12$\pm$ 1.19	73.25$\pm$ 1.25	75.07$\pm$ 5.05	78.56$\pm$ 3.32	75.85$\pm$ 1.82	56.27$\pm$1.00	83.44$\pm$0.45	93.38$\pm$1.64	73.75$\pm$0.89	77.12$\pm$0.12
GIN	78.13$\pm$ 2.11	76.75$\pm$ 2.91	72.97$\pm$ 4.59	71.10$\pm$ 4.63	70.80$\pm$ 4.07	52.13$\pm$1.42	79.84$\pm$1.05	93.33$\pm$1.48	71.60 $\pm$2.36	77.50$\pm$0.16
$+\mathcal{L}_{\text{consistency}}$	79.45$\pm$ 1.09	77.46$\pm$ 1.96	74.98$\pm$ 4.57	75.51$\pm$ 2.63	74.50$\pm$ 3.06	53.46$\pm$2.44	84.16$\pm$0.81	94.03$\pm$1.33	74.57$\pm$1.61	77.64$\pm$0.05
GraphSAGE	74.40$\pm$ 1.83	73.17$\pm$ 0.47	74.96$\pm$ 3.14	76.44$\pm$ 4.16	73.90$\pm$ 2.17	51.33$\pm$2.95	78.92$\pm$1.20	89.56$\pm$2.37	77.03$\pm$1.65	76.67$\pm$0.11
$+\mathcal{L}_{\text{consistency}}$	78.26$\pm$ 1.08	74.10$\pm$ 2.10	76.40$\pm$ 3.12	77.50$\pm$ 3.38	74.75$\pm$ 3.06	54.27$\pm$1.24	82.12$\pm$0.78	92.31$\pm$1.32	78.60$\pm$1.44	77.57$\pm$0.05
GTransformer	75.72$\pm$2.69	74.79$\pm$1.82	73.33$\pm$4.80	75.42$\pm$3.22	72.20$\pm$3.49	53.33$\pm$1.12	80.36$\pm$0.56	83.74$\pm$3.17	76.81$\pm$1.34	76.75$\pm$0.12
$+\mathcal{L}_{\text{consistency}}$	76.83$\pm$1.36	75.82$\pm$1.53	77.03$\pm$3.79	76.57$\pm$2.54	73.75$\pm$2.56	56.53$\pm$1.54	80.48$\pm$0.47	91.67$\pm$1.88	76.90$\pm$3.25	77.14$\pm$0.06
GMT	75.04$\pm$1.43	73.90$\pm$2.29	72.70$\pm$4.21	72.80$\pm$2.19	79.80$\pm$1.08	54.13$\pm$2.90	80.36$\pm$1.15	90.85$\pm$1.91	74.86$\pm$2.26	72.06$\pm$10.15
$+\mathcal{L}_{\text{consistency}}$	75.52$\pm$1.07	75.20$\pm$0.95	74.86$\pm$2.03	73.14$\pm$2.28	79.60$\pm$1.91	54.80$\pm$1.42	82.80$\pm$0.61	92.00$\pm$1.43	76.00$\pm$1.99	77.19$\pm$0.14

Furthermore, we analyze the complexity and scalability of our method on the TU Dataset, with additional details provided in Appendices E.1 and E.2. Additionally, we demonstrate the method's potential to enhance performance significantly, even with a marginal increase in computational cost, as illustrated in Appendix F.

5.3 Effect of the Consistency Loss on Rank Correlation

To answer **Q2**, we compare the consistency of graph representations across layers with and without the proposed consistency loss. Specifically, we use the Spearman's rank correlation coefficient, a widely accepted method for computing correlation between ranked variables, to quantitatively measure the consistency of graph similarities across layers. For a fair comparison, we construct a distance matrix D^h for all test data at each layer h, where each row $D^h_{x_i;}$ represents the distances from graph x_i to all other graphs. We then compute the rank correlation between $D^h_{x_i,:}$ and $D^{h+1}_{x_i;}$ for each graph x_i.

We average the correlation values for all graphs to obtain the overall correlation for layer h. Then, we compute the mean of these values across layers, enabling a global comparison of relational consistency throughout the model and dataset. All results were averaged over 5 repeated experiments with the same training setting.

Table 2: Spearman correlation was computed for graph representations from consecutive layers on the TU datasets, both with and without consistency loss. Values with higher rank correlation are highlighted. The consistency loss can enhance the rank correlation of graph similarities.

	NCI1	NCI109	PROTEINS	D&D	IMDB-B
GCN	0.753	0.920	0.584	0.709	0.846
$+\mathcal{L}_{\text{consistency}}$	0.859	0.958	0.946	0.896	0.907
GIN	0.666	0.674	0.741	0.721	0.598
$+\mathcal{L}_{\text{consistency}}$	0.877	0.821	0.904	0.847	0.816
GraphSAGE	0.903	0.504	0.845	0.741	0.806
$+\mathcal{L}_{\text{consistency}}$	0.911	0.709	0.916	0.872	0.933
GTransformer	0.829	0.817	0.867	0.865	0.884
$+\mathcal{L}_{\text{consistency}}$	0.863	0.883	0.915	0.880	0.917
GMT	0.872	0.887	0.980	0.826	0.893
$+\mathcal{L}_{\text{consistency}}$	0.906	0.908	0.983	0.856	0.908

We present our results on a series of datasets from the TU Dataset in Table 2. As shown in the table, it is evident that the representation space becomes more consistent with our proposed consistency loss. For example, a significant enhancement is observed for the GIN model. Another notable point is the result for the GCN model on the NCI109 dataset and for the GMT model on the IMDB-B dataset. We find that the correlation is already fairly high even without the implementation of $\mathcal{L}_{\text{consistency}}$, resulting in minimal correlation improvements with our method. This phenomenon provides a plausible explanation for why our method is not effective in these two cases.

5.4 Study on Task Complexity

To address **Q3** and further evaluate the performance of our method across different scenarios, we extended our study by conducting experiments on graph datasets with increasing task and structural complexity.

Increasing Task Complexity We increase the task complexity by expanding the number of classes that the model needs to classify. To assess the effect of increased class complexity on our method's performance, we sampled subsets from the REDDIT-MULTI 5K dataset [Yanardag and Vishwanathan, 2015] with a progressively greater number of classes, which originally consists of five classes. Specifically, we randomly sampled between 2 and 4 classes to construct new datasets from the original dataset and conducted classification tasks using both GCN and GCN with $\mathcal{L}_{\text{consistency}}$ on these newly constructed datasets. We report the mean test accuracy over five experiments for each subset, as presented in Table 3.

Table 3: Performance comparison across different subsets and the full set.

	Subset1 (2 classes)	**Subset2 (3 classes)**	**Subset3 (4 classes)**	**Fullset (5 classes)**
GCN	79.50	67.13	50.30	53.80
GCN+$\mathcal{L}_{\text{consistency}}$	81.10	68.00	57.15	57.12

The results demonstrate that the effectiveness of our method remains robust, even as the number of classes increases. In fact, it may provide greater advantages when applied to multi-class classification tasks. This resilience likely stems from our method's focus on identifying relational structures in the intermediate representations of GNN models, rather than relying heavily on label information. This approach helps mitigate the impact of potential label noise in the original data. These findings align with the noticeable performance improvements observed in both binary and multi-class classification tasks, as shown in Table 1.

Increasing Structural Complexity We assessed the impact of structural complexity by partitioning the IMDB-B dataset into three subsets with progressively increasing graph densities. Graph density, denoted as $d = \frac{2M}{N(N-1)}$, where N is the number of nodes and M is the number of edges in graph

$\mathcal{G}$, was used as the criterion for creating these subsets. The dataset was divided into three groups: (small) for graphs with densities below the 33rd percentile, (median) for densities between the 33rd and 67th percentiles, and (large) for graphs with densities above the 67th percentile. We applied both GCN and GCN+$\mathcal{L}_{\text{consistency}}$ models to these subsets, and the results are summarized in Table 4.

Table 4: Performance comparison on IMDB-B datasets of different densities.

	IMDB-B **(small)**	IMDB-B **(medium)**	IMDB-B **(large)**
GCN	$77.58_{\pm 4.11}$	$66.25_{\pm 5.38}$	$67.61_{\pm 6.21}$
GCN+$\mathcal{L}_{\text{consistency}}$	$84.24_{\pm 4.85}$	$69.06_{\pm 4.06}$	$71.43_{\pm 4.43}$

The above results show that the GCN model, enhanced with the $\mathcal{L}_{\text{consistency}}$ loss function, consistently outperforms the original version across different structural complexity groups, demonstrating the robustness and effectiveness of the proposed method.

Furthermore, we evaluate the method's efficiency under various complexity scenarios, as detailed in the Appendix E.3.

