Beyond Coordinates: Encoding Graph Structure via Contextual Distribution and Relational Similarity

Feifei Qian[1], **Lu Bai**[1,*], **Lixin Cui**[2], **Ming Li**[3],
Hangyuan Du[4], **Bo Jiang**[5], **Lixiang Xu**[6], **Edwin R. Hancock**[7]

[1]School of Artificial Intelligence, Beijing Normal University, Beijing, China
[2]School of Information, Central University of Finance and Economics, Beijing, China
[3]Zhejiang Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University, Jinhua, China
[4]School of Computer and Information Technology, Shanxi University, Taiyuan, China
[5]School of Computer Science and Technology, Anhui University, Hefei, China
[6]School of Artificial Intelligence, Hefei Institute of Technology, Hefei, China
[7]Department of Computer Science, University of York, York, United Kingdom

feifei_qian@mail.bnu.edu.cn, bailu@bnu.edu.cn,

Abstract

The Message Passing Neural Networks (MPNNs) and Graph Transformers (GTs) have emerged as two dominant paradigms for graph representation learning. However, the expressive power of standard MPNNs is fundamentally bounded by the 1-dimensional Weisfeiler-Lehman (1-WL) test, while GTs lack the inductive bias for graph structure. To enhance structural representation, existing methods typically resort to subgraph-based aggregation or coordinate-based positional encodings. However, the former suffers from prohibitive computational memory overheads, while the latter is limited by rigid reference frames that fail to distinguish fine-grained local topology. To address these limitations, we introduce a novel structural encoding framework based on contextual distributions. Specifically, we move beyond fixed coordinate systems to capture intrinsic local topology by encoding structural information through a compact distributional statistic, i.e., the entropy of the node context. Furthermore, instead of relying on relative positioning, we introduce a kernelized mechanism to encode relational similarity by quantifying the structural affinity between node contexts. Extensive experiments on both synthetic and real-world datasets demonstrate that the proposed framework achieves superior performance, striking a favorable balance between effectiveness and efficiency.

1 Introduction

Graph Neural Networks (GNNs) have emerged as a powerful tool for learning on graph-structured data [21]. Among the various GNN architectures, the Message Passing Neural Networks (MPNNs) have become one of the dominant paradigms [20], as their neighborhood aggregation mechanism naturally encodes local structural inductive bias, resulting in strong performance across a broad range of graph representation learning tasks. Nevertheless, as demonstrated in the Graph Isomorphism Network (GIN) [53], the expressive power of most existing MPNNs is fundamentally bounded by the 1-dimensional Weisfeiler-Lehman (1-WL) test [46]. To overcome this limitation, finer-grained structural encoding is essential for enhancing the expressive power of MPNNs. Meanwhile, inspired by the success of the Transformer in natural language processing [49], an increasing number of

*Corresponding author: Lu Bai (bailu@bnu.edu.cn).

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

studies have adopted this architecture for graph learning, giving rise to the new paradigm of Graph Transformers (GTs) [15, 45]. GTs can be viewed as performing message passing on fully connected graphs, thereby performing global attention but neglecting the sparse connectivity and topological priors essential for graph data. Consequently, how to efficiently and effectively encode graph structural information remains a key challenge in the graph learning field.

To address this challenge, existing approaches typically follow two primary strategies to encode structural information. The first line of work focuses on architectural designs, most notably subgraph-based GNNs. These methods explicitly decompose the input graph into collections of local subgraphs and encode them via message passing performed within each subgraph, thereby enriching node representations with higher-order structural information [6, 61]. While such architectures are theoretically expressive, processing bags of subgraphs inherently incurs prohibitive memory and computational overheads, limiting their scalability to large-scale graphs. The second strategy focuses on feature augmentation by directly injecting additional node features, i.e., the commonly used Positional and Structural Encodings (PSE) [45]. Several works explicitly augment node features with structural descriptors derived from substructure patterns, ranging from heuristic counts of predefined motifs, e.g., cycles and cliques [48, 8], to formalized homomorphism counts [5]. However, these approaches often suffer from poor generalization due to their reliance on predefined patterns. More recently, with the rise of GTs, an increasing line of work has explored continuous spectral and statistical encodings as a means to inject structural inductive bias. A common thread among these methods is that they incorporate explicit positional information, either as absolute coordinates derived from spectral embeddings [27, 34, 50, 30, 25] or as relative coordinates encoded by pairwise distances [57, 29, 12], to enrich node representations. However, these encodings face limitations in capturing fine-grained local topology. For instance, Laplacian eigenvectors suffer from sign and basis ambiguities that require specialized invariant architectures. Distance-based descriptors capture only pairwise relationships and miss higher-order local structure. These deficiencies highlight the critical need for a structural encoding paradigm that moves beyond fixed coordinate systems, effectively capturing fine-grained local topology while mitigating the excessive memory overhead inherent in subgraph-based GNNs.

Departing from these prevailing paradigms, we propose a novel structural encoding method that conceptualizes structural information not as discrete counts or absolute positions, but from the new perspective of **contextual distributions**, i.e., the distributions of node ego-subgraph. As a theoretical foundation, we formally demonstrate that incorporating contextual distribution is sufficient to strictly enhance expressive power beyond the standard 1-WL test. Building on this theoretical insight, we instantiate our framework by introducing **Entropy Structural Encoding (EntSE)**, which leverages entropy derived from spectral distributions as a compact yet expressive descriptor of local structural complexity. This work marks a pioneering exploration of the link between expressiveness and the distribution of eigenvalues.

Moreover, we move beyond absolute structural descriptions by introducing KerSE, a kernelized instantiation of our framework that formalizes **relational similarity** through pair-level structural resonance. Instead of relying on Euclidean distances or rigid positional embeddings, the KerSE maps structural features into a normalized similarity space defined over their contextual entropies, thereby mitigating the impact of extreme structural outliers. This kernelized perspective enables stable and semantically meaningful message propagation, offering a robust alternative to absolute or relative positioning. Overall, the proposed framework, instantiated through EntSE and KerSE, achieves a favorable trade-off between expressive power and computational efficiency. It also offers a principled structural inductive bias that can be seamlessly integrated into modern graph learning backbones, including MPNNs and Graph Transformers. The main contributions are summarized as follows.

- **A Novel Contextual Distribution Encoding:** We propose a general structural encoding framework that explicitly models the distribution of node contexts.
- **Theoretical Guarantees:** We provide theoretical analysis of structural encodings based on contextual distributions, characterizing the entropy dynamics under structural perturbations.
- **Relational Similarity via Kernel-based Variant:** We introduce a kernelized mechanism that moves beyond coordinate systems to encode relational similarity, and we present an *efficient realization* that accelerates the computation through neighborhood sampling.
- **Experimental Results:** We show that our method distinguishes graph pairs that are indistinguishable by 1-WL to 3-WL tests and achieves superior performance on real-world datasets, offering a favorable trade-off between expressive power and computational efficiency.

2 Related Works

Since the expressive power of standard MPNN is fundamentally upper-bounded by the 1-WL isomorphism test [53], this intrinsic limitation naturally motivates a line of work that seeks to enhance MPNNs by emulating the more powerful k-dimensional WL (k-WL) hierarchy. Indeed, some works have been proposed to approximate k-WL by performing message passing over k-tuples of nodes [36, 33, 2, 38]. However, the computational costs of these models grow exponentially with k, making them impractical for large-scale graph datasets. Consequently, the research focus has shifted towards more scalable strategies, primarily subgraph-based architectures and positional or structural encoding (PSE) methods.

Subgraph-based GNNs. Since the 1-WL test can not distinguish the non-isomorphic graphs with high symmetry, various subgraph-based methods are proposed. The Nested GNN [60] and GNN-AK+ [61] propose to perform message passing independently on the subgraph rooted at each node, implementing a subgraph-level GNN. Furthermore, the ESAN [6] proposes a subgraph-based GNN framework that performs message passing within each subgraph, and aggregates their representations in a permutation-invariant manner. These methods require performing MPNN on full sets of subgraphs or rely on additional policies to sample subgraph collections. Thus, the OSAN [43] and the Policy-learn [7] respectively propose adaptive methods for learning sampled subgraphs, rather than relying on handcrafted subgraph selection. However, these methods face limitations in performance due to the sampling bias and additional computational overhead.

Positional and Structural Encoding (PSE) Methods. Existing literature on PSE can be systematically taxonomized into four primary paradigms. The first category involves node identity and random features. For instance, the ID-GNN [58] and CLIP [14] assigns a unique identity to each node, while the GCN-RNI [1] enhances discriminative power by injecting randomly initialized features. However, these methods typically compromise permutation invariance, limiting their inductive generalization. The second paradigm focuses on local structural statistics, such as node degree centrality [56], Random Walk Structure Encoding (RWSE) [18], random walk probability encodings [31], and substructure information [48, 8, 55]. These approaches are typically lightweight, but their expressiveness is often limited. The third class leverages Laplacian eigenvectors. As pointed out by [26], a GNN can be viewed as a nonlinear mapping applied to the input eigenvectors. LapPE [27] directly uses Laplacian eigenvectors as node features, while subsequent methods such as SignNet [30] and SPE [23] introduce invariant transformations to mitigate the sign and basis ambiguities inherent in spectral decomposition. Despite these advances, these encodings remain fundamentally tied to global eigenstructures. Complementary to absolute encodings, relative positional encodings capture pairwise structure using distances such as shortest-path [29] or resistance distance [59]. Orthogonal to the above, recent work explores pre-trained structural encodings, e.g., GPSE [9] or GFSE [11]. These approaches generally require extra data and pre-training to obtain transferable representations.

3 Preliminary Concepts

Notations. Let $G = (V, E)$ be an undirected graph, where V denotes the set of nodes and E denotes the set of edges. For a node u, let $\mathcal{N}_u^k$ represent the set of its k-hop neighbors. We define the neighborhood up to k-hops as $\mathcal{N}_u^{1:k} = \bigcup_{i=1}^{k} \mathcal{N}_u^i$.

Von Neumann Entropy of Graphs. Given a graph G, let A be its adjacency matrix and D be degree matrix. The Laplacian matrix can be computed as $L = D - A$. Let $\{\lambda_i\}_{i=1}^{|V|}$ denote the eigenvalues of L, and let $\mathrm{tr}(L)$ represent the trace of the Laplacian matrix. The von Neumann entropy of G is:

$$\mathcal{H}_{\mathrm{VN}}(G) = -\sum_{i=1}^{|V|} \frac{\lambda_i}{\mathrm{tr}(L)} \log\left(\frac{\lambda_i}{\mathrm{tr}(L)}\right). \tag{1}$$

Graph Isomorphism. Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, if there exists a bijection $f : V_1 \to V_2$ such that for every edge $(u, v) \in E_1$, there is a corresponding edge $(f(u), f(v)) \in E_2$, then G_1 and G_2 are isomorphic. Since there is no clear metric that can definitively characterize the structure of a graph, the graph isomorphism problem still has no known polynomial-time algorithm.

Weisfeiler-Lehman Isomorphism Test. The Weisfeiler-Lehman (1-WL) test is a classic algorithm used to test graph isomorphism [46]. It operates by iteratively refining the node labels by aggregating

the neighborhood information. Given a graph G, assume $l^0(u)$ represents the initial label of node u. In each iteration t, the 1-WL algorithm constructs the multi-set label $\mathcal{L}^t_{\mathcal{N}}$ for each node u by aggregating the labels of its 1-hop neighborhood nodes $\mathcal{N}^1_u$, i.e., $\mathcal{L}^t_{\mathcal{N}}(u) = \{l^{t-1}(v) \mid v \in \mathcal{N}^1_u\}$. The 1-WL algorithm then merges the multi-set label $\mathcal{L}^t_{\mathcal{N}}$ with the current label of node u, and maps the sequence into a new label $l^t(u)$ through a Hash function as below

$$l^t(u) = \text{Hash}(l^{t-1}(u), \mathcal{L}^{t-1}_{\mathcal{N}}(u)). \tag{2}$$

The 1-WL test iterates until node labels stabilize or the maximum number of iterations is reached.

Message Passing Neural Network (MPNN). The MPNN framework generalizes the 1-WL test to continuous node and edge feature spaces by iteratively propagating and aggregating messages across a graph. It consists of two main components, i.e., a message passing phase and a readout phase. Let G be an undirected graph, and $\boldsymbol{h}^t_u$ denote the hidden state of node u at iteration t. In the message passing phase, the node representations are updated by aggregating the features of neighboring nodes and the associated edges. The node state at iteration $t+1$ is computed as

$$\boldsymbol{h}^{t+1}_u = \mathbf{U}^t(\boldsymbol{h}^t_u, \boldsymbol{m}^{t+1}_u), \quad \boldsymbol{m}^{t+1}_u = \textstyle\sum_{v \in \mathcal{N}^1_u} \mathbf{M}^t(\boldsymbol{h}^t_u, \boldsymbol{h}^t_v, \boldsymbol{e}_{uv}), \tag{3}$$

where $\mathbf{M}^t$ and $\mathbf{U}^t$ denote the message and update functions at iteration t, and $\boldsymbol{e}_{uv}$ represents the edge attribute for (u, v). After T iterations, a readout function aggregates the final node embeddings to form a global graph representation

$$\boldsymbol{y} = \mathbf{R}(\{\boldsymbol{h}^T_u \mid u \in G\}), \tag{4}$$

where $\mathbf{R}$ is a permutation-invariant readout function, e.g., sum, mean, or a learned pooling function.