6 Related Work

Graph Distance and Similarity Measuring distances or similarities between graphs is a fundamental problem in graph learning. Graph kernels, which define graph similarity, have gained significant attention. Most graph kernels use the $\mathcal{R}$-Convolution framework [Haussler et al., 1999] to compare substructure similarities. A trailblazing kernel by [Kashima et al., 2003] used node and edge attributes to generate label sequences through a random walk. The WL-subtree kernel [Shervashidze et al., 2011] generates graph-level features by summing node representation contributions. Recent works align matching substructures between graphs. For instance, Kriege et al. [2016] proposed a discrete optimal assignment kernel based on vertex kernels from WL labels. Togninalli et al. [2019] extended this to include fractional assignments using the Wasserstein distance. Additionally, measuring graph distances is also a prevalent problem. Vayer et al. [2019] combined the Wasserstein and Gromov-Wasserstein distances [Villani and Society, 2003, Mémoli, 2011]. Chen et al. [2022] proposed a polynomial-time WL distance for labeled Markov chains, treating labeled graphs as a special case.

Bridging Graph Kernels and GNNs Many studies have explored the connection between graph kernels and GNNs, attempting to integrate them into a unified framework. Certain approaches focus on leveraging GNN architecture to design novel kernels. For instance, Mairal et al. [2014] presents neural network architectures that learn graph representations within the Reproducing Kernel Hilbert Space (RKHS) of graph kernels. Similarly, Du et al. [2019] proposed a graph kernel equivalent to infinitely wide GNNs, which can be trained using gradient descent. Conversely, other studies have incorporated kernel methods directly into GNNs. For example, Nikolentzos and Vazirgiannis [2020] utilized graph kernels as convolutional filters within GNN architectures. Additionally, Lee et al. [2024] proposes a novel Kernel Convolution Network that employs the random walk kernel as the core mechanism for learning descriptive graph features. Instead of applying specific kernel patterns as mentioned in previous work, we introduce a general method for GNNs to capture consistent similarity relationships, thereby enhancing classification performance.

7 Conclusion

In this paper, we study a class of graph kernels and introduce the concept of the consistency property for graph classification tasks. We theoretically prove that this property leads to a more structure-aware representation space for classification using kernel methods. Based on this analysis, we extend this principle to enhance GNN models. We propose a novel, model-agnostic consistency learning framework for GNNs that enables them to capture relational structures in the graph representation space. Experiments show that our proposed method universally enhances the performance of backbone networks on graph classification benchmarks, providing new insights into bridging the gap between traditional kernel methods and GNN models.

References

Jinheon Baek, Minki Kang, and Sung Ju Hwang. Accurate learning of graph representations with graph multiset pooling. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=JHcqXGaqiGn.

Franka Bause and Nils Morten Kriege. Gradual weisfeiler-leman: Slow and steady wins the race. In Bastian Rieck and Razvan Pascanu, editors, *Learning on Graphs Conference, LoG 2022, 9-12 December 2022, Virtual Event*, volume 198 of *Proceedings of Machine Learning Research*, page 20. PMLR, 2022. URL https://proceedings.mlr.press/v198/bause22a.html.

Karsten M. Borgwardt and Hans-Peter Kriegel. Shortest-path kernels on graphs. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005), 27-30 November 2005, Houston, Texas, USA*, pages 74–81. IEEE Computer Society, 2005. doi: 10.1109/ICDM.2005.132. URL https://doi.org/10.1109/ICDM.2005.132.

Christopher J. C. Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Gregory N. Hullender. Learning to rank using gradient descent. In Luc De Raedt and Stefan Wrobel, editors, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 89–96. ACM, 2005. doi: 10.1145/1102351.1102363. URL https://doi.org/10.1145/1102351.1102363.

Samantha Chen, Sunhyuk Lim, Facundo Mémoli, Zhengchao Wan, and Yusu Wang. Weisfeiler-lehman meets gromov-wasserstein. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 3371–3416. PMLR, 2022. URL https://proceedings.mlr.press/v162/chen22o.html.

Simon S. Du, Kangcheng Hou, Ruslan Salakhutdinov, Barnabás Póczos, Ruosong Wang, and Keyulu Xu. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5724–5734, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/663fd3c5144fd10bd5ca6611a9a5b92d-Abstract.html.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR, 2017. URL http://proceedings.mlr.press/v70/gilmer17a.html.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1024–1034, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html.

David Haussler et al. Convolution kernels on discrete structures. Technical report, Citeseer, 1999.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/fb60d411a5c5b72b2e7d3527cfc84fd0-Abstract.html.

Zheng Huang, Qihui Yang, Dawei Zhou, and Yujun Yan. Enhancing size generalization in graph neural networks through disentangled representation learning. In *Forty-first International Conference on Machine Learning*, 2024.

Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Marginalized kernels between labeled graphs. In Tom Fawcett and Nina Mishra, editors, *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pages 321–328. AAAI Press, 2003. URL `http://www.aaai.org/Library/ICML/2003/icml03-044.php`.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL `https://openreview.net/forum?id=SJU4ayYgl`.

Nils M. Kriege, Pierre-Louis Giscard, and Richard C. Wilson. On valid optimal assignment kernels and applications to graph classification. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1615–1623, 2016. URL `https://proceedings.neurips.cc/paper/2016/hash/0efe32849d230d7f53049ddc4a4b0c60-Abstract.html`.

Nils M. Kriege, Fredrik D. Johansson, and Christopher Morris. A survey on graph kernels. *Appl. Netw. Sci.*, 5(1):6, 2020. doi: 10.1007/S41109-019-0195-3. URL `https://doi.org/10.1007/s41109-019-0195-3`.

Meng-Chieh Lee, Lingxiao Zhao, and Leman Akoglu. Descriptive kernel convolution network with improved random walk kernel. In Tat-Seng Chua, Chong-Wah Ngo, Ravi Kumar, Hady W. Lauw, and Roy Ka-Wei Lee, editors, *Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, May 13-17, 2024*, pages 457–468. ACM, 2024. doi: 10.1145/3589334.3645405. URL `https://doi.org/10.1145/3589334.3645405`.

Gaotang Li, Marlena Duda, Xiang Zhang, Danai Koutra, and Yujun Yan. Interpretable sparsification of brain graphs: Better practices and effective designs for graph neural networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1223–1234, 2023a.

Gaotang Li, Danai Koutra, and Yujun Yan. Size generalization of graph neural networks on biological data: Insights and practices from the spectral perspective. *arXiv preprint arXiv:2305.15611*, 2023b.

Lu Lin, Jinghui Chen, and Hongning Wang. Spectral augmentation for self-supervised learning on graphs. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL `https://openreview.net/forum?id=DjzBCrMBJ_p`.

Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. Pre-training molecular graph representation with 3d geometry. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL `https://openreview.net/forum?id=xQUe1pOKPam`.

Julien Mairal, Piotr Koniusz, Zaïd Harchaoui, and Cordelia Schmid. Convolutional kernel networks. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2627–2635, 2014. URL `https://proceedings.neurips.cc/paper/2014/hash/81ca0262c82e712e50c580c032d99b60-Abstract.html`.

Facundo Mémoli. Gromov-wasserstein distances and the metric approach to object matching. *Found. Comput. Math.*, 11(4):417–487, 2011. doi: 10.1007/S10208-011-9093-5. URL `https://doi.org/10.1007/s10208-011-9093-5`.

Mehryar Mohri. Foundations of machine learning, 2018.

Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *CoRR*, abs/2007.08663, 2020. URL https://arxiv.org/abs/2007.08663.

Giannis Nikolentzos and Michalis Vazirgiannis. Random walk graph neural networks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/ba95d78a7c942571185308775a97a3a0-Abstract.html.

Giannis Nikolentzos, Giannis Siglidis, and Michalis Vazirgiannis. Graph kernels: A survey. *J. Artif. Intell. Res.*, 72:943–1027, 2021. doi: 10.1613/JAIR.1.13225. URL https://doi.org/10.1613/jair.1.13225.

Nino Shervashidze, S. V. N. Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten M. Borgwardt. Efficient graphlet kernels for large graph comparison. In David A. Van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009*, volume 5 of *JMLR Proceedings*, pages 488–495. JMLR.org, 2009. URL http://proceedings.mlr.press/v5/shervashidze09a.html.

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. doi: 10.5555/1953048.2078187. URL https://dl.acm.org/doi/10.5555/1953048.2078187.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 1548–1554. ijcai.org, 2021. doi: 10.24963/IJCAI.2021/214. URL https://doi.org/10.24963/ijcai.2021/214.

Giannis Siglidis, Giannis Nikolentzos, Stratis Limnios, Christos Giatsidis, Konstantinos Skianis, and Michalis Vazirgiannis. Grakel: A graph kernel library in python. *J. Mach. Learn. Res.*, 21:54:1–54:5, 2020. URL https://www.jmlr.org/papers/v21/18-370.html.