As a specific instantiation of MPNNs, the GIN [53] defines the message and update function as

$$\boldsymbol{h}^{t+1}_u = \text{MLP}((1+\epsilon^t)\boldsymbol{h}^t_u + \boldsymbol{m}^{t+1}_u), \quad \boldsymbol{m}^{t+1}_u = \textstyle\sum_{v \in \mathcal{N}^1_u} \boldsymbol{h}^t_v, \tag{5}$$

where ϵ^t is a learnable parameter that controls the importance of the node's previous state.

Graph Transformers (GTs). While the MPNN framework relies on local neighborhood aggregation, the GT framework adopts a global self-attention mechanism, effectively treating the input graph as fully connected to capture long-range dependencies. Formally, let $\mathbf{H}^t = [\mathbf{h}^t_1, ..., \mathbf{h}^t_{|V|}]^T$ denote the node feature matrix at layer t. A standard GT layer updates the representation of node u by aggregating information from all nodes $v \in V$ as

$$\mathbf{h}^{t+1}_u = \sum_{v \in \mathcal{V}} \alpha_{uv}(\mathbf{W}_V \mathbf{h}^t_v), \tag{6}$$

where $\mathbf{W}_V \in \mathbb{R}^{d \times d}$ is the value projection matrix. The attention coefficients α_{uv} are computed via the scaled dot-product mechanism

$$\alpha_{uv} = \text{softmax}_{v \in \mathcal{V}}\left(\frac{(\mathbf{W}_Q \mathbf{h}^t_u)^\top (\mathbf{W}_K \mathbf{h}^t_v)}{\sqrt{d}}\right), \tag{7}$$

where $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{d \times d}$ are the query and key projection matrices respectively.

4 Structural Encoding via Contextual Distribution and Relational Similarity

In this section, we introduce a unified structural encoding framework designed to capture intrinsic topological properties. The framework is grounded in two core perspectives, i.e., the **Contextual Distribution** for absolute node descriptions and the **Relational Similarity** for relative structural affinities. We begin by formally defining the concept of contextual distribution and establishing the Contextual Distribution WL (ConD-WL) test. We theoretically prove that the ConD-WL test is strictly more expressive than the standard 1-WL test. Moreover, we demonstrate that this principle naturally extends from the WL test to neural message passing architectures. To instantiate the framework in a continuous and scalable manner, we introduce the Entropy Structural Encoding (EntSE). Building on the entropy representation, we introduce a Kernelized Relational Structural Encoding (KerSE). Finally, we demonstrate how EntSE and KerSE can be seamlessly integrated into modern backbones.

4.1 Theoretical Foundations: Contextual Distributions

We begin by formally defining the concept of contextual distribution as Definition 4.1.

Definition 4.1 (The k-hop Contextual Distribution)**.** Given a graph $G(V, E)$ and a center node $u \in V$, the k-hop context S_u^k is the subgraph induced by $V_{S_u^k} = \{v \in V \mid d(u, v) \leq k\}$, where $d(\cdot, \cdot)$ denotes shortest-path or random-walk distance. The contextual distribution is a mapping

$$\mathcal{D}_u^k := \Phi(S_u^k), \tag{8}$$

where Φ extracts a permutation-invariant summary of the local topology and node features within S_u^k. The codomain of Φ depends on the chosen instantiation. Representative instantiations in this work include the degree histogram used in WL-HistSE and a scalar summary of the spectral distribution used in EntSE. The detailed algorithm for extracting the k-hop context S_u^k is shown in the Appendix A.

The ConD-WL Test. We now formally introduce the ConD-WL test. The ConD-WL method aggregates not only the labels (as in the 1-WL test) but also the distribution of the k-hop contexts centered at each node. Specifically, the ConD-WL constructs multisets based on the contextual distributions rooted at neighboring nodes, and combines them with label information for joint hash encoding. Thus, in each iteration t, the label of node u is updated as

$$l^t(u) = \mathrm{Hash}(l^{t-1}(u), \{l^{t-1}(v) \mid v \in \mathcal{N}_u^1\}, \{\mathcal{D}_v^k \mid v \in \mathcal{N}_u^1\}). \tag{9}$$

We further conduct a theoretical analysis of the expressive power of ConD-WL.

Proposition 4.2. *There exists an instantiation of Φ under which ConD-WL is strictly more expressive than 1-WL. That is, (i) for any instantiation of Φ, any pair of non-isomorphic graphs distinguishable by 1-WL is also distinguishable by ConD-WL; and (ii) there exists an instantiation of Φ and a pair of non-isomorphic graphs that are indistinguishable by 1-WL but distinguishable by ConD-WL.*

The proof is shown in the Appendix B. To illustrate the effectiveness of ConD-WL, we present a

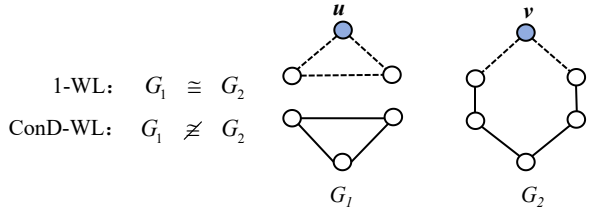


Figure 1: A top example of the ConD-WL algorithm, where $\cong$ denotes graph isomorphism.

toy example in Figure 1. To seamlessly integrate with the discrete aggregation mechanism of the standard WL test, we instantiate the contextual distribution as a degree multiset, i.e., $\mathcal{D}_v^k = \{\!\{\deg(j) \mid j \in V_{S_v^k}\}\!\}$. We refer to this instantiation as **WL-HistSE**. In Figure 1, the two graphs G_1 and G_2 are indistinguishable by the 1-WL test. In contrast, the WL-HistSE can capture fine-grained topological nuances. For example, the 1-hop context S_u^1 rooted at node u in G_1 yields a distribution multiset $\mathcal{D}_u^1 = \{\!\{2, 2, 2\}\!\}$, whereas the corresponding context S_v^1 in G_2 corresponds to $\mathcal{D}_v^1 = \{\!\{2, 1, 1\}\!\}$. These distinct multisets are subsequently mapped to different labels via a hashing function, allowing WL-HistSE to effectively distinguish between the two non-isomorphic graphs.

Neural Parameterization: ConD-MPNN. Based on the established correspondence between WL tests and MPNNs[53], we can derive the expressiveness of a neural architecture that parameterizes the ConD-WL mechanism. We term this general framework ConD-MPNN, a learnable, continuous version of ConD-WL that integrates contextual distribution encoding into the message-passing paradigm. Formally, the message passing phase in the ConD-MPNN can be defined as follows

$$\boldsymbol{m}_u^{t+1} = \textstyle\sum_{v \in \mathcal{N}_u^1} \mathbf{M}^t(\boldsymbol{h}_u^t, \boldsymbol{h}_v^t, \phi(\mathcal{D}_u^k), \phi(\mathcal{D}_v^k), \boldsymbol{e}_{uv}), \tag{10}$$

where ϕ denotes a learnable linear transformation applied to the contextual distribution.We now provide a theoretical analysis to characterize the expressive power of ConD-MPNN. It is known that MPNNs are at most as expressive as the 1-WL test [53]. Thus, the following Corollary 4.3 holds.

Corollary 4.3. *When MPNN is expressive as the 1-WL, the ConD-MPNN is strictly more powerful than both the 1&2-WL tests in distinguishing non-isomorphic graphs.*

4.2 Node-Level Instantiation: Entropy Structural Encoding (EntSE)

Building upon the theoretical foundation established above, we instantiate the contextual distribution (ConD) via Entropy Structural Encoding (EntSE) tailored for differentiable graph learning frameworks such as MPNNs and GTs. Unlike the discrete degree histogram used in WL-HistSE, which is typically represented as a high-dimensional multiset scaling with the graph size, the EntSE condenses this intricate structural information into a concise and continuous scalar representation. Formally, the EntSE instantiates the mapping function Φ using the Von Neumann entropy

$$\text{EntSE}(u) = \Phi(S_u^k) := \mathcal{H}_{\text{VN}}(S_u^k), \tag{11}$$

where $\mathcal{H}_{\text{VN}}(\cdot)$ is calculated as defined in Eq. (1). Having formally defined the encoding, a critical question arises: **Does this scalar compression retain sufficient discriminative power to effectively capture the topological variations of complex structures?** To investigate this, we first analyze the sensitivity of EntSE to structural perturbations using a sequence of non-isomorphic d-regular graphs generated via edge rewiring. These graphs are indistinguishable not only by the 1-WL and 2-WL tests, but also by the 3-WL test. As established in Theorem 4.4, the entropy converges to the mean of uniform random graphs, quantitatively tracking the structural transition towards randomness.

Theorem 4.4. *Let G_0 be a d-regular graph on n vertices, and let $\{G_t\}_{t=0}^{T}$ be a sequence of graphs generated by a random edge rewiring Markov chain. Specifically, there exists a mixing time τ_{mix} which is polynomial in n and d, such that for $T \geq \tau_{\text{mix}}$, the expected von Neumann entropy $\mathbb{E}[\mathcal{H}_{\text{VN}}(G_t)]$ converges to the average entropy of a uniform random d-regular graph.*

The detailed proofs and definitions of edge rewiring are provided in Appendix C. To intuitively visualize the expressive power of EntSE, we present a concrete case study in Figure 2 using the Shrikhande graph and the 4×4-Rook's graph. Specifically, we denote the 4×4 Rook's graph as G_1

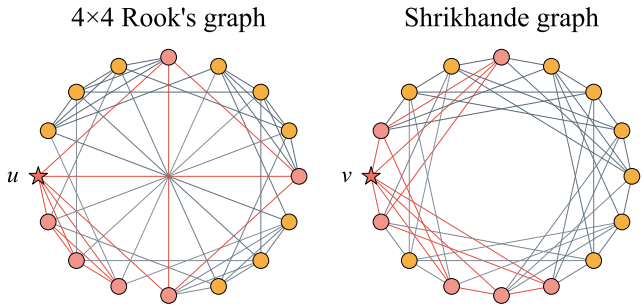


Figure 2: Notable examples of strongly regular graphs: 4×4 Rook's graph and Shrikhande graph.

and the Shrikhande graph as G_2. We select one node in each graph as the center node, marking them with a star. Focusing on the starred center nodes $u \in G_1$ and $v \in G_2$, we extract their 1-hop node contexts S_u^1 and S_v^1. The edges of the node contexts are highlighted by red edges in Figure 2. While these two strongly regular graphs are indistinguishable by the 3-WL test, the EntSE successfully differentiates them by yielding distinct values: $\mathcal{H}_{\text{VN}}(S_u^1) = 1.686$ and $\mathcal{H}_{\text{VN}}(S_v^1) = 1.698$, i.e., $\mathcal{H}_{\text{VN}}(S_u^1) \neq \mathcal{H}_{\text{VN}}(S_v^1)$. A detailed discussion analyzing the failure of degree-based metrics on these instances is provided in **Appendix D**.

Remark: Unlike LapPE [27], which encodes global spectral coordinates and requires invariant architectures for permutation invariance, EntSE is invariant by construction. We further prove in **Appendix E** that the two encodings are formally incomparable in distinguishing power.

4.3 Pair-Level Extension: Kernelized Relational Similarity (KerSE)

While EntSE provides a rich set of local descriptors, it remains inherently an absolute characterization of individual node identity. To capture the underlying structural resonance between nodes, it is imperative to introduce a mechanism that quantifies neighborhood consistency from a relational perspective. Departing from conventional paradigms that rely on coordinate-based positional encodings, e.g., the LapPE [27], our method focuses on the relational similarity of topological distributions. We instantiate this principle through KerSE, a structural encoding framework that redefines node relationships through distributional alignment rather than spatial proximity, thereby offering a more robust and permutation-invariant representation of local graph topology. Formally, given the contextual entropies

$\mathcal{H}_{\mathrm{VN}}(S_u^k)$ and $\mathcal{H}_{\mathrm{VN}}(S_v^k)$ for a node pair (u, v), the KerSE quantify their relational similarity the Reproducing kernel proposed in [54] as follows

$$\mathrm{KerSE}(u, v) = \mathcal{K}(S_u^k, S_v^k) = \frac{1}{2} e^{-|\mathcal{H}_{VN}(S_u^k) - \mathcal{H}_{VN}(S_v^k)|}. \tag{12}$$

Crucially, the basic Reproducing kernel admits an explicit feature mapping into a high-dimensional Hilbert space. This ensures that the KerSE is not a mere heuristic but a principled inner product in a feature space that captures high-order structural resonance. We also provide a formal derivation of this explicit mapping in the Appendix F.