Matteo Togninalli, M. Elisabetta Ghisu, Felipe Llinares-López, Bastian Rieck, and Karsten M. Borgwardt. Wasserstein weisfeiler-lehman graph kernels. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 6436–6446, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/73fed7fd472e502d8908794430511f4d-Abstract.html.

Titouan Vayer, Nicolas Courty, Romain Tavenard, Laetitia Chapel, and Rémi Flamary. Optimal transport for structured data with application on graphs. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6275–6284. PMLR, 2019. URL http://proceedings.mlr.press/v97/titouan19a.html.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

C. Villani and American Mathematical Society. *Topics in Optimal Transportation*. Graduate studies in mathematics. American Mathematical Society, 2003. ISBN 9781470418045. URL https://books.google.com/books?id=MyPjjgEACAAJ.

Haohui Wang, Yuzhen Mao, Yujun Yan, Yaoqing Yang, Jianhui Sun, Kevin Choi, Balaji Veeramani, Alison Hu, Edward Bowen, Tyler Cody, et al. Evolunet: Advancing dynamic non-iid transfer learning on graphs. In *Forty-first International Conference on Machine Learning*, 2024.

Boris Weisfeiler and AA Lehman. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, 2(9):12–16, 1968.

Zhang Xinyi and Lihui Chen. Capsule graph neural network. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Byl8BnRcYm.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Minkai Xu, Alexander S. Powers, Ron O. Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3d molecule generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 38592–38610. PMLR, 2023. URL https://proceedings.mlr.press/v202/xu23n.html.

Yujun Yan, Jiong Zhu, Marlena Duda, Eric Solarz, Chandra Sripada, and Danai Koutra. Groupinn: Grouping-based interpretable neural network for classification of limited, noisy brain data. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 772–782, 2019.

Pinar Yanardag and S. V. N. Vishwanathan. Deep graph kernels. In Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams, editors, *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 1365–1374. ACM, 2015. doi: 10.1145/2783258.2783417. URL https://doi.org/10.1145/2783258.2783417.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In Yike Guo and Faisal Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 974–983. ACM, 2018. doi: 10.1145/3219819.3219890. URL https://doi.org/10.1145/3219819.3219890.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3fe230348e9a12c13120749e3f9fa4cd-Abstract.html.

Zhen Zhang, Mianzhi Wang, Yijian Xiang, Yan Huang, and Arye Nehorai. Retgk: Graph kernels based on return probabilities of random walks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 3968–3978, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/7f16109f1619fd7a733daf5a84c708c1-Abstract.html.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph Contrastive Learning with Adaptive Augmentation. In *Proceedings of The Web Conference 2021*, WWW '21, pages 2069–2080, New York, NY, USA, April 2021. Association for Computing Machinery. ISBN 9781450370233. doi: 10.1145/3442381.3449802. URL https://doi.org/10.1145/3442381.3449802.

A Proof of Theorem 3.4

A.1 Learning Correct Similarity Order with Iterative Kernels

In this section, we first prove that the iterative kernel with bounded feature mapping is capable of learning the pairwise similarity order given sufficient training data, as shown by the PAC-learning framework. We consider a dataset consisting of instances $x \in \chi$, where χ is the input space. The dataset is denoted as $\chi = \{x_1, x_2, \ldots, x_n\}$. We employ a kernel function $\mathbb{K} : \chi \times \chi \to \mathbb{R}$ with an associated feature mapping $\phi : \chi \to \mathcal{H}_{\mathbb{K}}$ such that $\mathbb{K}(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}_{\mathbb{K}}}$. Here, $\mathcal{H}_{\mathbb{K}}$ is the Reproducing Kernel Hilbert Space (RKHS) associated with $\mathbb{K}$, and $\langle \cdot, \cdot \rangle_{\mathcal{H}\mathbb{K}}$ denotes the inner product in $\mathcal{H}_{\mathbb{K}}$. We assume that the feature mappings are bounded, i.e., $\|\phi(x)\|_{\mathcal{H}_{\mathbb{K}}} \leq R, \forall x \in \chi$ for some $R > 0$.

Using this setup, we focus on an arbitrary fixed data point x_i and compute its similarity with other data points $x_j (j \neq i)$ using the kernel function:

$$S = \mathbb{K}\left(x_i, x_j\right) = \left\langle \phi\left(x_i\right), \phi\left(x_j\right) \right\rangle_{\mathcal{H}_{\mathbb{K}}} \tag{1}$$

Thus, we define the function f as $f\left(x_j\right) = \left\langle w, \phi\left(x_j\right) \right\rangle_{\mathcal{H}_{\mathbb{K}}}$ where $w = \phi\left(x_i\right)$. Since $w = \phi\left(x_i\right)$ and $\left\|\phi\left(x_i\right)\right\|_{\mathcal{H}_{\mathbb{K}}} \leq R$, it follows that $\|w\|_{\mathcal{H}_{\mathbb{K}}} \leq R$. So the function f representing the similarity between x_i and any other point x_j can be formulated as:

$$f\left(x_j\right) = \left\langle \phi\left(x_i\right), \phi\left(x_j\right) \right\rangle_{\mathcal{H}_{\mathbb{K}}} = \mathbb{K}\left(x_i, x_j\right) = S_{ij}. \tag{2}$$

Since we are dealing with a ranking problem, we use the hinge loss function for pairs $\left(x_j, x_k\right) (j, k \neq i)$:

$$\ell_{\text{hinge}}\left(f, x_j, x_k\right) = \max\left(0, 1 - s_{jk}^*\left(f\left(x_j\right) - f\left(x_k\right)\right)\right), \tag{3}$$

where $s_{jk}^* = \operatorname{sign}\left(f^*\left(x_j\right) - f^*\left(x_k\right)\right)$, and f^* represents the ground-truth ranking function.

Thus, the expected loss over all pairs is given by:

$$L(f) = \mathbb{E}_{\left(x_j, x_k\right)}\left[\ell_{\text{hinge}}\left(f, x_j, x_k\right)\right], \tag{4}$$

and the empirical loss over the sample S is given by:

$$\hat{L}(f) = \frac{2}{(n-1)(n-2)} \sum_{\substack{j<k \\ j,k \neq i}} \ell_{\text{hinge}}\left(f, x_j, x_k\right), \tag{5}$$

since there are $\frac{(n-1)(n-2)}{2}$ pairs among the $n-1$ data points excluding x_i. To assess how well the learned function f generalizes to test data, we derive a generalization bound using Rademacher complexity. We define the function class for pairs by:

$$\mathcal{F}_{\text{pair}} = \left\{f\left(x_j, x_k\right) = f\left(x_j\right) - f\left(x_k\right)\right\}. \tag{6}$$

which represents the similarity difference between x_j and x_k given the reference graph x_i.

Given that $f(x) = \langle w, \phi(x) \rangle$ with $\|w\| \leq R$ and $\|\phi(x)\| \leq R$, we can bound $f\left(x_j, x_k\right)$ by:

$$\left|f\left(x_j, x_k\right)\right| = \left|f\left(x_j\right) - f\left(x_k\right)\right| \leq \left|f\left(x_j\right)\right| + \left|f\left(x_k\right)\right| \leq 2R^2, \tag{7}$$

since $|f(x)| = |\langle w, \phi(x) \rangle| \leq \|w\| \cdot \|\phi(x)\| \leq R \cdot R = R^2$.

Given $f\left(x_j, x_k\right)$ takes values in the range $\left[-2R^2, 2R^2\right]$. Therefore, following the PAC-learning framework in [Mohri, 2018], we derive the following generalization bound. For any $\delta > 0$, with probability at least $1-\delta$ over a sample S of size m drawn, the following holds for all $f \in \mathcal{F}_{\text{pair}}$:

$$L(f) \leq \hat{L}(f) + 2\hat{\mathfrak{R}}_S\left(\mathcal{F}_{\text{pair}}\right) + M\sqrt{\frac{\ln(1/\delta)}{2m}}, \tag{8}$$

where:

- $\hat{\mathfrak{R}}_m\left(\mathcal{F}_{\text{pair}}\right)$ is the empirical Rademacher complexity of $\mathcal{F}_{\text{pair}}$.

- $M = 1 + 2R^2$ is an upper bound on the hinge loss function ℓ_{hinge} .
- $m = \frac{(n-1)(n-2)}{2}$ is the number of pairs.
- δ is the confidence level.

It is observed that as m increases, the term $\sqrt{\frac{\ln(1/\delta)}{2m}}$ decreases at a rate proportional to $\sqrt{1/m}$. Next, we proceed by proving that the Rademacher complexity is bounded and decreases as m increases.