In contrast to coordinate-based embeddings that may exhibit high sensitivity to graph perturbations, the KerSE remains robust against localized structural noise or fluctuations in contextual entropy. To rigorously quantify this effect, we establish the following theorem regarding the Lipschitz stability of KerSE. Detailed proofs are provided in **Appendix G**.

Theorem 4.5. ***(Stability of KerSE under graph perturbations)****. Let $\hat{S}_u^k, \hat{S}_v^k$ be obtained from S_u^k, S_v^k by perturbing at most m_u, m_v edges respectively. Then $|\mathrm{KerSE}(u, v) - \mathrm{KerSE}(\hat{u}, \hat{v})| \leq \frac{1}{2}(\varepsilon(m_u, n) + \varepsilon(m_v, n))$, where $\varepsilon(m, n) = T_m \log(n-1) + h(T_m)$, $T_m = \frac{m}{|E|-m}$, and $h(x) = -x \log x - (1-x)\log(1-x)$ is the binary entropy function.*

4.4 Integration with Graph Neural Architectures

To bridge the theoretical framework with practical graph learning, we present how EntSE and KerSE can be seamlessly integrated into modern neural backbones.

Integration into MPNNs and GTs. Both encodings are incorporated through a unified injection template. EntSE augments each node with an absolute, permutation-invariant structural descriptor, providing an alternative to spectral positional encodings such as LapPE [27]. In contrast, KerSE serves as a pairwise relational bias that quantifies contextual affinity between nodes.

For MPNNs, the EntSE replaces the contextual distribution $\mathcal{D}_u^k$ in Eq. 10 via $\phi(\mathcal{H}_{\mathrm{VN}}(S_u^k))$. For the KerSE invariant, node representations are updated via kernel-based aggregation over entropy-encoded contexts. We refer to this framework as the Kernel Aggregation MPNN (KerA-MPNN), defined as

$$\boldsymbol{m}_u^{t+1} = \sum\nolimits_{v \in \mathcal{N}_u^1} \mathbf{M}^t(\boldsymbol{h}_u^t, \boldsymbol{h}_v^t, \mathcal{K}(S_u^k, S_v^k), \boldsymbol{e}_{uv}). \tag{13}$$

For Graph Transformers, the same template specializes to a node-level concatenation of EntSE as

$$\mathbf{h}_u^{t+1} = \mathrm{CONCAT}(\mathbf{h}_u^t, \phi(\mathcal{H}_{\mathrm{VN}}(S_u^k))). \tag{14}$$

In contrast to the node-level augmentation, the KerSE affinities are integrated into the self-attention mechanism as a relational bias via a learnable mapping $\psi(\cdot)$. The attention score is defined as

$$\alpha_{uv} = \mathrm{softmax}_{v \in \mathcal{V}} \left(\frac{(\mathbf{W}_Q \mathbf{h}_u^t)^\top (\mathbf{W}_K \mathbf{h}_v^t)}{\sqrt{d}} + \psi(\mathrm{KerSE}(u, v)) \right). \tag{15}$$

Efficient Realization. To improve message-passing efficiency, we introduce two optimization strategies. First, since entropy depends only on the isomorphism class of a context, we cache entropy values of structurally identical subgraphs and reuse them across nodes, further reducing redundant computation. Second, as entropy reflects structural complexity [41], contexts with richer structural information exhibit higher entropy. Therefore, we normalize the entropy values of the node contexts into a probability distribution and sample representative neighbors with higher entropy. We will empirically validate these in Appendix J.11. The message passing mechanism is updated as follows

$$\boldsymbol{m}_u^{t+1} = \sum\nolimits_{v \in \mathcal{N}_u^1} \mathbf{M}^t(\boldsymbol{h}_u^t, \boldsymbol{h}_v^t, \boldsymbol{e}_{uv}) + \phi\{\mathcal{K}(S_u^k, S_v^k) | v \sim \mathrm{Multinomial}(\mathbf{p}, m)\}, \tag{16}$$

where $v \sim \mathrm{Multinomial}(\mathbf{p}, m)$ denotes that m neighbors are sampled from a multinomial distribution $\mathbf{p}$ defined over the normalized context entropy values.

The Complexity Analysis. Our framework incurs a one-time offline preprocessing cost for extracting local k-hop contexts and computing entropy-based structural encodings, while the online training stage introduces only negligible overhead and preserves the asymptotic time and space complexity of the backbone architecture. A detailed complexity analysis is provided in **Appendix H**.

In addition to the standard MPNN, we describe three representative backbone architectures and present how our methods can be incorporated into these backbones in **Appendix I**. Specifically, K-hop MPNNs [40] are designed to capture higher-order neighborhood information, PPGN [32] targets enhanced expressive power, and GraphGPS [45] bridges GNNs with GTs.

5 Experiments

In this section, we empirically evaluate how the EntSE and KerSE enhance expressive power and improve performance on graph classification and graph regression tasks when integrated into various architectures. To better bridge theory and practice, we aim to answer the following key questions through our experiments: **Q1:** Can we empirically verify that the EntSE-based structural encoding enhances expressive power beyond the 1-WL test, enabling models to distinguish graph pairs that are indistinguishable by the 3-WL test? **Q2:** Can EntSE and KerSE deliver superior performance on real-world graph classification and regression benchmarks, and consistently outperform existing positional and structural encodings across different backbone architectures? **Q3:** Can empirical results demonstrate that the proposed method scales to larger graphs and achieves favorable computational efficiency, in terms of both runtime and memory, compared to subgraph-based GNNs? Detailed descriptions of datasets and the experimental settings are reported in the Appendix J.1 and Appendix J.2. The code is available in https://github.com/SophiaQian02/Contextual-Distribution.

Table 1: Performance comparison on the BREC benchmark across different backbones and encodings. The best and second-best results within each backbone are marked in bold and underlined.

Method	Basic (60)	Regular (140)	Extension (100)	CFI (100)	Total (400)	Acc (%)
GNN-AK+	60	50	97	15	222	55.5
GRIT	60	50	100	16	226	56.5
DSS-GNN	58	48	100	15	221	55.2
PPGN	60	50	100	23	233	58.2
KP-GIN	60	106	98	11	275	68.8
I^2-GNN	60	100	100	21	281	70.2
N^2GNN	60	100	100	27	287	71.8
PPGN+RWSE	60	50	100	16	226	56.5
PPGN+LapSE	57	46	97	3	203	50.8
PPGN+EntSE	60	121	100	20	**301**	**75.3**
PPGN+KerSE	60	119	100	19	<u>298</u>	<u>74.5</u>
K-GIN+RWSE	60	94	100	15	269	67.3
K-GIN+LapSE	60	100	100	14	274	68.5
K-GIN+EntSE	60	114	100	16	**290**	**72.5**
K-GIN+KerSE	60	116	100	5	<u>281</u>	<u>70.3</u>
GPS+RWSE	60	48	100	3	211	52.8
GPS+LapSE	57	50	96	4	207	51.8
GPS+EntSE	60	114	100	4	<u>278</u>	<u>69.5</u>
GPS+KerSE	60	116	100	3	**279**	**69.8**

5.1 Evaluation on Expressive Power

To answer **Q1**, we evaluate the expressive power of EntSE and KerSE using the K-GIN [40], PPGN [32], and GraphGPS [45] backbones. On the BREC benchmark, which is specifically designed to evaluate expressive power beyond standard WL tests, the methods that PPGN augmented with EntSE or KerSE achieve state-of-the-art performance. Furthermore, the results demonstrate that compared to existing encoding schemes, EntSE and KerSE substantially enhance the expressive power of GNNs in distinguishing challenging non-isomorphic graphs. In addition, we evaluate EntSE and KerSE on the EXP, CSL, and SR datasets using the GIN [53] and K-GIN [40] backbones. Due to space limitations, detailed experimental results on these datasets are reported in **Appendix J.3**.

5.2 Evaluation on Graph Classification and Regression

Results on Graph Regression. To answer **Q2**, we first evaluate the regression performance on the **ZINC** dataset across three representative backbones, as summarized in Table 2. The results show that both EntSE and KerSE can be seamlessly incorporated into both MPNNs and GTs, providing informative structural priors that enhance molecular property prediction. Moreover, both methods are consistently competitive with conventional positional and structural encodings, e.g., RWSE and LapSE, highlighting the benefits of structural representations beyond coordinate-based embeddings.

We further evaluate our framework on the **QM9** dataset in Table 3. On QM9, we adopt K-GIN as the backbone due to its lightweight architecture and favorable efficiency. We compare our method with the baselines, e.g., the DTNN [52], PPGN [32], NGNN [60], I^2-GNN [24], SEK-GIN [55],

Table 2: Performance comparison on the ZINC dataset. Results are reported as MAE (↓).

Encoding Method	PPGN	K-GIN	GraphGPS
none	0.0846 ± 0.0117	0.0970 ± 0.0040	0.1182 ± 0.0049
RWSE	0.0695 ± 0.0016	0.1213 ± 0.0188	0.0700 ± 0.0040
LapPE	0.0726 ± 0.0065	0.1593 ± 0.0251	0.1078 ± 0.0084
EntSE	0.0789 ± 0.0105	$\mathbf{0.0860 \pm 0.0070}$	$\mathbf{0.0667 \pm 0.0030}$
KerSE	$\mathbf{0.0662 \pm 0.0029}$	0.1010 ± 0.0030	0.0671 ± 0.0021

KP-GIN [19], standard MPNN and K-hop MPNN. Notably, the K-GIN+EntSE achieves the best results on 7 out of 12 targets, including α, ϵ_{HOMO}, ϵ_{LUMO}, $\Delta\epsilon$, $\langle R^2 \rangle$, U_0, U, H, and G, indicating that our entropy-based context representations generalize well to complex quantum property prediction tasks. The model with KerSE also achieves the best performance on 4 targets, further validating the effectiveness of the proposed kernel-based strategy.

Table 3: Evaluation on the QM9 dataset (MAE). Gray shading highlights results in the last two columns that outperform all preceding baselines. Lower values are better.

Target	DTNN	MPNN	PPGN	NGNN	I^2-GNN	SEK-GIN	K-GIN	KP-GIN	K-GIN+EntSE	K-GIN+KerSE
μ	0.244	0.358	**0.231**	0.433	0.428	0.358	0.430	0.358	0.360	0.365
α	0.95	0.89	0.382	0.265	0.230	0.228	0.310	0.233	**0.224**	0.233
ϵ_{HOMO}	0.00388	0.00541	0.00276	0.00279	0.00261	**0.00106**	0.00301	0.00240	0.00237	0.00245
ϵ_{LUMO}	0.00512	0.00623	0.00287	0.00276	0.00267	0.00229	0.00288	0.00236	**0.00228**	**0.00228**
$\Delta\epsilon$	0.0112	0.0066	0.00406	0.00390	0.00380	0.00335	0.00416	0.00333	**0.00330**	0.00346
$\langle R^2 \rangle$	17.0	28.5	16.7	20.1	18.64	16.91	23.37	16.51	**16.26**	16.29
ZPVE	0.00172	0.00216	0.00064	0.00015	0.00014	**0.00013**	0.00033	0.00017	0.00018	0.00017
U_0	2.43	2.05	0.234	0.205	0.211	**0.0587**	0.0602	0.0682	0.0667	0.0811
U	2.43	2.00	0.234	0.200	0.206	0.0672	0.0504	0.0696	**0.0323**	0.0583
H	2.43	2.02	0.229	0.249	0.269	0.073	0.0534	0.0641	0.0344	**0.0268**
G	2.43	2.02	0.238	0.253	0.261	0.0592	0.0533	0.0484	**0.0309**	0.0389
C_ν	0.27	0.42	0.184	0.0811	**0.0730**	0.0924	0.1365	0.0869	0.0916	0.0899

Results on Graph Classification. We first conduct graph classification experiments on the TU datasets, as reported in Table 4. To align with our theoretical analysis in Section 4.1, we evaluate the WL framework equipped with HistSE, i.e., WL+HistSE. As shown in Table 4, the HistSE consistently improves over the original WL kernel across almost all datasets, providing empirical evidence that contextual distributional encoding can effectively represent structural information, thereby improving graph classification performance. Second, to demonstrate the practical effectiveness of our framework beyond the WL setting, we integrate EntSE and KerSE into representative neural backbones, including GIN [53], K-GIN [40], and other backbones in the Appendix J.4. We further compare against strong expressive baselines, e.g., subgraph-based GNNs [60, 61]. The results in Table 4 demonstrate that our models built on the K-GIN backbone outperform several expressive GNNs. In addition, we evaluate our method on the **OGB-molhiv** and **OGB-molpcba** datasets in Appendix J.5. We use GraphGPS [45] as the backbone and compare our approach with other encoding methods.