Theorem A.1 *Assuming $\|\phi(x)\|_{\mathcal{H}_K} \leq R$ for all $x \in \chi$, the empirical Rademacher complexity of the function class $\mathcal{F}_{pair}$ can be bounded as:*

$$\hat{\mathfrak{R}}_S(\mathcal{F}_{pair}) \leq \frac{2R^2}{\sqrt{m}}. \tag{9}$$

Proof 1 *Based on the definition of Rademacher complexity, for function class $\mathcal{F}_{pair}$ with a sample size m, the complexity $\hat{\mathfrak{R}}_S(\mathcal{F}_{pair})$ is defined as:*

$$\hat{\mathfrak{R}}_S\left(\mathcal{F}_{pair}\right) = \mathbb{E}_\sigma\left[\sup_{f \in \mathcal{F}_{pair}} \frac{1}{m} \sum_{i=1}^{m} \sigma_i f\left(x_{j_i}, x_{k_i}\right)\right], \tag{10}$$

where $\sigma = (\sigma_1, \ldots, \sigma_m)$ are independent Rademacher variables (i.e., each σ_k takes the value +1 or -1 with equal probability), and $\{(x_{j_i}, x_{k_i})\}_{i=1}^{m}$ are pairs sampled from the data.

We can express $\hat{\mathfrak{R}}_S\left(\mathcal{F}_{pair}\right)$ as:

$$\hat{\mathfrak{R}}_S\left(\mathcal{F}_{pair}\right) = \frac{1}{m}\mathbb{E}_\sigma\left[\sup_{\|w\| \leq R}\left\langle w, \sum_{i=1}^{m} \sigma_i\left(\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right)\right\rangle\right]. \tag{11}$$

Applying the Cauchy-Schwarz inequality:

$$\langle w, v\rangle_{\mathcal{H}_{\mathbb{K}}} \leq \|w\|_{\mathcal{H}_{\mathbb{K}}}\|v\|_{\mathcal{H}_{\mathbb{K}}} \leq R\|v\|_{\mathcal{H}_{\mathbb{K}}} \tag{12}$$

we have:

$$\hat{\mathfrak{R}}_S\left(\mathcal{F}_{pair}\right) \leq \frac{1}{m} R \cdot \mathbb{E}_\sigma\left[\left\|\sum_{i=1}^{m} \sigma_i\left(\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right)\right\|\right]. \tag{13}$$

Using Jensen's inequality in the form $\mathbb{E}[\|X\|] \leq \left(\mathbb{E}\left[\|X\|^2\right]\right)^{1/2}$, we obtain:

$$\hat{\mathfrak{R}}_S(\mathcal{F}_{pair}) \leq \frac{R}{m}\left(\mathbb{E}_\sigma\left[\left\|\sum_{k=1}^{m} \sigma_k\left(\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right)\right\|_{\mathcal{H}_K}^2\right]\right)^{1/2} \tag{14}$$

Since σ_i are independent Rademacher variables, we have $\mathbb{E}\sigma\left[\sigma_i^2\right] = 1$. Therefore:

$$\begin{aligned}\mathbb{E}_\sigma\left[\left\|\sum_{i=1}^{m} \sigma_i\left(\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right)\right\|^2\right] &\leq \sum_{i=1}^{m} \mathbb{E}_\sigma\left[\sigma_i^2\right]\left\|\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right\|^2 \\ &= \sum_{i=1}^{m}\left\|\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right\|^2 .\end{aligned} \tag{15}$$

Given that $\|\phi(y)\|_{\mathcal{H}_{\mathbb{K}}} \leq R$, we have:

$$\left\|\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right\|^2 \leq (2R)^2 = 4R^2. \tag{16}$$

Therefore,

$$\mathbb{E}_\sigma\left[\left\|\sum_{i=1}^{m} \sigma_i\left(\phi\left(x_{j_i}\right) - \phi\left(x_{k_i}\right)\right)\right\|\right] \leq \sqrt{m \cdot 4R^2} = 2R\sqrt{m}. \tag{17}$$

Thus

$$\hat{\mathfrak{R}}_S\left(\mathcal{F}_{\text{pair}}\right) \leq \frac{1}{m} R \cdot 2R\sqrt{m} = \frac{2R^2}{m}\sqrt{m} = \frac{2R^2}{\sqrt{m}}. \tag{18}$$

Therefore, the empirical Rademacher complexity decreases with m.

□

Thus, consider Equation 8:

- The term $\sqrt{\frac{\ln(1/\delta)}{2m}}$ decreases proportionally to $1/\sqrt{m}$.
- Empirical Rademacher complexity $\hat{\mathfrak{R}}_S\left(\mathcal{F}_{\text{pair}}\right) \leq \frac{2R^2}{\sqrt{m}}$ also decreases proportionally to $1/\sqrt{m}$.

Therefore, the generalization gap between $L(f)$ and $\hat{L}(f)$ decreases as m increases, demonstrating that the kernel-based ranking function is PAC-learnable.

A.2 Linking Two Key Properties to Generalizability in Binary Classification

Building on the setup and conclusions from the previous section, we establish that the kernel can learn the correct ranking with sufficient training data. Here, we investigate how, once the correct ranking is achieved, an iterative kernel with monotonic decrease and order consistency properties ensures a decreasing error rate for test set graphs as iterations progress in binary classification tasks.

To formalize this, we define a probability space over the whole graph space where addition and scalar multiplication are defined. Let Ω be the set of all possible data points (graphs). Each data point $g_i \in \Omega$ represents a possible outcome, and the event of observing g_i is the singleton set $\{g_j\} \subseteq \Omega$. By considering all subsets of Ω, we construct an event space $\mathcal{F}$. Assuming the existence of an underlying probability distribution over these outcomes, we obtain the probability space $(\Omega, \mathcal{F}, P)$. Using this framework, let μ_x and μ_y represent the unknown mean graph in class 1 and class 2, respectively. Let $\mathbb{K}_{gt}(\cdot,\cdot) \in [0,1]$ represent the ground-truth kernel function for pairs of graphs.

Without loss of generality, we analyze the error rate for graphs belonging to class 1; the error rate for class 2 can be derived in a similar manner. We define two random variables associated with the ground-truth kernel: $\mathbf{s}_x = \mathbb{K}_{gt}(\mu_x, x)$ and $\mathbf{s}_y = \mathbb{K}_{gt}(\mu_y, x)$, where x is a randomly sampled instance from class 1. The terms $\mathbf{s}_x(x)$ and $\mathbf{s}_y(x)$ denote specific realizations of $\mathbf{s}_x$ and $\mathbf{s}_y$, respectively, for a given realization of the graph x belonging to class 1. Since $\mathbf{s}_x = 0$ implies that there exists graph from class 1 that is completely different from μ_x, which is counterintuitive, we assume that the probability density function $p(u)$, $u \in [0,1]$, of $\mathbf{s}_x$ has the following asymptotic property on the right neighborhood of $u = 0$:

$$\lim_{u \to 0^+} \frac{p(u)}{u} = C, \tag{19}$$

where constant C satisfies that $0 \leq C < +\infty$. Let $\bar{\mathbf{s}}_x^{(i)}(x) = \mathbb{K}^{(i)}(\bar{\mu}_x^{(i)}, x)$ and $\bar{\mathbf{s}}_y^{(i)}(x) = \mathbb{K}^{(i)}(\bar{\mu}_y^{(i)}, x)$ represent the estimated realizations of $\mathbf{s}_x$ and $\mathbf{s}_y$ at iteration i, respectively, for a given graph realization x. $\bar{\mu}_x^{(i)}$ and $\bar{\mu}_y^{(i)}$ are estimates for μ_x and μ_y, respectively. We assume that $\bar{\mu}_x^{(i)}$ and $\bar{\mu}_y^{(i)}$ are obtained through operations on the training graphs, where these operations are closed in the graph space. Due to the capability of the iterative kernels, we further assume that as the iteration $i \to \infty$, the sequences $\{\bar{\mu}_x^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$, $\{\bar{\mu}_y^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$ and $\{\mathbb{K}^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$ converge to μ_x, μ_y and $\mathbb{K}_{gt}$, respectively. We assume that the convergence rate of $\{\mathbb{K}^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$ is much slower than the convergence rate of $\{\bar{\mu}_x^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$ and $\{\bar{\mu}_y^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$. Furthermore, every function in the series of function $\{\mathbb{K}^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$ is continuous.

Denote the training data for class 1 and class 2 as $\{x_1, x_2, \ldots, x_n\} \subset \Omega$ and $\{y_1, y_2, \ldots, y_m\} \subset \Omega$, respectively. Since the learned kernels $\mathbb{K}^{(i)}$ preserve the similarity order, we can assume, without loss of generality, the following empirical ordering of the training graphs for the kernel at the i-th iteration:

$$\bar{\mathbf{s}}_x^{(i)}(x_1) < \bar{\mathbf{s}}_x^{(i)}(x_2) < \cdots < \bar{\mathbf{s}}_x^{(i)}(x_n) \tag{20}$$

In this setting, for an arbitrary test graph g sampled from the graph space, the decision boundary of the kernel learned during the i-th iteration is defined as follows:

$$\begin{aligned}&\forall g:\ \text{if } \bar{\mathbf{s}}_x^{(i)}(g) > \bar{\mathbf{s}}_y^{(i)}(g) \text{ then } g \text{ is classified as class 1}\\&\forall g:\ \text{if } \bar{\mathbf{s}}_x^{(i)}(g) < \bar{\mathbf{s}}_y^{(i)}(g) \text{ then } g \text{ is classified as class 2}\end{aligned} \tag{21}$$

We assume that $\bar{\mathbf{s}}_x^{(i)}(g)$ and $\bar{\mathbf{s}}_y^{(i)}(g)$, as well as $\mathbf{s}_x(g)$ and $\mathbf{s}_y(g)$, are not equal for any of the training data points. Next, we analyze the misclassification rate for graphs in class 1. Let the integer k satisfies: $\bar{\mathbf{s}}_x^{(i)}(x_k) < \bar{\mathbf{s}}_y^{(i)}(x_k)$, while $\bar{\mathbf{s}}_x^{(i)}(x_{k+1}) > \bar{\mathbf{s}}_y^{(i)}(x_{k+1})$ or a given iteration i, which means x_k is the misclassified training graph in class 1 closest to the decision boundary. If the training error is 0, we take $x_0 = y_1$. We note that the integer k is independent of the iteration i due to the consistency principle. Next, we demonstrate this using the following lemma.

Lemma A.2 *For any sufficiently large iteration number i, there exists a fixed index k such that $\bar{\mathbf{s}}_x^{(i)}(x_k) < \bar{\mathbf{s}}_y^{(i)}(x_k)$, while $\bar{\mathbf{s}}_x^{(i)}(x_{k+1}) > \bar{\mathbf{s}}_y^{(i)}(x_{k+1})$.*

Proof 2 *Since the series $\{\bar{\mu}_x^{(i)}\}_{i=1}^{+\infty}$ converges to μ_x, it satisfies Cauchy's convergence criterion. Specifically, $\forall \delta > 0$, $\exists N_1 > 0$, such that when $i > N_1$,*

$$|\bar{\mu}_x^{(i+1)} - \bar{\mu}_x^{(i)}| < \delta. \tag{22}$$

Since we have assumed that $\mathbb{K}^{(i)}(\cdot,\cdot)$ are continuous: $\forall \epsilon > 0$, $\exists \delta > 0$, $\exists N_2 > 0$, such that for any x in class 1, $\forall i > N_2$ and $|\bar{\mu}_x^{(i+1)} - \bar{\mu}_x^{(i)}| < \delta$, we have:

$$|\mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i+1)}, x) - \mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x)| < \epsilon. \tag{23}$$

Based on Equation 23, we have:

$$\begin{aligned}&\bar{\mathbf{s}}_x^{(i+1)}(x_k) - \bar{\mathbf{s}}_y^{(i+1)}(x_k)\\=&\bar{\mathbf{s}}_x^{(i+1)}(x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k) + \mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k) - \bar{\mathbf{s}}_y^{(i+1)}(x_k) + \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k)\\<&|\bar{\mathbf{s}}_x^{(i+1)}(x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k)| + |\bar{\mathbf{s}}_y^{(i+1)}(x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k)| + (\mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k))\end{aligned} \tag{24}$$

Since the consistency principle holds, $\mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k)$ and $\bar{\mathbf{s}}_x^{(i)}(x_k) - \bar{\mathbf{s}}_y^{(i)}(x_k)$ have the same sign. If $\bar{\mathbf{s}}_x^{(i)}(x_k) - \bar{\mathbf{s}}_y^{(i)}(x_k) < 0$, then $\mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k) < 0$. For a sufficiently large N, we select an $\epsilon > 0$ such that $\epsilon < \min_{\{i>N\}} \frac{|\mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k)|}{2}$. Given our assumption that no training data point x_k lies precisely on the decision boundary, such an ϵ is guaranteed to exist. Consequently, Equation 24 simplifies to:

$$\bar{\mathbf{s}}_x^{(i+1)}(x_k) - \bar{\mathbf{s}}_y^{(i+1)}(x_k) < 2\epsilon + \mathbb{K}^{(i+1)}(\bar{\mu}_x^{(i)}, x_k) - \mathbb{K}^{(i+1)}(\bar{\mu}_y^{(i)}, x_k) < 0. \tag{25}$$

Thus, $\bar{\mathbf{s}}_x^{(i+1)}(x_k) - \bar{\mathbf{s}}_y^{(i+1)}(x_k)$ have the same sign as $\bar{\mathbf{s}}_x^{(i)}(x_k) - \bar{\mathbf{s}}_y^{(i)}(x_k)$. If $\bar{\mathbf{s}}_x^{(i)}(x_k) - \bar{\mathbf{s}}_y^{(i)}(x_k) > 0$, the same conclusion can be derived in a similar manner. Since the number of training samples is finite, we can find a sufficiently large iteration such that Equation 25 holds for all x_k after that iteration.

Thus, some point between $\bar{\mathbf{s}}_x^{(i)}(x_k)$ and $\bar{\mathbf{s}}_x^{(i)}(x_{k+1})$ is the decision boundary learned at the i-th iteration.

Based on our assumption that the convergence rate of $\{\mathbb{K}^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$ is much slower than the convergence rate of $\{\bar{\mu}_x^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$ and $\{\bar{\mu}_y^{(i)}(\cdot,\cdot)\}_{i=1}^{+\infty}$, $\bar{\mathbf{s}}_x^{(i)}(x_k)$ is asymptotically monotonically decreasing.

$$\bar{\mathbf{s}}_x^{(i)}(x_k) > \bar{\mathbf{s}}_x^{(i+1)}(x_k), \text{when } i \to +\infty \tag{26}$$

Thus,

$$\bar{\mathbf{s}}_x^{(i)}(x_k) \to \mathbb{K}_{gt}(\mu_x, x_k)^+ \text{ when } i \to +\infty \implies \bar{\mathbf{s}}_x^{(i)}(x_k) > \mathbb{K}_{gt}(\mu_x, x_k) \tag{27}$$

Thus, $\forall x \in$ class 1, the probability of misclassifying x is given by: $\mathbb{P}\{\bar{\mathbf{s}}_x^{(i)}(x_k) < \mathbb{K}_{gt}(\mu_x, x_k)\}$. Specifically, we have:

$$\mathbb{P}\{\mathbf{s}_x < \mathbb{K}_{gt}(\mu_x, x_k)\} \le \mathbb{P}\{\mathbf{s}_x \le \bar{\mathbf{s}}_x^{(i)}(x_k)\} \tag{28}$$

In the following theorem, we show that the probability of misclassifying any graph from class 1 can be bounded using Markov's inequality.

Theorem A.3 *If the iterative kernel decreases monotonically and preserves the similarity order, the probability of misclassifying a graph $x \in \chi$ from class 1 has an upper bound that decreases monotonically as the number of iterations increases.*

Proof 3 *To upper bound the error rate, we begin by applying the transformation $\frac{1}{\mathbf{s}_x} - 1$ to the original random variable $\mathbf{s}_x$. The expectation of this transformed variable is:*

$$\begin{aligned}\mathbb{E}\left(\frac{1}{\mathbf{s}_x} - 1\right) &= \int_0^1 \left(\frac{1}{u} - 1\right) p(u)\, du \\ &= \int_0^1 \frac{1}{u}\, p(u)\, du - 1\end{aligned} \tag{29}$$

Since $u = 0$ is singular point, we reformulate the improper integral as:

$$\lim_{\eta \to 0^+} \int_\eta^1 \frac{1}{u} p(u) du - 1$$

Here, since $\lim_{u \to 0^+} \frac{p(u)}{u} = C$, the limit exists and thus the given integral converges, ensuring the existence of the expectation. With this result, we can apply Markov's inequality to the transformed random variable $\frac{1}{\mathbf{s}_x} - 1$. For any nonnegative a, we have:

$$\mathbb{P}\left[(\frac{1}{\mathbf{s}_x} - 1) \geqslant a\right] \leqslant \frac{\mathbb{E}\left(\frac{1}{\mathbf{s}_x} - 1\right)}{a} \tag{30}$$

Thus

$$\mathbb{P}\left[\mathbf{s}_x \leqslant \frac{1}{1+a}\right] \leqslant \frac{\mathbb{E}\left(\frac{1}{\mathbf{s}_x} - 1\right)}{a} \tag{31}$$

Let $a = \frac{1}{\bar{\mathbf{s}}_x^{(\mathbf{i})}(x_k)} - 1$. Referring to Equation 28, we have:

$$\mathbb{P}\{\mathbf{s}_x < \mathbb{K}_{gt}(\mu_x, x_k)\} \leqslant \mathbb{P}\{\mathbf{s}_x \leq \bar{\mathbf{s}}_x^{(i)}(x_k)\} \leqslant \frac{\mathbb{E}\left(\frac{1}{\mathbf{s}_x} - 1\right)}{\frac{1}{\bar{\mathbf{s}}_x^{(\mathbf{i})}(x_k)} - 1} \tag{32}$$

Since $\bar{\mathbf{s}}_x^{(\mathbf{i})}(x_k)$ decreases monotonically and $\mathbb{E}\left(\frac{1}{\mathbf{s}_x} - 1\right)$ is a constant, it follows that $\mathbb{P}\{\mathbf{s}_x < \mathbb{K}_{gt}(\mu_x, x_k)\}$ has a monotonically decreasing upper bound.