Table 4: Evaluation on TuDatasets. The best and second-best results within each backbone are marked in bold and underlined. Gray shading highlights results outperform the backbone baseline. "OOM" denotes out-of-memory.

Method	MUTAG	PTC	PROTEINS	IMDB-BINARY	IMDB-MULTI	DD	COLLAB
WL	82.9 ± 0.6	56.8 ± 0.6	73.5 ± 0.4	71.9 ± 0.8	49.5 ± 0.5	78.6 ± 0.2	77.4 ± 0.4
WL+HistSE	84.8 ± 0.6	57.9 ± 0.5	75.4 ± 0.4	72.7 ± 0.2	50.6 ± 0.2	78.4 ± 0.2	77.8 ± 0.1
GIN	89.4 ± 5.6	64.6 ± 7.0	75.9 ± 2.8	75.1 ± 5.1	52.3 ± 2.8	75.3 ± 2.9	80.2 ± 1.9
GIN+EntSE	89.9 ± 5.0	66.0 ± 6.5	76.0 ± 3.9	76.3 ± 4.3	53.2 ± 3.4	78.9 ± 3.5	80.5 ± 1.7
GIN+KerSE	91.5 ± 5.4	66.9 ± 7.5	76.4 ± 4.4	75.3 ± 2.7	53.1 ± 3.6	79.4 ± 4.4	80.6 ± 2.1
IGN	83.9 ± 13.0	58.5 ± 6.9	76.6 ± 5.5	72.0 ± 5.5	48.7 ± 3.4	N/A	78.3 ± 2.5
GNN-AK+	91.3 ± 7.0	$\mathbf{67.7 \pm 8.8}$	76.5 ± 2.5	75.6 ± 3.7	50.4 ± 3.2	OOM	OOM
Nestd GNN	87.9 ± 8.2	54.1 ± 7.7	73.9 ± 5.1	71.4 ± 5.9	N/A	77.8 ± 3.9	N/A
GSN	92.2 ± 7.5	67.4 ± 5.7	74.6 ± 5.0	76.8 ± 2.0	52.6 ± 3.6	N/A	82.7 ± 1.5
KP-GIN	92.2 ± 6.5	66.8 ± 6.8	75.8 ± 4.6	76.6 ± 4.2	52.3 ± 4.1	80.0 ± 5.3	N/A
K-GIN	88.9 ± 8.3	63.8 ± 8.8	76.0 ± 5.0	73.0 ± 3.5	52.7 ± 3.7	79.5 ± 6.6	80.8 ± 2.1
K-GIN+EntSE	$\mathbf{93.9 \pm 5.5}$	67.4 ± 9.4	$\mathbf{77.0 \pm 3.1}$	$\mathbf{76.8 \pm 1.9}$	$\mathbf{53.3 \pm 2.5}$	79.7 ± 3.0	$\mathbf{82.7 \pm 1.2}$
K-GIN+KerSE	$\mathbf{92.8 \pm 4.6}$	67.6 ± 6.4	$\mathbf{76.7 \pm 6.5}$	76.6 ± 4.1	$\mathbf{53.1 \pm 3.2}$	$\mathbf{80.4 \pm 4.9}$	82.4 ± 2.1

5.3 Computational Efficiency and Scalability

Comparisons of Runtime and Memory Cost. To answer **Q3**, we evaluate the end-to-end runtime including the preprocessing time, and the memory usage in Table 5. We select three datasets for

the evaluations, including the dense COLLAB dataset, the large-scale DD dataset, and the sparse PROTEINS dataset. The detailed statistics are shown in Table 13. The results show that our method outperforms KP-GIN [19] in runtime and demonstrates better memory efficiency than GNN-AK+ [61] and ESAN [6]. A detailed breakdown of preprocessing and training time in **Appendix J.6**.

Table 5: Efficiency comparison on three datasets. "OOM" denotes out-of-memory. The best and second-best results in each column are marked in bold and underlined, respectively.

Model	DD		COLLAB		PROTEINS	
	Time (s)	Memory	Time (s)	Memory	Time (s)	Memory
ESAN	OOM	OOM	OOM	OOM	**1645**	2170M
GNN-AK+	OOM	OOM	OOM	OOM	<u>1821</u>	5468M
KP-GIN	19096	**1116M**	$\gg$ 1 day	–	1960	476M
K-GIN+EntSE	<u>6830</u>	<u>1544M</u>	<u>20039</u>	<u>2620M</u>	1924	<u>474M</u>
K-GIN+KerSE	**6619**	1808M	**19214**	**2208M**	2019	**416M**

Effect of Subgraph Entropy Reuse. As discussed in Section 4.4, we improve computational efficiency by reusing entropy values across structurally identical ego-subgraphs. Specifically, we compute the entropy only once for each distinct ego-subgraph structure and cache the result for subsequent reuse, avoiding redundant computation. We evaluate the effect of this strategy through an ablation study. As shown in Table 6, this strategy substantially reduces preprocessing time, with the largest reduction observed on COLLAB. This suggests that entropy reuse is particularly effective when local structures recur frequently within a dataset.

Table 6: Preprocessing time comparison with entropy reuse.

Model	PROTEINS	DD	COLLAB
K-GIN+EntSE	29	605	8846
K-GIN+EntSE(reuse)	**14**	**298**	**826**
K-GIN+KerSE	31	614	7673
K-GIN+KerSE(reuse)	**18**	**329**	**854**

Scalability on Large-scale Graphs. To assess performance on larger graphs, we compare different methods on the Peptides-Func dataset. As shown in Table 7, K-GIN+EntSE achieves the highest average precision, outperforming GIN [53], K-GIN [40], GIN+CSE and HyMN [47], and GT+MoSE [5]. These results demonstrate the effectiveness of K-GIN+EntSE in this setting.

Table 7: Performance on the Peptides dataset.

Method	Peptides (Test AP $\uparrow$)
GIN	0.6555 ± 0.0088
K-GIN	0.6479 ± 0.0041
GIN + CSE	0.6619 ± 0.0077
HyMN	0.6857 ± 0.0055
GT+MoSE	0.6350 ± 0.0110
K-GIN+EntSE(ours)	$\mathbf{0.6901 \pm 0.0034}$
K-GIN+KerSE(ours)	$\mathbf{0.6948 \pm 0.0063}$

Further experiments are presented in the Appendix, including: (i) a study on entropy collisions (Appendix J.7); (ii) results of stability (Appendix J.8); (iii) a study on different context size (Appendix J.9); (iv) a study on various context construction strategy (Appendix J.10); (v) a study on efficient realization (Appendix J.11); (vi) an ablation study on entropy choices (Appendix J.12); and (vii) a comparison of kernel functions (Appendix J.13).

6 Conclusion

In this work, we first introduce the concept of contextual distribution and develop two coordinate-free structural encodings to jointly characterize local structure and relational semantics in graphs. Both theoretical analyses and empirical results consistently demonstrate that EntSE can enhance the structural expressiveness of graph models, while KerSE can provide a stability guarantee. Moreover, both EntSE and KerSE can be seamlessly integrated into various backbones, and deliver stable and competitive improvements across multiple real-world graph classification and graph regression benchmarks, highlighting the generality and practical value of our approach.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 62676036, 62576371 and 62622615, and the Open Project Foundation of Key Laboratory of Computation Intelligence and Chinese Information Processing of Ministry of Education and Key Laboratory of Data Intelligence and Cognitive Computing of Shanxi Province.

References

[1] Ralph Abboud, İsmail İlkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization. In *Proceedings of IJCAI*, pages 2112–2118, 2021.

[2] Waïss Azizian and Marc Lelarge. Expressive power of invariant and equivariant graph neural networks. In *Proceedings od ICLR*, 2021.

[3] Lu Bai and Edwin R. Hancock. A fast jensen-shannon subgraph kernel. In Alfredo Petrosino, editor, *Image Analysis and Processing - ICIAP 2013 - 17th International Conference, Naples, Italy, September 9-13, 2013. Proceedings, Part I*, volume 8156 of *Lecture Notes in Computer Science*, pages 181–190. Springer, 2013.

[4] Muhammet Balcilar, Pierre Héroux, Benoit Gaüzère, Pascal Vasseur, Sébastien Adam, and Paul Honeine. Breaking the limits of message passing graph neural networks. In *Proceedings of ICML*, volume 139, pages 599–608. PMLR, 2021.

[5] Linus Bao, Emily Jin, Michael M. Bronstein, İsmail İlkan Ceylan, and Matthias Lanzinger. Homomorphism counts as structural encodings for graph learning. In *Proceedings of ICLR*, 2025.

[6] Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael M. Bronstein, and Haggai Maron. Equivariant subgraph aggregation networks. In *Proceedings of ICLR*, 2022.

[7] Beatrice Bevilacqua, Moshe Eliasof, Eli Meirom, Bruno Ribeiro, and Haggai Maron. Efficient subgraph gnns by learning effective selection policies. In *Proceedings of ICLR*, 2024.

[8] Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M. Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(1):657–668, 2023.

[9] Semih Cantürk, Renming Liu, Olivier Lapointe-Gagné, Vincent Létourneau, Guy Wolf, Dominique Beaini, and Ladislav Rampášek. Graph positional and structural encoder. *arXiv preprint arXiv:2307.07107*, 2023.

[10] Semih Cantürk, Renming Liu, Olivier Lapointe-Gagné, Vincent Létourneau, Guy Wolf, Dominique Beaini, and Ladislav Rampášek. Graph positional and structural encoder. In *Proceedings of ICML*, pages 5533–5566. PMLR, 2024.

[11] Jialin Chen, Haolan Zuo, Haoyu Peter Wang, Siqi Miao, Pan Li, and Rex Ying. Towards a universal graph structural encoder. *arXiv preprint arXiv:2504.10917*, 2025.

[12] Rongqin Chen, Yan Li, Dan Wu, Fan Mo, Shenghui Zhang, Pak Lon Ip, Hoi Cheong Iam, Ye Li, and Leong Hou U. Enhanced subgraph learning in 2-fwl gnns via local connectivity, spectral, and distance encodings. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 227–238, 2025.

[13] Colin Cooper, Martin Dyer, and Catherine Greenhill. Sampling regular graphs and a peer-to-peer network. *Combinatorics, Probability and Computing*, 16(4):557–593, 2007.

[14] George Dasoulas, Ludovic Dos Santos, Kevin Scaman, and Aladin Virmaux. Coloring graph neural networks for node disambiguation. In *Proceedings of IJCAI*, pages 2126–2132, 2020.

[15] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

[16] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. *Proceedings of NeurIPS*, 35: 22326–22340, 2022.

[17] Vijay Prakash Dwivedi, Chaitanya K. Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *J. Mach. Learn. Res.*, 24:43:1–43:48, 2023.

[18] Moshe Eliasof, Fabrizio Frasca, Beatrice Bevilacqua, Eran Treister, Gal Chechik, and Haggai Maron. Graph positional encoding via random feature propagation. In *International conference on machine learning*, pages 9202–9223. PMLR, 2023.

[19] Jiarui Feng, Yixin Chen, Fuhai Li, Anindya Sarkar, and Muhan Zhang. How powerful are k-hop message passing graph neural networks. In *Proceedings of NeurIPS*, 2022.

[20] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of ICML*, volume 70, pages 1263–1272, 2017.

[21] William L. Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *IEEE Data Eng. Bull.*, 40(3):52–74, 2017.

[22] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. Kernel methods in machine learning. 2008.

[23] Yinan Huang, William Lu, Joshua Robinson, Yu Yang, Muhan Zhang, Stefanie Jegelka, and Pan Li. On the stability of expressive positional encodings for graphs. *arXiv preprint arXiv:2310.02579*, 2023.

[24] Yinan Huang, Xingang Peng, Jianzhu Ma, and Muhan Zhang. Boosting the cycle counting power of graph neural networks with i^2-gnns. In *Proceedings of ICLR*, 2023.

[25] Yinan Huang, William Lu, Joshua Robinson, Yu Yang, Muhan Zhang, Stefanie Jegelka, and Pan Li. On the stability of expressive positional encodings for graphs. In *The Twelfth International Conference on Learning Representations*, 2024.

[26] Charilaos Kanatsoulis, Evelyn Choi, Stefanie Jegelka, Jure Leskovec, and Alejandro Ribeiro. Learning efficient positional encodings with graph neural networks. In *Proceedings of ICLR*, 2025.

[27] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Proceedings of NeurIPS*, 34:21618–21629, 2021.

[28] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[29] Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. Distance encoding: Design provably more powerful neural networks for graph representation learning. In *Proceedings of NeurIPS*, 2020.

[30] Derek Lim, Joshua David Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. In *The Eleventh International Conference on Learning Representations*, 2023.

[31] Liheng Ma, Chen Lin, Derek Lim, Adriana Romero-Soriano, Puneet K Dokania, Mark Coates, Philip Torr, and Ser-Nam Lim. Graph inductive biases in transformers without message passing. In *Proceedings of ICML*, pages 23321–23337. PMLR, 2023.