B Theoretical Verification with WL-based Kernels

B.1 Proof of Theorem 3.5

Following Equation 2.4, the normalized WL-subtree kernel after h iterations can be expressed as:

$$\tilde{\mathbb{K}}^{(h)}_{wl\_subtree}(\mathcal{G}, \mathcal{G}') = \frac{\sum_{i=1}^{h} \langle \phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G}')) \rangle}{\sqrt{\sum_{i=1}^{h} \langle \phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G})) \rangle} \sqrt{\sum_{i=1}^{h} \langle \phi(\psi^i(\mathcal{G}')), \phi(\psi^i(\mathcal{G}')) \rangle}}$$

Next, we show by counterexample that the WL-subtree kernel may not preserve relational structure.

For example, consider two graphs, $\mathcal{G}$ and $\mathcal{G}'$, in the h-th iteration. The color (label) set of the nodes in these graphs is denoted as $\{C_a, C_b, C_c, \ldots\}$. Graph $\mathcal{G}$ contains 200 nodes labeled C_a and 4 nodes labeled C_b, whereas graph $\mathcal{G}'$ contains 4 nodes labeled C_a and 200 nodes labeled C_b. The similarity is computed using the vectors $\phi\left(\psi^i(\mathcal{G})\right) = [200, 4]$ and $\phi\left(\psi^i(\mathcal{G}')\right) = [4, 200]$. Consequently, the resulting similarity $\tilde{\mathbb{K}}^{(h)}_{wl\_subtree}$ is 0.0400.

In the next iteration, for graph $\mathcal{G}$, assume that half of the nodes currently labeled C_a are relabeled to C_c, while the other half are relabeled to C_d. Additionally, all nodes labeled C_b are relabeled to

C_e. As a result, the updated histogram vector for $\mathcal{G}$ becomes $\phi\left(\psi^{i+1}(\mathcal{G})\right) = [100, 100, 4]$. Similarly, in graph $\mathcal{G}'$, half of the nodes labeled C_a are relabeled to C_c and the remaining half to C_d, with all nodes labeled C_b relabeled to C_e, resulting in a histogram vector of $\phi\left(\psi^{i+1}(\mathcal{G}')\right) = [2, 2, 200]$. Here, the normalized similarity $\tilde{\mathbb{K}}^{(h)}_{wl\_subtree}$ is 0.0404, which exceeds $\tilde{\mathbb{K}}^{(h)}_{wl\_subtree}$.

This situation violates the principle of monotonic decrease, indicating that the WL-subtree kernel does not exhibit monotonicity—a phenomenon frequently observed in real-world datasets. □

B.2 Proof of Theorem 3.6

B.2.1 Monotonic Decrease in WLOA Kernel

The WLOA kernel after h iterations can be computed as follows:

$$\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}') = \sum_{i=1}^{h} \texttt{histmin}\left\{\phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G}'))\right\} \cdot \omega(i)$$

where $\omega(i)$ is a monotonically increasing function. The normalized kernel is given by:

$$\tilde{\mathbb{K}}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}') = \frac{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}')}{\sqrt{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G})}\sqrt{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}', \mathcal{G}')}}$$

At any iteration i, the expression $\texttt{histmin}\left\{\phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G}))\right\} \cdot \omega(i)$ equals to $\omega(i)|\mathcal{V}|$. Consequently, for the h-th iteration, this implies:

$$\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}) = \sum_{i=1}^{h} \omega(i)|\mathcal{V}|$$

Thus, the normalized kernel value for the graph pair $(\mathcal{G}, \mathcal{G}')$ can be expressed as:

$$\tilde{\mathbb{K}}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}') = \frac{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}')}{\sqrt{|\mathcal{V}||\mathcal{V}'|}\sum_{h}^{i=1}\omega(i)}$$

Given this, the difference between the kernel value at the h-th iteration, $\tilde{\mathbb{K}}^{(h)}_{WLOA}$, and the $(h+1)$-th iteration, $\tilde{\mathbb{K}}^{(h+1)}_{WLOA}$, can be expressed as follows:

$$\begin{aligned}
&\tilde{\mathbb{K}}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}') - \tilde{\mathbb{K}}^{(h+1)}_{WLOA}(\mathcal{G}, \mathcal{G}') \\
=&\frac{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}')}{\sqrt{|\mathcal{V}||\mathcal{V}'|}\sum_{i=1}^{h}\omega(i)} - \frac{\mathbb{K}^{(h+1)}_{WLOA}(\mathcal{G}, \mathcal{G}')}{\sqrt{|\mathcal{V}||\mathcal{V}'|}\sum_{i=1}^{h+1}\omega(i)} \\
=&\frac{1}{\sqrt{|\mathcal{V}||\mathcal{V}'|}} \cdot \left(\frac{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}')}{\sum_{i=1}^{h}\omega(i)} - \frac{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}') + \texttt{histmin}\{\phi(\psi^{h+1}(\mathcal{G})), \phi(\psi^{h+1}(\mathcal{G}'))\omega(h+1)\}}{\sum_{i=1}^{h+1}\omega(i)}\right) \\
:=&D
\end{aligned}$$

Considering the hierarchical structure of the refinement processes, one can deduce the following inequality:

$$\texttt{histmin}\{\phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G}'))\} \leq \texttt{histmin}\{\phi(\psi^{i-1}(\mathcal{G})), \phi(\psi^{i-1}(\mathcal{G}'))\} \leq \cdots \texttt{histmin}\{\phi(\psi^1(\mathcal{G})), \phi(\psi^1(\mathcal{G}'))\}$$

Thus, for the kernel at the $(h+1)$-th iteration, it follows that:

$$\texttt{histmin}\{\phi(\psi^{h+1}(\mathcal{G})), \phi(\psi^{h+1}(\mathcal{G}'))\} \leq \frac{\sum_{i=1}^{h}\texttt{histmin}\{\phi(\psi^i(\mathcal{G})), \phi(\psi^i(\mathcal{G}')) \cdot \omega(i)\}}{\sum_{i=1}^{h}\omega(i)} = \frac{\mathbb{K}^{(h)}_{WLOA}(\mathcal{G}, \mathcal{G}')}{\sum_{i=1}^{h}\omega(i)}$$

Given this inequality, we get

$$
\begin{aligned}
D &\geq \frac{1}{\sqrt{|\mathcal{V}||\mathcal{V}'|}} \cdot \Big(\frac{\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}')}{\sum_{i=1}^{h}\omega(i)} - \frac{\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}') + \frac{\omega(h+1)}{\sum_{i=1}^{h}\omega(i)}\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}')}{\sum_{i=1}^{h+1}\omega(i)}\Big) \\
&= \frac{1}{\sqrt{|\mathcal{V}||\mathcal{V}'|}} \cdot \Big(\frac{\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}')}{\sum_{i=1}^{h}\omega(i)} - \frac{\sum_{i=1}^{h}\omega(i)\frac{\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}')}{\sum_{i=1}^{h}\omega(i)} + \frac{\omega(h+1)}{\sum_{i=1}^{h}\omega(i)}\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}')}{\sum_{i=1}^{h+1}\omega(i)}\Big) \qquad (33) \\
&= \frac{1}{\sqrt{|\mathcal{V}||\mathcal{V}'|}} \cdot \frac{\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}')}{\sum_{i=1}^{h}\omega(i)} \cdot \Big(1 - \frac{\sum_{i=1}^{h}\omega(i) + \omega(h+1)}{\sum_{i=1}^{h}\omega(i) + \omega(h+1)}\Big) \\
&= 0
\end{aligned}
$$

Therefore, we get $D \geq 0$. The value of Equation 33 is non-negative, indicating that WLOA is monotonically decreasing.