[32] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. In *Proceedings of NeurIPS*, pages 2153–2164, 2019.

[33] Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. In *Proceedings of ICLR*, 2019.

[34] Sohir Maskey, Ali Parviz, Hannes Stärk, Ylli Sadikaj, Haggai Maron, et al. Generalized laplacian positional encoding for graph representation learning. In *NeurIPS Workshop on Symmetry and Geometry in Neural Representations*, 2022.

[35] L. MIRSKY. Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics*, 11(1):50–59, 01 1960. ISSN 0033-5606.

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

[37] 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.

[38] Christopher Morris, Gaurav Rattan, Sandra Kiefer, and Siamak Ravanbakhsh. Speqnets: Sparsity-aware permutation-equivariant graph networks. In *Proceedings of ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 16017–16042. PMLR, 2022.

[39] Ryan L. Murphy, Balasubramaniam Srinivasan, Vinayak A. Rao, and Bruno Ribeiro. Relational pooling for graph representations. In *Proceedings of ICML*, volume 97, pages 4663–4673, 2019.

[40] Giannis Nikolentzos, George Dasoulas, and Michalis Vazirgiannis. k-hop graph neural networks. *Neural Networks*, 130:195–205, 2020.

[41] Filippo Passerini and Simone Severini. The von neumann entropy of networks. *arXiv preprint arXiv:0812.2597*, 2008.

[42] Omri Puny, Derek Lim, Bobak Kiani, Haggai Maron, and Yaron Lipman. Equivariant polynomials for graph neural networks. In *Proceedings of ICML*, pages 28191–28222. PMLR, 2023.

[43] Chendi Qian, Gaurav Rattan, Floris Geerts, Mathias Niepert, and Christopher Morris. Ordered subgraph aggregation networks. In *Proceedings of NeurIPS*, volume 35, pages 21030–21045, 2022.

[44] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

[45] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Proceedings of NeurIPS*, 35:14501–14515, 2022.

[46] 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.

[47] Joshua Southern, Yam Eitan, Guy Bar-Shalom, Michael M Bronstein, Haggai Maron, and Fabrizio Frasca. Balancing efficiency and expressiveness: Subgraph gnns with walk-based centrality. In *Proceedings of ICML*, 2025.

[48] Behrooz Tahmasebi, Derek Lim, and Stefanie Jegelka. Counting substructures with higher-order graph neural networks: Possibility and impossibility results. *arXiv preprint arXiv:2012.03174*, 2020.

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Proceedings of NeurIPS*, 30, 2017.

[50] Haorui Wang, Haoteng Yin, Muhan Zhang, and Pan Li. Equivariant and stable positional encoding for more powerful graph neural networks. In *International Conference on Learning Representations*, 2022.

[51] Yanbo Wang and Muhan Zhang. An empirical study of realized gnn expressiveness. In *Proceedings of ICML*, pages 52134–52155, 2024.

[52] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[53] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *Proceedings of ICLR, 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[54] Lixiang Xu, Xiaoyi Jiang, Lu Bai, Jin Xiao, and Bin Luo. A hybrid reproducing graph kernel based on information entropy. *Pattern Recognit.*, 73:89–98, 2018.

[55] Tianjun Yao, Yingxu Wang, Kun Zhang, and Shangsong Liang. Improving the expressiveness of k-hop message-passing gnns by injecting contextualized substructure information. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 3070–3081. ACM, 2023.

[56] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Proceedings of NeurIPS*, 34:28877–28888, 2021.

[57] Jiaxuan You, Rex Ying, and Jure Leskovec. Position-aware graph neural networks. In *International conference on machine learning*, pages 7134–7143. PMLR, 2019.

[58] Jiaxuan You, Jonathan Michael Gomes Selman, Rex Ying, and Jure Leskovec. Identity-aware graph neural networks. In *Proceedings of AAAI*, pages 10737–10745, 2021.

[59] Bohang Zhang, Shengjie Luo, Liwei Wang, and Di He. Rethinking the expressive power of gnns via graph biconnectivity. In *Proceedings of ICLR*, 2023.

[60] Muhan Zhang and Pan Li. Nested graph neural networks. In *Proceedings of NeurIPS*, pages 15734–15747, 2021.

[61] Lingxiao Zhao, Wei Jin, Leman Akoglu, and Neil Shah. From stars to subgraphs: Uplifting any GNN with local structure awareness. In *Proceedings of ICLR*, 2022.

Algorithm 1 Algorithm for extracting node context based on $(1 \sim k)$-hop neighborhood

Input: A graph G, adjacency matrix A, number of hops k, number of nodes $|V|$, method type (random walk or shortest path)
Output: The node context based on the $(1 \sim k)$-hop neighborhood, the edge attributes for k-hop MPNN (the list of adjacency matrices A_{list}).

Construct an empty adjacency matrix A_{final} and a list A_{list} containing k adjacency matrices, where the i-th element in the list corresponds to the adjacency matrix for the i-hop neighbors.
for $i \leftarrow 1$ to k **do**
  Compute the i-th power of the adjacency matrix $A^{(i)} = A^i$
  if method == "random walk" **then**
    Update $A_{final} = A_{final} + A^{(i)}$
    Update $A_{list}[i] = A^{(i)}$
  else if method == "shortest path" **then**
    Mask nodes already existing in previous hops $A^{(i)}[A_{final} > 0] = 0$
    Update $A_{final} = A_{final} + A^{(i)}$
    Update $A_{list}[i] = A^{(i)}$
  end if
  for each node $u \in G$ **do**
    Extract neighbors $\mathcal{N}_u^{1:i}$ such that $A_{final}[u, v] > 0$ for $v \in \mathcal{N}_u^{1:i}$
    Update node context $S_u^i = A_{final}[\mathcal{N}_u^{1:i}][:, \mathcal{N}_u^{1:i}]$
  end for
end for
return the $(1-k)$-hop node context $S_u^{1:k}$, the A_{list}

A Algorithm for Extracting the Node Context

We present the procedure for extracting the k-hop node context in Algorithm 1. For each node u, we first construct its k-hop neighborhood based on shortest-path or random walk distances. We then extract the corresponding induced k-hop node context, based on the associated node and edge sets.

B Proof of Proposition 4.2

The ConD-WL enhances the standard WL test by introducing a hash function over contextual distribution sequences. This design induces a more injective mapping that theoretically enhances expressive power, facilitating the distinction of complex non-isomorphic substructures often conflated by standard aggregation methods.

We first prove **the first part** of the proposition by contradiction. We begin by assuming, for the sake of contradiction, that there exist two non-isomorphic graphs G_1 and G_2 which can be distinguished by the 1-WL test, but are incorrectly identified as isomorphic by ConD-WL. Specifically, at the final iteration T, there exist two nodes $u_1 \in G_1$ and $u_2 \in G_2$ with similar but non-isomorphic local topological structures, yet the ConD-WL assigns them the same label

$$l^T(u_1) = l^T(u_2), \tag{17}$$

and their labels are computed through the following iterative update process

$$l^T(u) = \text{Hash}(l^{T-1}(u), \{l^{T-1}(v) \mid v \in \mathcal{N}_u^1\}, \{\mathcal{D}_v^k \mid v \in \mathcal{N}_u^1\}), \tag{18}$$

Moreover, since the hash function is injective, all three components must be identical as well, i.e.,

$$\begin{aligned} l^{T-1}(u_1) &= l^{T-1}(u_2), \\ \text{Hash}\{l^{T-1}(v_1) \mid v_1 \in \mathcal{N}_{u_1}^1\} &= \text{Hash}\{l^{T-1}(v_2) \mid v_2 \in \mathcal{N}_{u_2}^1\}, \\ \text{Hash}\{\mathcal{D}_{v_1}^k \mid v_1 \in \mathcal{N}_{u_1}^1\} &= \text{Hash}\{\mathcal{D}_{v_2}^k \mid v_2 \in \mathcal{N}_{u_2}^1\}. \end{aligned} \tag{19}$$

However, by assumption, the two non-isomorphic graphs can be distinguished by the 1-WL test, which implies that

$$\text{Hash}(l^{T-1}(u_1), \{l^{T-1}(v_1) \mid v_1 \in \mathcal{N}_{u_1}^1\}) \neq \text{Hash}(l^{T-1}(u_2), \{l^{T-1}(v_2) \mid v_2 \in \mathcal{N}_{u_2}^1\}) \tag{20}$$

for corresponding nodes u in G_1 and u_2 in G_2. This contradicts the earlier conclusion that $l^T(u_1) = l^T(u_2)$ under the ConD-WL. Hence, the first part of the proposition is proven, as any pair of non-isomorphic graphs that are distinguishable by the 1-WL test can also be distinguished by ConD-WL.

Now we prove **the second part** of the proposition, namely that there exists an instantiation of Φ and a pair of non-isomorphic graphs that are indistinguishable by 1-WL but distinguishable by ConD-WL. We establish this by an explicit construction.

We assume that there exist two nodes $u_1 \in G_1$ and $u_2 \in G_2$ at the $T-1$ iteration with the same label, and the labels of all their neighbors are also the same, i.e.,

$$l^{T-1}(u_1) = l^{T-1}(u_2), \quad l^{T-1}(v_1) = l^{T-1}(v_2), \quad \forall v_1, v_2 \in \mathcal{N}^1_{u_1}, \mathcal{N}^1_{u_2}. \tag{21}$$

Let $\Phi := \Phi_{\text{hist}}$ be the degree-histogram descriptor (WL-HistSE in Section 4.1), defined by $\mathcal{D}^k_v = \{\!\{\deg(j) \mid j \in V_{S^k_v}\}\!\}$. Consider the non-isomorphic graphs G_1 and G_2 shown in Figure 1, which are indistinguishable by the 1-WL test. As discussed in Section 4.1, the 1-hop degree multisets at the marked center nodes $u_1 \in G_1$ and $u_2 \in G_2$ are $\mathcal{D}^1_{u_1} = \{\!\{2,2,2\}\!\}$ and $\mathcal{D}^1_{u_2} = \{\!\{2,1,1\}\!\}$, respectively, which differ as multisets. Aggregating over the 1-hop neighborhoods, this implies

$$\text{Hash}(\{\mathcal{D}^k_{v_1} \mid v_1 \in \mathcal{N}^1_{u_1}\}) \neq \text{Hash}(\{\mathcal{D}^k_{v_2} \mid v_2 \in \mathcal{N}^1_{u_2}\}). \tag{22}$$

It can be directly derived that

$$\begin{aligned} &\text{Hash}(l^{T-1}(u_1), \{l^{T-1}(v_1) \mid v_1 \in \mathcal{N}^1_{u_1}\}, \{\mathcal{D}^k_{v_1} \mid v_1 \in \mathcal{N}^1_{u_1}\}) \\ &\quad \neq \\ &\text{Hash}(l^{T-1}(u_2), \{l^{T-1}(v_2) \mid v_2 \in \mathcal{N}^1_{u_2}\}, \{\mathcal{D}^k_{v_2} \mid v_2 \in \mathcal{N}^1_{u_2}\}), \end{aligned} \tag{23}$$

so $l^T(u_1) \neq l^T(u_2)$ under ConD-WL, while 1-WL assigns them the same label since

$$\text{Hash}(l^{T-1}(u_1), \{l^{T-1}(v_1) \mid v_1 \in \mathcal{N}^1_{u_1}\}) = \text{Hash}(l^{T-1}(u_2), \{l^{T-1}(v_2) \mid v_2 \in \mathcal{N}^1_{u_2}\}). \tag{24}$$

Hence, the ConD-WL with Φ_{hist} distinguishes G_1 and G_2, while 1-WL does not. This proves part (ii), and combined with part (i), establishes that ConD-WL is strictly more expressive than 1-WL. Therefore, Proposition 4.2 holds.

C Proof of Theorem 4.4

Proof. Due to the high symmetry of d-regular graphs, many non-isomorphic instances are indistinguishable by the 1&2-WL, and even 3-WL tests. To construct a family of such WL-equivalent yet structurally diverse graphs, we employ *edge rewiring*, a local operation that preserves degree regularity while altering higher-order connectivity.

Definition C.1. (The Edge Rewiring). Let $G_1 = (V_1, E_1)$ be a simple undirected graph, we can construct $G_2 = (V_2, E_2)$ by selecting two disjoint edges $(u, v), (x, y) \in E_1$ and replacing them with (u, y) and (x, v), provided that the new edges do not introduce self-loops or parallel edges. Formally, $V_1 = V_2$ and $E_2 = (E_1 \setminus \{(u, v), (x, y)\}) \cup \{(u, y), (x, v)\}$.

The iterative application of this random edge rewiring operation induces a reversible Markov chain $\mathcal{M}$, formally known as the edge-switching Markov chain [13]. This chain operates over the state space $\Omega_{n,d}$ consisting of all d-regular graphs on n vertices. It is well-established that $\mathcal{M}$ is irreducible and aperiodic, ensuring convergence to a unique stationary distribution π_n that is uniform over $\Omega_{n,d}$. Crucially, the convergence rate of this chain has been rigorously established:

Theorem C.2 (Polynomial Mixing Time [13])**.** *The edge-switching Markov chain is rapidly mixing. Formally, the mixing time $\tau_{\text{mix}}(\varepsilon)$ is defined as the smallest time t such that the total variation distance to stationarity is at most ε, i.e., $\tau_{\text{mix}}(\varepsilon) = \min\{t : d_{\text{TV}}(P_t, \pi_n) \leq \varepsilon\}$. This mixing time is bounded by a polynomial in n and d*

$$\tau_{\text{mix}}(\varepsilon) \leq \text{poly}(n, d) \cdot \log(\varepsilon^{-1}). \tag{25}$$

By definition, for any $T \geq \tau_{\text{mix}}(\varepsilon)$, the distribution P_T of the generated graph G_T satisfies $d_{\text{TV}}(P_T, \pi_n) \leq \varepsilon$. We now relate this probabilistic convergence to the convergence of the expected

von Neumann entropy. The entropy $\mathcal{H}_{\mathrm{VN}}(G)$ is a bounded function, specifically $0 \leq \mathcal{H}_{\mathrm{VN}}(G) \leq \ln n$. We utilize the variational characterization of total variation distance, which bounds the difference in expectations for any bounded function f under arbitrary distributions P and Q [28]

$$|\mathbb{E}_P[f]] - \mathbb{E}_Q[f]| \leq (\max f - \min f) \cdot 2d_{\mathrm{TV}}(P, Q). \tag{26}$$

Applying this property to our context, where $f = \mathcal{H}_{\mathrm{VN}}$, $P = P_T$, and $Q = \pi_n$, and substituting the mixing condition$d_{\mathrm{TV}}(P_T, \pi_n) \leq \varepsilon$, we arrive at the following bound

$$|\mathbb{E}[\mathcal{H}_{\mathrm{VN}}(G_T)] - \mathbb{E}_{G\sim\pi_n}[\mathcal{H}_{\mathrm{VN}}(G)]| \leq 2(lnn)\varepsilon, \tag{27}$$

Therefore, the evolution of EntSE exhibits two distinct regimes. In the pre-mixing phase ($T \ll \tau_{\mathrm{mix}}$), the distribution deviates significantly from uniformity, allowing the entropy to retain structural discriminability. In the post-mixing phase ($T \geq \tau_{\mathrm{mix}}$), as the distribution of G_T converges to the uniform distribution over all d-regular graphs [13], the bound derived above enforces the expected entropy $\mathbb{E}[\mathcal{H}_{\mathrm{VN}}(G_T)]$ to converge strictly to the ensemble mean $\mu_n = \mathbb{E}_{G\sim\pi_n}[\mathcal{H}_{\mathrm{VN}}(G)]$. Thus, Theorem 4.4 is proved. □

D Comparison with Structural Shannon Entropy on Strongly Regular Graphs

This section justifies the necessity of adopting von Neumann entropy by comparing it with the standard Structural Shannon Entropy. We first present the definition of graph Shannon entropy.

Definition D.1. The graph Shannon entropy is defined as

$$\mathcal{H}_S(G) = -\sum_{v\in V_G} \frac{deg(v)}{\sum_{m\in V_G} deg(m)} log \frac{deg(v)}{\sum_{m\in V_G} deg(m)}. \tag{28}$$

where $deg(v)$ denotes the degree of node v.

We also examine the challenging pair of strongly regular graphs presented in Figure 2: the 4×4 Rook's graph G_1 and the Shrikhande graph G_2. Specifically, we focus on the 1-hop node contextsS_u^1 and S_v^1, we observe that both exhibit identical degree distributions, i.e., $\{6, 3, 3, 3, 3, 3\}$. Consequently, computing the structural Shannon entropy for degree distribution yields $H_S(S_u^1) = H_S(S_v^1)$, indicating that Shannon entropy fails to distinguish these non-isomorphic structures. Further experiments comparing alternative entropy measures are detailed in Appendix J.12.

E Comparison with LapPE

Proposition E.1 (Incomparability between EntSE and LapPE). *There exist two infinite families of non-isomorphic graph pairs, $\{(G_1^{(n)}, G_2^{(n)})\}_{n\geq8}$ and $\{(G_3^{(n)}, G_4^{(n)})\}_{n\geq8}$, where each graph contains n nodes, such that LapPE distinguishes every pair $(G_1^{(n)}, G_2^{(n)})$ while EntSE does not, and EntSE distinguishes every pair $(G_3^{(n)}, G_4^{(n)})$ while LapPE does not.*

Proof. We exhibit two infinite families of witness pairs.

Family 1: LapPE distinguishes, EntSE does not. For any integer $n \geq 8$ and any partition $n = a_1 + a_2 + \cdots + a_r$ with $r \geq 2$ and $a_i \geq 4$ for all i, we define

$$G_1^{(n)} := C_n, \qquad G_2^{(n)} := C_{a_1} \sqcup C_{a_2} \sqcup \cdots \sqcup C_{a_r}, \tag{29}$$

where C_m denotes the cycle on m nodes and $\sqcup$ denotes disjoint union. Both graphs have n nodes, and we consider EntSE at $k = 1$.

EntSE fails to distinguish them. In both $G_1^{(n)}$ and $G_2^{(n)}$, every node u has degree 2. Since each cycle component has length $a_i \geq 4$, the two neighbors of any node are non-adjacent. Therefore, for every $u \in V(G_1^{(n)}) \cup V(G_2^{(n)})$, the 1-hop context S_u^1 is rooted-isomorphic to (P_3, c), where P_3 is

the path on three nodes rooted at its center c. Since von Neumann entropy is invariant under graph isomorphism, the contextual descriptor satisfies

$$\mathcal{D}_u^1 = \mathrm{EntSE}(u) = \mathcal{H}_{\mathrm{VN}}(S_u^1) = \mathcal{H}_{\mathrm{VN}}(P_3), \quad \forall\, u \in V(G_1^{(n)}) \cup V(G_2^{(n)}). \tag{30}$$

It follows that

$$\left\{\!\!\left\{ \mathcal{D}_u^1 \mid u \in V(G_1^{(n)}) \right\}\!\!\right\} = \left\{\!\!\left\{ \mathcal{D}_v^1 \mid v \in V(G_2^{(n)}) \right\}\!\!\right\} = \left\{\!\!\left\{ \mathcal{H}_{\mathrm{VN}}(P_3) \right\}\!\!\right\}^n. \tag{31}$$

Therefore, EntSE assigns identical descriptors to all nodes in $G_1^{(n)}$ and $G_2^{(n)}$ at every iteration, and thus fails to distinguish the two graphs.

LapPE distinguishes them. The multiplicity of the zero eigenvalue of the graph Laplacian equals the number of connected components. Since $G_1^{(n)} = C_n$ is connected, while $G_2^{(n)}$ consists of $r \geq 2$ connected components, the zero eigenvalue has multiplicity 1 for $G_1^{(n)}$ and multiplicity r for $G_2^{(n)}$. Consequently, their Laplacian eigenspaces, and hence the resulting LapPE representations, are fundamentally different. Thus, LapPE distinguishes the pair. The smallest instance is $(C_8,\ C_4 \sqcup C_4)$, as illustrated in Figure 3. This proves claim (1).

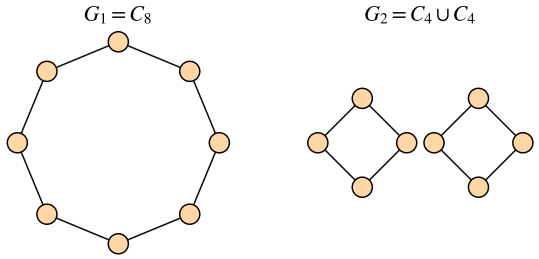


Figure 3: Sample instances of Family 1.

Family 2: EntSE distinguishes, LapPE does not. We construct the second family from a pair of connected, non-isomorphic L-cospectral graphs on six nodes. Let G_3^* and G_4^* be the two graphs defined by the following edge sets:

$$\begin{aligned} E(G_3^*) &= \{(0,2),(1,2),(1,3),(1,4),(2,5),(3,5),(4,5)\}, \\ E(G_4^*) &= \{(0,2),(0,5),(1,3),(1,4),(2,3),(3,4),(3,5)\}. \end{aligned} \tag{32}$$

As illustrated in Figure 4, we visualize this pair of graphs and refer to it as the *seed pair*.

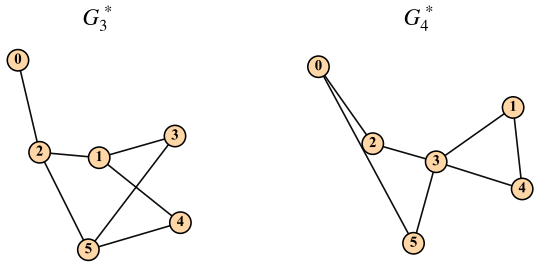


Figure 4: The seed pair of Family 2.

We compute their common Laplacian spectrum as

$$\sigma(L_{G_3^*}) = \sigma(L_{G_4^*}) = \{0,\ 0.764,\ 2,\ 3,\ 3,\ 5.236\}, \tag{33}$$

which implies that LapPE cannot distinguish the two graphs. However, their multisets of 1-hop contextual entropies are different:

$$\begin{aligned} \left\{\!\!\left\{ \mathcal{D}_u^1 \mid u \in V(G_3^*) \right\}\!\!\right\} &= \{0,\ 0.562,\ 0.562,\ 0.868,\ 0.868,\ 0.868\}, \\ \left\{\!\!\left\{ \mathcal{D}_v^1 \mid v \in V(G_4^*) \right\}\!\!\right\} &= \{0.562,\ 0.562,\ 0.562,\ 0.693,\ 0.693,\ 1.168\}. \end{aligned} \tag{34}$$

Therefore, EntSE distinguishes the seed pair.

We now extend the seed pair to an infinite family. For each $n \geq 8$, let H_n be any graph on $n-6$ nodes, and define

$$G_3^{(n)} := G_3^* \sqcup H_n, \qquad G_4^{(n)} := G_4^* \sqcup H_n, \tag{35}$$

where $\sqcup$ denotes the disjoint union of graphs.

LapPE fails to distinguish them. The Laplacian of a disjoint union is block-diagonal. Therefore, its spectrum is the multiset union of the spectra of its connected components. Let $\sigma(L_G)$ denote the spectrum of the graph Laplacian L_G. Then, for any graphs A and B,

$$\sigma(L_{A \sqcup B}) = \sigma(L_A) \uplus \sigma(L_B), \tag{36}$$

where $\uplus$ denotes multiset union, preserving eigenvalue multiplicities. Applying this identity to the cospectral pair $G_3^{(n)}$ and $G_4^{(n)}$, we obtain

$$\begin{aligned} \sigma\big(L_{G_3^{(n)}}\big) &= \sigma(L_{G_3^*}) \uplus \sigma(L_{H_n}) \\ &= \sigma(L_{G_4^*}) \uplus \sigma(L_{H_n}) \\ &= \sigma\big(L_{G_4^{(n)}}\big). \end{aligned} \tag{37}$$

Hence, LapPE cannot distinguish $G_3^{(n)}$ from $G_4^{(n)}$ for any $n \geq 8$.

EntSE distinguishes them. Since G_3^* and H_n are vertex-disjoint and no edges connect them in $G_3^{(n)} = G_3^* \sqcup H_n$, the 1-hop context of each node in G_3^* remains unchanged after taking the disjoint union with H_n. The same argument applies to G_4^* in $G_4^{(n)} = G_4^* \sqcup H_n$. Therefore,

$$\begin{aligned} \{\!\{\mathcal{D}_u^1 \mid u \in V(G_3^{(n)})\}\!\} &= \{\!\{\mathcal{D}_u^1 \mid u \in V(G_3^*)\}\!\} \uplus \{\!\{\mathcal{D}_w^1 \mid w \in V(H_n)\}\!\}, \\ \{\!\{\mathcal{D}_v^1 \mid v \in V(G_4^{(n)})\}\!\} &= \{\!\{\mathcal{D}_v^1 \mid v \in V(G_4^*)\}\!\} \uplus \{\!\{\mathcal{D}_w^1 \mid w \in V(H_n)\}\!\}. \end{aligned} \tag{38}$$

By Equation 34, the two seed graphs have different multisets of 1-hop contextual entropies. Since these seed contexts are preserved under the disjoint union with H_n, we have

$$\{\!\{\mathcal{D}_u^1 \mid u \in V(G_3^{(n)})\}\!\} \neq \{\!\{\mathcal{D}_v^1 \mid v \in V(G_4^{(n)})\}\!\}. \tag{39}$$

Therefore, EntSE distinguishes $G_3^{(n)}$ and $G_4^{(n)}$.