B.2.2 Order Consistency in WLOA Kernel

Assume two graph pairs satisfy the following relation:

$$\tilde{\mathbb{K}}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}') \geq \tilde{\mathbb{K}}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}'')$$

The normalized WLOA kernel for a graph pair $(\mathcal{G},\mathcal{G}')$ at iteration $h+1$ can be expressed as:

$$\tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}') = \frac{\mathbb{K}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}')}{\sqrt{|\mathcal{V}|\,|\mathcal{V}'|}\sum_{i=1}^{h+1}\omega(i)} = \frac{\sum_{i=1}^{h}\omega(i)}{\sum_{i=1}^{h+1}\omega(i)} \cdot \frac{\mathbb{K}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}')}{\sqrt{|\mathcal{V}|\,|\mathcal{V}'|}\sum_{i=1}^{h}\omega(i)}$$

$\mathbb{K}_{WLOA}^{(h)}$ is monotonically increasing with the iteration number h, as it follows from: $\mathbb{K}_{WLOA}^{(h+1)} = \mathbb{K}_{WLOA}^{(h)} + \texttt{histmin}\left\{\phi(\psi^{h+1}(\mathcal{G})), \phi(\psi^{h+1}(\mathcal{G}'))\right\} \cdot \omega(h+1)$. Thus, we have:

$$
\begin{aligned}
\tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}') &\geq \frac{\sum_{i=1}^{h}\omega(i)}{\sum_{i=1}^{h+1}\omega(i)} \cdot \frac{\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}')}{\sqrt{|\mathcal{V}||V'|}\sum_{i=1}^{h}\omega(i)} \\
&\geq \frac{\sum_{i=1}^{h}\omega(i)}{\sum_{i=1}^{h+1}\omega(i)} \cdot \frac{\mathbb{K}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}'')}{\sqrt{|\mathcal{V}||\mathcal{V}''|}\sum_{i=1}^{h}\omega(i)} \\
&= \frac{\sum_{i=1}^{h}\omega(i)}{\sum_{i=1}^{h+1}\omega(i)} \cdot \tilde{\mathbb{K}}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}'')
\end{aligned}
$$

Given the monotonic decrease property of $\tilde{\mathbb{K}}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}'')$,we have:

$$\tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}'') \leq \tilde{\mathbb{K}}_{WLOA}^{(h)}(\mathcal{G},\mathcal{G}'')$$

Thus, we can conclude:

$$
\begin{aligned}
\tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}') &\geq \frac{\sum_{i=1}^{h}\omega(i)}{\sum_{i=1}^{h+1}\omega(i)}\tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}'') \\
&= \left(1 - \frac{\omega(h+1)}{\sum_{i=1}^{h+1}\omega(i)}\right)\tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}'')
\end{aligned}
$$

When $\omega(i) = 1$, $\lim_{h\to\infty}\frac{\omega(h+1)}{\sum_{i=1}^{h+1}\omega(i)} = \lim_{h\to\infty}\frac{h+1}{(h+2)(h+1)/2} = \lim_{h\to\infty}\frac{2}{h+2} = 0$. Therefore, $\tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}') \geq \tilde{\mathbb{K}}_{WLOA}^{(h+1)}(\mathcal{G},\mathcal{G}'')$ when $h \to \infty$. □

Table 5: Dataset statistics.

Dataset	Task Description	# Class	# Size	Ave.Nodes	Ave.Edges	Node Label.
ogbg-molhiv	Molecular property prediction	2	41127	25.5	54.1	+
PROTEINS	Enzyme classification	2	1113	39.06	72.82	+
D&D	Enzyme classification	2	1178	284.32	715.66	+
NCI1	Molecular classification	2	4110	29.87	32.30	+
NCI109	Molecular classification	2	4127	29.68	32.13	+
IMDB-B	Movie venue categorization	2	1000	19.77	96.53	-
IMDB-M	Movie venue categorization	3	1500	13.00	65.94	-
COLLAB	Collaboration classification	3	5000	74.49	2457.78	-
COIL-RAG	Computer Vision	100	3900	3.01	3.02	-
Reddit-T	Reddit Thread Classification	2	203088	23.93	24.99	-

C Dataset

In this section, we provide the statistics of the datasets used, as shown in Table 5.

D Detailed Set-Up

TU Dataset We restrict the hyperparameters and ensure the same architecture is used for both the base and enhanced models on the same dataset for a fair comparison. Specifically, we fixed the hidden size to 32, the number of layers to 3, and used a global mean pooling layer to generate the graph representations. The Adam optimizer was used for optimization. During the training process, we tuned both the base and enhanced models with the same search ranges: batch size {64, 128}, dropout rate {0, 0.3} and learning rate {0.0001, 0.001, 0.01}. The only additional parameter for the enhanced model is the regularization term, which ranges from {0.1, 0.5, 1, 10}.

OGB Considering the complexity of the OGB dataset, we slightly expand the hyperparameter search range. Initially, we train the model with consistency loss, exploring hidden sizes {32, 64, 128, 256}, batch sizes {64, 128, 256}, and the number of layers {3, 4}, while keeping other parameters consistent with our experiments on the TU dataset. Subsequently, we fix the hidden size and the number of layers to ensure an identical network structure. We then repeat the parameter search process and employ the optimal settings for testing on the base model. All experiments on the OGB dataset are repeated 5 times to calculate the mean and standard deviation.

Reddit-T Given that the Reddit-T and OGB datasets are of comparable size, we employ similar experimental settings for training models on these datasets. The architecture is kept fixed, and we search for optimal hyperparameters for both the base model and the models incorporating consistency loss. Each model is trained and evaluated over 5 independent runs, with the mean and standard deviation of the results recorded for performance comparison.

E Complexity& Scalability

E.1 Time Complexity

We present the time complexity analysis for our proposed consistency loss. The loss computation involves the computation of pairwise similarities of graphs in a batch, resulting in a computation complexity of $O\left(\text{batchsize} \cdot \frac{\text{batchsize}-1}{2}\right) = O(\text{batchsize}^2)$. Given that there are $\frac{\text{datasetsize}}{\text{batchsize}}$ batches in each training epoch and that the similarities are computed between consecutive layers, the total complexity is:

$$O(\text{ loss }) = O\left\{ \text{batchsize}^2 \times (\text{ layernum } - 1) \times \frac{\text{datasetsize}}{\text{batchsize}} \right\}$$
$$= O(\text{ datasetsize} \times \text{batchsize} \times \text{layernum }).$$

This analysis shows that the time required to compute consistency loss scales linearly with dataset size, batch size, and the number of layers. It is important to note that the training time for baseline models also scales linearly with dataset size.

Since batch size and the number of layers are generally small compared to dataset size, our experiments primarily focus on how dataset size affects training time. We evaluate the training time of several models—GCN, GIN, GraphSAGE, GTransformer, and GMT—each enhanced with our consistency loss. This evaluation is conducted on different subsets of the ogbg-molhiv dataset, with subset sizes adjusted by varying the sampling rates. The training times, measured in seconds, are presented in Figure 3 . As shown, our findings confirm that training time increases linearly with dataset size, indicating that our method maintains training efficiency comparable to baselines without adding significant time burdens.

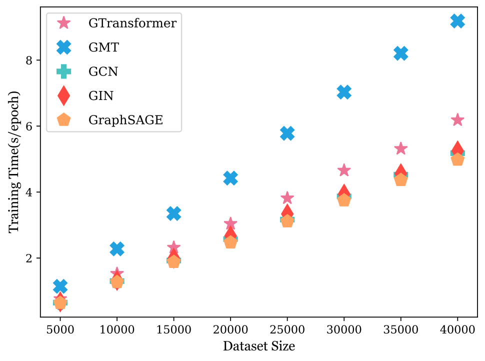


Figure 3: Training Cost Escalates Linearly with Dataset Size Increase

Furthermore, we empirically measure the training time for both the baseline models and our proposed methods. Each model comprises three layers and is trained on the ogbg-molhiv dataset (40,000+ graphs) for 100 epochs. We calculate the average training time per epoch in seconds and present the results in Table 6, showing that while the inclusion of the consistency loss slightly increases the training time, the impact is minimal.

Table 6: Average training time per epoch for different models on the ogbg-molhiv dataset, measured in seconds.

	GMT	**GTransformer**	**GIN**	**GCN**	**GraphSAGE**
GCN	8.380	4.937	4.318	4.221	3.952
GCN+$\mathcal{L}_{\text{consistency}}$	8.861	6.358	5.529	5.382	5.252

E.2 Space Complexity

Next, we present the space complexity analysis for our consistency loss. At each iteration, the loss function requires storing two pairwise similarity matrices corresponding to two consecutive layers, which is given by:

$$O(\text{ loss }) = O(\text{batchsize}^2)$$

Since we use stochastic gradient descent, similarity matrices are not retained for the next iteration. The consistency loss requires significantly less space than node embeddings, making the additional space requirement minimal. Table 7 shows the peak memory usage in megabytes (MB) for different models when training on the ogbg-molhiv dataset, illustrating that the space costs are negligible.