In summary, EntSE and LapPE are incomparable in terms of distinguishing power. □

F Proof of the property of Reproducing kernel

Intuitively, our kernel is a bounded, symmetric function that is monotonically decreasing in the entropy distance between x and y. In our setting, x and y are entropy-based representations of the structural distributions of the contexts S_u^k and S_v^k. Thus, when two node contexts have identical structural distributions, their entropy representations are the same and then $\mathcal{K}(x, y) = 1$. As the structural distributions diverge, the entropy distance $\|x - y\|$ increases and the kernel value gradually drops toward 0. Thus, similar structural distributions produce large kernel values, while dissimilar ones produce small kernel values, which is the notion of structural similarity that we aim to capture.

From a more rigorous mathematical standpoint, the kernel can be represented as the inner product of two feature maps, thereby making its role in capturing similarity explicit. A detailed proof is given below. Based on the definition of the kernel method [22], the kernel serves as a measure of similarity

$$\mathcal{K}(x, y) = \langle \Phi(x), \Phi(y) \rangle \tag{40}$$

where $\Phi(x)$ is the feature map and maps into the dot product space. For the reproducing kernel $\mathcal{K}(x, y) = \frac{1}{2} e^{-|x-y|}$, we can also represent it as an inner product between feature mappings, which explicitly shows how the kernel quantifies similarity.

Theorem F.1. *For the reproducing kernel $\mathcal{K}(x, y) = \frac{1}{2} e^{-|x-y|}$, there exists an explicit feature mapping ϕ defined by*

$$\Phi(x) = \left\{ \sqrt{\frac{1}{\pi(1+\omega^2)}} \cos(\omega x),\ \sqrt{\frac{1}{\pi(1+\omega^2)}} \sin(\omega x) \right\}_{\omega \geq 0}, \tag{41}$$

such that the kernel function can be expressed as the inner product in this feature space.

proof. Since $e^{-|x|}$ is an even function, its Fourier transform is real, $\mathcal{F}[e^{-|x|}](\omega) = \frac{2}{1+\omega^2}$, then

$$\mathcal{F}[\frac{1}{2}e^{-|x|}](\omega) = \frac{1}{1+\omega^2}. \tag{42}$$

Therefore, we can construct a real-valued feature mapping through Fourier transform as

$$\frac{1}{2}e^{-|x-y|} = \frac{1}{2\pi}\int_{-\infty}^{\infty}\frac{1}{1+\omega^2}e^{i\omega(x-y)}d\omega = \frac{1}{\pi}\int_{0}^{\infty}\frac{\cos(\omega(x-y))}{1+\omega^2}d\omega \tag{43}$$

Finally, we verify the validity by computing the inner product integral

$$\begin{aligned}
\langle\Phi(x),\Phi(y)\rangle &= \int_0^\infty\left[\frac{\cos(\omega x)\cos(\omega y)}{\pi(1+\omega^2)} + \frac{\sin(\omega x)\sin(\omega y)}{\pi(1+\omega^2)}\right]d\omega \\
&= \frac{1}{\pi}\int_0^\infty\frac{\cos(\omega x)\cos(\omega y)+\sin(\omega x)\sin(\omega y)}{1+\omega^2}d\omega \\
&= \frac{1}{\pi}\int_0^\infty\frac{\cos(\omega(x-y))}{1+\omega^2}d\omega \\
&= \frac{1}{2}e^{-|x-y|}.
\end{aligned}$$

Thus, the Theorem F.1 holds.

G Proof of Theorem 4.5

Proof. Let $\delta_w := |\hat{\mathcal{H}}_w - \mathcal{H}_w|$ denote the change in the von Neumann entropy of subgraph S_w^k under edge perturbation, for $w \in \{u, v\}$. The proof proceeds in three steps: (i) we bound $|\delta_w|$ via the Fannes-Audenaert inequality, (ii) we bound T_w via spectral perturbation, and (iii) we combine these bounds with the Lipschitz continuity of the kernel function $\mathcal{K}$.

(i) Bounding $|\delta_w|$ via the Fannes-Audenaert inequality. We first provide a bound on the change in von Neumann entropy under edge perturbations for each $w \in \{u, v\}$. Let S_w^k be the k-hop subgraph centered at w, and let L_w denote its combinatorial Laplacian. Since L_w is positive semidefinite and $\mathrm{tr}(L_w) = 2|E_w|$, we can normalize it into a density operator:

$$\rho_w := \frac{L_w}{\mathrm{tr}(L_w)} = \frac{L_w}{2|E_w|}. \tag{44}$$

Let $\lambda_1^{(w)}, \ldots, \lambda_n^{(w)}$ be the eigenvalues of L_w. Then the eigenvalues of ρ_w are

$$p_i^{(w)} = \frac{\lambda_i^{(w)}}{2|E_w|}, \qquad i = 1, \ldots, n. \tag{45}$$

thus the vector $\mathbf{p}^{(w)}$ forms a probability distribution. Therefore, the von Neumann entropy of S_w^k can be written as the Shannon entropy of this spectral distribution

$$\mathcal{H}_{\mathrm{VN}}(S_w^k) = -\mathrm{tr}(\rho_w \log \rho_w) = H(\mathbf{p}^{(w)}) = -\sum_{i=1}^{n} p_i^{(w)} \log p_i^{(w)}. \tag{46}$$

After edge perturbation, let $\hat{L}_w$ be the perturbed Laplacian and $\hat{\rho}_w$ be the corresponding normalized density operator. We denote the eigenvalue distribution of $\hat{\rho}_w$ by $\mathbf{q}^{(w)} = \left(q_1^{(w)}, \ldots, q_n^{(w)}\right)$.

Then the entropy change caused by the perturbation is

$$\delta_w := \left|\hat{\mathcal{H}}_w - \mathcal{H}_w\right| = \left|H(\mathbf{q}^{(w)}) - H(\mathbf{p}^{(w)})\right|. \tag{47}$$

To bound this entropy change, we apply the Fannes–Audenaert inequality to the two spectral distributions $\mathbf{p}^{(w)}$ and $\mathbf{q}^{(w)}$. Let $T_w = \frac{1}{2}\left\|\mathbf{p}^{(w)} - \mathbf{q}^{(w)}\right\|_1$ be their total variation distance. Then, for $0 \le T_w \le 1 - 1/n$, we have

$$\delta_w \le T_w \log(n-1) + h(T_w), \tag{48}$$

where $h(x) = -x \log x - (1-x)\log(1-x)$ is the binary entropy function.

(ii) Bounding T_w via spectral perturbation. We next bound the spectral total variation distance T_w in terms of the edge-perturbation budget m_w. For clarity, we suppress the superscript (w) in this step. Let E and $\hat{E}$ denote the edge sets of S_w^k before and after perturbation, respectively. We assume $m_w < |E|$, so that

$$|\hat{E}| \geq |E| - m_w > 0.$$

Recall that

$$p_i = \frac{\lambda_i}{2|E|}, \qquad q_i = \frac{\hat{\lambda}_i}{2|\hat{E}|},$$

where $\{\lambda_i\}_{i=1}^n$ and $\{\hat{\lambda}_i\}_{i=1}^n$ are the eigenvalues of L_w and $\hat{L}_w$, respectively, arranged in the same order. Then

$$q_i - p_i = \frac{\hat{\lambda}_i - \lambda_i}{2|\hat{E}|} + \lambda_i \left(\frac{1}{2|\hat{E}|} - \frac{1}{2|E|} \right). \tag{49}$$

We first bound the eigenvalue-perturbation term $\frac{\hat{\lambda}_i - \lambda_i}{2|\hat{E}|}$. Adding or deleting a single edge (a, b) corresponds to a rank-one update of the Laplacian:

$$\Delta L_{ab} = \pm(\mathbf{e}_a - \mathbf{e}_b)(\mathbf{e}_a - \mathbf{e}_b)^\top. \tag{50}$$

whose nuclear norm is

$$\|\Delta L_{ab}\|_* = \|\mathbf{e}_a - \mathbf{e}_b\|_2^2 = 2.$$

Thus, by the triangle inequality, m_w such perturbations satisfy

$$\|\hat{L}_w - L_w\|_* \leq 2m_w. \tag{51}$$

By Mirsky's inequality for unitarily invariant norms [35],

$$\sum_{i=1}^n |\hat{\lambda}_i - \lambda_i| \leq \|\hat{L}_w - L_w\|_* \leq 2m_w. \tag{52}$$

Therefore,

$$\sum_{i=1}^n \frac{|\hat{\lambda}_i - \lambda_i|}{2|\hat{E}|} \leq \frac{2m_w}{2(|E| - m_w)} = \frac{m_w}{|E| - m_w}. \tag{53}$$

We then bound the normalization term $\lambda_i(\frac{1}{2|\hat{E}|} - \frac{1}{2|E|})$. Since

$$\sum_{i=1}^n \lambda_i = \mathrm{tr}(L_w) = 2|E|$$

and an m_w-edge perturbation changes the number of edges by at most m_w, we have

$$\big||\hat{E}| - |E|\big| \leq m_w.$$

Hence,

$$\begin{aligned} \sum_{i=1}^n \lambda_i \left| \frac{1}{2|\hat{E}|} - \frac{1}{2|E|} \right| &= 2|E| \cdot \frac{\big||E| - |\hat{E}|\big|}{2|E|\,|\hat{E}|} \\ &= \frac{\big||E| - |\hat{E}|\big|}{|\hat{E}|} \\ &\leq \frac{m_w}{|E| - m_w}. \end{aligned} \tag{54}$$

Combining (53) and (54) with the triangle inequality gives

$$\|\mathbf{p}^{(w)} - \mathbf{q}^{(w)}\|_1 \leq \frac{2m_w}{|E_w| - m_w}. \tag{55}$$

Since

$$T_w = \frac{1}{2}\|\mathbf{p}^{(w)} - \mathbf{q}^{(w)}\|_1,$$

we obtain

$$T_w \leq \frac{m_w}{|E_w| - m_w} =: T_{m_w}. \tag{56}$$

Finally, substituting this bound into the Fannes–Audenaert inequality yields

$$|\delta_w| \leq T_{m_w} \log(n-1) + h(T_{m_w}) =: \varepsilon(m_w, n). \tag{57}$$

(iii) Composition with Lipschitz continuity of $\mathcal{K}$. Let $\Delta_{uv} = |\mathcal{H}_{\mathrm{VN}}(S_u^k) - \mathcal{H}_{\mathrm{VN}}(S_v^k)|$ denote the entropic difference between a node pair (u, v). Let the kernelized similarity be defined as $\mathrm{KerSE}(u, v) = f(\Delta_{uv})$, where $f(x) = \frac{1}{2}e^{-x}$ over the domain $x = \Delta_{uv} = |\mathcal{H}_u - \mathcal{H}_v| \in [0, +\infty)$. The function $f(x)$ is differentiable on $(0, +\infty)$ with its derivative given by

$$f'(x) = -\frac{1}{2}e^{-x}. \tag{58}$$

Given that $|e^{-x}| \leq 1$ for all $x \geq 0$, the absolute value of the derivative is bounded such that

$$\sup_{x \geq 0} |f'(x)| = \frac{1}{2}. \tag{59}$$

This confirms that $f(x)$ is an L-Lipschitz continuous function with a Lipschitz constant $L = \frac{1}{2}$. By the definition of Lipschitz continuity, for the original input Δ_{uv} and the perturbed input $\hat{\Delta}_{uv}$, the difference in kernel values satisfies the following inequality

$$|\mathrm{KerSE}(u, v) - \mathrm{KerSE}(\hat{u}, \hat{v})| \leq \frac{1}{2}|\Delta_{uv} - \hat{\Delta}_{uv}| \tag{60}$$

Expanding the term $|\Delta_{uv} - \hat{\Delta}_{uv}|$ and applying the reverse triangle inequality, i.e., $||a| - |b|| \leq |a - b|$, we obtain as

$$|\Delta_{uv} - \hat{\Delta}_{uv}| = \Big||\mathcal{H}_u - \mathcal{H}_v| - |\hat{\mathcal{H}}_u - \hat{\mathcal{H}}_v|\Big| \leq |(\mathcal{H}_u - \mathcal{H}_v) - (\hat{\mathcal{H}}_u - \hat{\mathcal{H}}_v)| \tag{61}$$

By substituting the perturbed entropy terms $|\hat{\mathcal{H}}_u - \mathcal{H}_u| = \delta_u$ and $|\hat{\mathcal{H}}_v - \mathcal{H}_v| = \delta_v$, the expression simplifies to

$$|(\mathcal{H}_u - \hat{\mathcal{H}}_u) - (\mathcal{H}_v - \hat{\mathcal{H}}_v)| = |\delta_u - \delta_v|. \tag{62}$$

By applying the triangle inequality, i.e., $|\delta_v - \delta_u| \leq |\delta_u| + |\delta_v|$, we arrive at the following upper bound for the variation in relational similarity

$$|\mathrm{KerSE}(u, v) - \mathrm{KerSE}(\hat{u}, \hat{v})| \leq \frac{1}{2}(|\delta_u| + |\delta_v|). \tag{63}$$

Combining this with Step 2 gives

$$|\mathrm{KerSE}(u, v) - \mathrm{KerSE}(\hat{u}, \hat{v})| \leq \frac{1}{2}\big(\varepsilon(m_u, n) + \varepsilon(m_v, n)\big), \tag{64}$$

which is the claimed end-to-end stability bound. □

This result establishes that the influence of local structural perturbations is strictly bounded and attenuated by the kernel mechanism, thereby completing the proof of Theorem 4.5.