Table 7: Peak memory usage for different models on the ogbg-molhiv dataset, measured in megabytes.

	GMT	**GTransformer**	**GIN**	**GCN**	**GraphSAGE**
GCN	1334.0	1267.8	1291.3	1274.2	1288.4
GCN+$\mathcal{L}_{\text{consistency}}$	1370.0	1330.6	1338.9	1320.1	1321.3
Cost Increase (%)	2.70	4.96	3.68	3.60	2.55

E.3 Efficiency on Different Task and Structural Complexities

Task Complexity We measured the runtime of the models on different subsets to evaluate how task complexity, in terms of the number of classes, influences the efficiency of the proposed method. The results are presented in Table 8. As demonstrated, the additional computational time remains minimal even with an increasing number of classes, suggesting that the method scales effectively with growing class complexity.

Table 8: Average training time per epoch on REDDIT subsets with varying class complexity, measured in seconds

	Subset1 (2 classes)	**Subset2 (3 classes)**	**Subset3 (4 classes)**	**Fullset (5 classes)**
GCN	0.203	0.345	0.408	0.493
GCN+$\mathcal{L}_{\text{consistency}}$	0.227	0.355	0.430	0.557

Structure Complexity We also conducted experiments to assess the training costs on datasets with varying structural complexities when introducing the $\mathcal{L}_{\text{consistency}}$. The results, summarized below in Table 9, show that the additional training cost remains minimal across datasets with different structures. This demonstrates the broad applicability of the proposed method, regardless of structural complexity.

Table 9: Average training time per epoch for subsets of varying structural complexity from IMDB-B, measured in seconds.

	IMDB-B **(small)**	IMDB-B **(medium)**	IMDB-B **(large)**
GCN	0.0308	0.0311	0.0321
GCN+$\mathcal{L}_{\text{consistency}}$	0.0371	0.0378	0.0392

F Efficient Consistent Learning

To further minimize the overhead of our proposed consistency loss, we examined a scenario where the consistency loss, denoted as $\mathcal{L}_{FL}$, was only applied to the first and last layers.

Building upon the experimental setup described in Section 5, we conducted experiments using various backbone models. The results are summarized in Table 10. The penultimate column of this table highlights the performance gains achieved by applying the consistency loss across all layers, while the final column demonstrates the improvements observed when the consistency loss is only applied to the first and last layers.

Table 10: Graph classification performance with improvements of $\mathcal{L}_{ALL}$ and $\mathcal{L}_{FL}$ over base models.

	NCI1	**NCI109**	**PROTEINS**	**DD**	**IMDB-B**	**OGB-HIV**	$\mathcal{L}_{ALL}\uparrow$	$\mathcal{L}_{FL}\uparrow$
GCN+$\mathcal{L}_{FL}$	$75.96_{\pm 0.89}$	$74.67_{\pm 1.11}$	$72.97_{\pm 2.85}$	$76.27_{\pm 1.69}$	$74.6_{\pm 1.85}$	$74.44_{\pm 1.42}$	+5.49	+7.08
GIN+$\mathcal{L}_{FL}$	$79.08_{\pm 1.21}$	$77.0_{\pm 2.01}$	$73.15_{\pm 2.76}$	$74.07_{\pm 1.38}$	$74.8_{\pm 4.66}$	$74.2_{\pm 1.62}$	+10.95	+15.12
GraphSAGE+$\mathcal{L}_{FL}$	$78.88_{\pm 2.01}$	$74.24_{\pm 1.21}$	$75.32_{\pm 2.46}$	$73.90_{\pm 2.03}$	$76.6_{\pm 1.96}$	$80.06_{\pm 1.21}$	+9.10	+9.71
GTransformer+$\mathcal{L}_{FL}$	$76.79_{\pm 1.24}$	$74.38_{\pm 0.49}$	$73.69_{\pm 2.09}$	$75.08_{\pm 1.57}$	$76.80_{\pm 1.60}$	$80.53_{\pm 0.73}$	+9.00	+8.63
GMT+$\mathcal{L}_{FL}$	$76.40_{\pm 1.00}$	$75.64_{\pm 0.77}$	$72.25_{\pm 3.96}$	$73.39_{\pm 2.18}$	$76.00_{\pm 1.36}$	$81.05_{\pm 1.29}$	+6.23	+5.24

Notably, applying the consistency loss only to the first and last layers achieves performance comparable to that of applying it across all layers, with both configurations yielding substantial improvements over the original model. This finding suggests that our proposed approach can be accelerated with minimal additional computational cost while still enhancing performance, thereby validating the effectiveness of the consistent learning principle.

G Similarity/Difference with Contrastive learning

In this section, we discuss the similarities and differences between our method and graph contrastive learning. Graph Contrastive Learning (GCL) is a self-supervised technique for graph data that emphasizes instance discrimination [Lin et al., 2023, Zhu et al., 2021]. A typical GCL framework generates multiple graph views via augmentations and contrasts positive samples (similar instances) with negative samples (dissimilar instances). This approach facilitates effective representation learning by capturing relationships between views, ensuring positive pairs remain close in the embedding space while distinctly separating negative pairs.

While both GCL and our method leverage graph similarity, our approach focuses on **maintaining consistency across layers, rather than solely capturing similarities as in contrastive learning.** To demonstrate this, we integrated the GraphCL technique [You et al., 2020] into a GCN model (GCN+CL) and assessed its performance and layer consistency across various datasets. The results, detailed in Tables 11 and 12, use classification accuracy and Spearman rank correlation to measure performance and consistency, respectively.

Table 11: Graph classification accuracy of GCN with contrastive learning applied across various datasets.

	NCI1	**NCI109**	**PROTEINS**	**D&D**	**IMDB-B**
GCN+ CL	74.06 ± 1.91	73.14 ± 1.90	72.50 ± 2.73	75.80 ± 2.09	75.80 ± 1.90
GCN+$\mathcal{L}_{\text{consistency}}$	75.12 ± 1.19	73.25 ± 1.25	75.07 ± 5.05	78.56 ± 3.32	75.85 ± 1.82

Table 12: Spearman correlation for graph representations from consecutive layers.

	NCI1	**NCI109**	**PROTEINS**	**D&D**	**IMDB-B**
GCN+ CL	0.835	0.717	0.851	0.717	0.810
GCN+$\mathcal{L}_{\text{consistency}}$	0.859	0.958	0.946	0.896	0.907

As demonstrated by the results, our method consistently outperforms GCN+CL in both graph classification performance and in enhancing similarity consistency across layers. This underscores the significant differences between our approach and regular GCL methods.

H Boarder Impact

This paper aims to advance the field of graph learning by proposing a model-agnostic consistency learning framework. Our framework can be plugged into and improve current methods for graph classification tasks. This has potential benefits in sectors such as chemistry, bioinformatics, and social analysis, where graph classification is widely used. Furthermore, we do not foresee direct negative societal or ethical consequences arising from our work.

I Limitation

One limitation of our work is that the method involves additional computational costs, especially during large batch training processes. To extend our framework, sampling methodologies on data or layers can be applied during the consistency-preserving training process. By selectively sampling data points or specific layers, we can reduce the computational burden while still maintaining the effectiveness of the cross-layer consistency loss, making the framework more scalable and applicable to larger datasets.

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: The experiments and results to support the claims in the introduction and abstract are all in the main paper and the appendix.

   Guidelines:

   - The answer NA 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 NA 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: The limitations are discussed in the main paper Sec. I Limitations.

   Guidelines:

   - The answer NA 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: [Yes]

   Justification: This paper include theoretical result and proof involved in main paper and appendix.

Guidelines:

- The answer NA 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: The training schedule and hyper-parameters are listed in Appendix Sec. D.

   Guidelines:

   - The answer NA 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: The data are public, and code is provided in the anonymous github.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.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://nips.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, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have listed all the settings in the appenix for paramtere settings and training schedules. The data splits are decribed under each case study section in the main paper.

   Guidelines:

   - The answer NA 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: All of our results are listed with error bars.

   Guidelines:

   - The answer NA 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: The compute resources of experiments are reported in the appendix.

   Guidelines:

   - The answer NA 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: The research conducted in this paper conforms, in every respect, to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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: The broader impact is discussed in the appendix H.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer:[NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA 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: They are properly cited and credited.

    Guidelines:

    - The answer NA 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: the new model introduced in the paper is well documented.

    Guidelines:

- The answer NA 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: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA 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.