H The Time, Space Complexity Comparison

The computational overhead of our framework is decomposed into a one-time offline preprocessing phase and an efficient online training stage. The primary cost of preprocessing stems from the extraction of local k-hop contexts and the subsequent entropy-based analysis. Let $|V|$ and $|E|$ denote the numbers of nodes and edges in the graph, respectively, and let d be the average node degree. Extracting the k-hop context for each node incurs a one-time cost of $\mathcal{O}(|V| \cdot d^k)$, while

the corresponding entropy computation introduces an additional cost of $\mathcal{O}(|V| \cdot d^{3k})$. Although this localized spectral analysis is more computationally demanding than simple structural encodings such as node degree, it remains substantially more scalable than global spectral approaches. For example, Laplacian positional encoding (LapPE) [27] requires a prohibitive $\mathcal{O}(|V|^3)$ complexity due to full-graph eigendecomposition. In contrast, since both d and k are typically small constants in practice, the overall preprocessing cost of our method scales linearly with the number of nodes $|V|$, making it applicable to large-scale graphs where global spectral methods become infeasible.

Caching repeated contexts. We further reduce preprocessing cost in practice by caching the von Neumann entropy of structurally equivalent contexts. Since entropy is invariant under graph isomorphism, two nodes whose k-hop contexts are isomorphic share the same entropy value. Therefore, we hash each extracted context and reuse cached entropy values for repeated structures. This optimization is particularly effective for graphs with regular or repetitive local patterns, such as molecular graphs.

During training, the integration of EntSE and KerSE introduces only a lightweight linear projection layer for feature mapping. As a result, the proposed framework preserves the same asymptotic time and space complexity as the underlying backbone architecture. We also compare the time and space complexity of our MPNN-based models with several representative baselines, including the standard MPNN [53], Subgraph MPNN [60], I^2-GNN [24], PPGN [32], and 1-2-3 GNN [36]. The subgraph MPNN introduces an additional factor of s in complexity due to subgraph-level GNN, where s denotes the maximum subgraph size. The d denotes the average degree. While the other methods enhance expressive power, they do so at the cost of significantly increased time and space complexity. In contrast, our proposed models, ConD-MPNN and KerA-MPNN, maintain linear space and time complexity. Moreover, both theoretical analysis and empirical results demonstrate that our models achieve expressive power comparable to that of these more complex architectures. This highlights the scalability of our framework while preserving strong expressive power.

Table 8: Comparison of time and space complexity among different models.

Model	MPNN	Subgraph MPNN	I^2-GNN	PPGN	1-2-3 GNN	**ConD-MPNN**	**KerA-MPNN**
Space complexity	$\mathcal{O}(\|V\|)$	$\mathcal{O}(s\|V\|)$	$\mathcal{O}(s\|V\| \cdot d)$	$\mathcal{O}(\|V\|^2)$	$\mathcal{O}(\|V\| \cdot d^2)$	$\mathcal{O}(\|V\|)$	$\mathcal{O}(\|V\|)$
Time complexity	$\mathcal{O}(\|V\| \cdot d)$	$\mathcal{O}(s\|V\| \cdot d)$	$\mathcal{O}(s\|V\| \cdot d^2)$	$\mathcal{O}(\|V\|^3)$	$\mathcal{O}(\|V\| \cdot d^3)$	$\mathcal{O}(\|V\| \cdot d)$	$\mathcal{O}(\|V\| \cdot d)$

I Integration into Backbones: K-hop MPNN, PPGN, and GraphGPS.

In this section, we demonstrate the generality of our framework by integrating EntSE and KerSE into three representative GNN backbone architectures. All integrations follow the same principles described in the main paper and do not modify the core message passing or attention mechanisms of the original backbones.

K-hop Message Passing Neural Network (K-hop MPNN). In the K-hop MPNN [40], the messages from 1-hop to K-hop neighborhoods are simultaneously propagated and aggregated, and a combination function is applied to integrate the resulting multi-hop node representations. The standard K-hop MPNN updates the node representation at iteration t as

$$\boldsymbol{h}_u^{t+1,k} = \mathbf{U}^{t,k}(\boldsymbol{h}_u^t, \boldsymbol{m}_u^{t+1,k}), \quad \boldsymbol{m}_u^{t+1,k} = \sum\nolimits_{v \in \mathcal{N}_u^k} \mathbf{M}^{t,k}(\boldsymbol{h}_u^t, \boldsymbol{h}_v^t, \boldsymbol{e}_{uv}^k), \tag{65}$$

where v is the k-hop neighbor of the rooted node u, $\boldsymbol{h}_u^{t+1,k}$ denotes the representation of the rooted node u at layer $t+1$, aggregated from its k-hop neighborhood, and $\boldsymbol{e}_{uv}^k$ denotes the shortest path or random walk between u and v. The final node representation at iteration $t+1$ is updated through a combination function $\mathbf{C}$

$$\boldsymbol{h}_u^{t+1} = \mathbf{C}^{t+1}\{\boldsymbol{h}_u^{t+1,k} | k = 1, 2, .., K\}. \tag{66}$$

The K-hop MPNN is particularly suitable for our framework, as the K-hop contexts are naturally obtained during the extraction of K-hop neighboring nodes, eliminating the need for additional preprocessing steps. In both the discrete ConD-WL and the learnable ConD-MPNN variants introduced earlier, the receptive field is determined by the context diameter K and the layer number t, i.e., $max\{K, t\}$. When extending ConD-MPNN to the K-hop MPNN framework [40], yielding in

ConD-KMPNN. In this setting, the receptive field expands to Kt, enabling efficient coverage of larger neighborhoods with fewer layers.

We now instantiate EntSE into the K-hop MPNN. At layer $t+1$, the node representation associated with the k-hop neighboring aggregation is updated as

$$\begin{aligned} \boldsymbol{h}_u^{t+1,k} &= \mathbf{U}^{t,k}(\boldsymbol{h}_u^t, \boldsymbol{m}_u^{t+1,k}), \\ \boldsymbol{m}_u^{t+1,k} &= \sum\nolimits_{v \in \mathcal{N}_u^k} \mathbf{M}^{t,k}(\boldsymbol{h}_u^t, \boldsymbol{h}_v^t, \phi(\mathcal{H}_{\mathrm{VN}}(S_u^k)), \phi(\mathcal{H}_{\mathrm{VN}}(S_v^k)), \boldsymbol{e}_{uv}^k). \end{aligned} \tag{67}$$

Furthermore, the KerSE can also be integrated into the K-hop MPNN. At the $t+1$-th iteration, the node representation can be updated based on the aggregation of k-hop neighbors as follows

$$\boldsymbol{h}_u^{t+1,k} = \mathbf{U}^{t,k}(\boldsymbol{h}_u^t, \boldsymbol{m}_u^{t+1,k}), \quad \boldsymbol{m}_u^{t+1,k} = \sum\nolimits_{v \in \mathcal{N}_u^k} \mathbf{M}^{t,k}(\boldsymbol{h}_u^t, \boldsymbol{h}_v^t, \mathcal{K}(S_u^k, S_v^k), \boldsymbol{e}_{uv}^k), \tag{68}$$

where $\mathcal{K}(S_u^k, S_v^k)$ denotes the Reproducing kernel between S_u^k and S_v^k discussed in Section 4.3.

Provably Powerful Graph Neural Network (PPGN). The PPGN [32] achieves expressivity matching the 3-WL test by composing multilayer perceptrons (MLPs) with matrix multiplication operations over pairwise node representations. The input is a third-order tensor $\boldsymbol{X} \in \mathbb{R}^{n \times n \times d}$. Formally, the PPGN architecture can be written as

$$F = M \circ L \circ B_l \circ B_{l-1} \circ \cdots \circ B_1 \tag{69}$$

where M denotes an MLP, L is a permutation-invariant layer, and each B_i consists of MLPs and matrix multiplication operations. We enhance PPGN by incorporating EntSE and KerSE into the construction of the initial input tensor, thereby enriching the pairwise representations with contextual structural information while preserving the original architecture. The initial tensor $\boldsymbol{X}^0$ in PPGN with EntSE is enriched as follows

$$\boldsymbol{X}^0 = \left[(\boldsymbol{A} + \boldsymbol{I}) \parallel \boldsymbol{E}^{(1)} \parallel \boldsymbol{E}^{(2)} \parallel \cdots \parallel \boldsymbol{E}^{(K)}\right] \in \mathbb{R}^{n \times n \times (K+1)}, \tag{70}$$

where the diagonal entries capture the entropic structural identity of each node, i.e.,

$$E_{uv}^{(k)} = \begin{cases} \mathcal{H}_{\mathrm{VN}}(S_u^k), & \text{if } u = v, \\ 0, & \text{if } u \neq v. \end{cases} \tag{71}$$

For the KerSE variant, the input tensor $\boldsymbol{X}$ is constructed by concatenating the adjacency matrix and m kernel-based feature matrices along the channel dimension

$$\boldsymbol{X}^0 = \left[(\boldsymbol{A} + \boldsymbol{I}) \parallel \boldsymbol{\Phi}^{(1)} \parallel \boldsymbol{\Phi}^{(2)} \parallel \cdots \parallel \boldsymbol{\Phi}^{(m)}\right], \tag{72}$$

where m denotes the sampling number of neighbors in Eq. 16. The $(p+1)$-th channel of the input tensor is constructed as a diagonal matrix encoding the p-th kernel-based feature

$$\boldsymbol{\Phi}_{uv}^{(p)} = \begin{cases} f_{u,p}, & \text{if } u = v, \\ 0, & \text{if } u \neq v, \end{cases} \qquad p = 1, \ldots, m, \tag{73}$$

where $f_{u,p}$ represents the structural similarity score between node u and its neighbor p, obtained by averaging entropy-based kernel values over the 1-to-K hop induced subgraphs.

$$f_{u,p} = \frac{1}{K} \sum_{k=1}^{K} \mathcal{K}(S_u^k, S_p^k). \tag{74}$$

This design enables PPGN to exploit high-order structural priors, enhancing its ability to distinguish complex graph structures.

GraphGPS. GraphGPS [45] is a state-of-the-art hybrid framework that combines a local MPNN branch with a global Transformer branch, where each GPS layer jointly performs neighborhood aggregation and global self-attention. We integrate our method into GraphGPS via a dual-track structural design. For EntSE, we simply concatenate the entropy encoding to the original node features, following Eq. 14. For KerSE, we adopt branch-specific integrations. In the MPNN branch, for each node u, we retain the use of EntSE to capture the local contextual distribution around each node. In the Transformer branch, we use the kernel similarity as a relational structural bias for self-attention as Eq. 15. This hybrid integration enables the Transformer branch to explicitly account for structural relationships beyond raw connectivity, improving its ability to model long-range dependencies in graphs with complex topological patterns.

J Additional Experimental Information

In this section, we provide further information about the datasets, describe the implementation details, and report additional experimental results to support our main findings.

J.1 Datasets

We evaluate the expressive power on four synthetic datasets specifically designed to test the limitations of MPNNs: (1) the EXP dataset [1], consisting of 600 pairs of non-isomorphic graphs; (2) the CSL dataset [39], comprising 150 4-regular circulant graphs; (3) the SR25 dataset [4], containing 15 non-isomorphic strongly regular graphs; (4) the BREC dataset [51]. Notably, all of these datasets cannot be distinguished by the 1-WL test, especially BREC, which remains indistinguishable even under the more powerful 3-WL and 4-WL tests. For real-world benchmarks, we evaluate the models on seven TU datasets [37] for graph classification, and on QM9 [44] and the ZINC [17] for graph regression. To test the scalability on larger graphs, we also evaluate performance using the Peptides-Func dataset [16]. Table 9 summarizes the key statistics of the graph datasets used in our experiments. These datasets exhibit diverse characteristics in terms of the number of graphs, nodes, edges, and classes, thereby enabling a comprehensive evaluation of model scalability, expressive power, and generalization across various domains and graph structures.

Table 9: Information of the graph datasets used in our experiments.

Datasets	# of graphs	Mean # nodes	Mean # edges	Classes	Description
MUTAG	188	17.93	19.79	2	Mole
PROTEINS	1113	39.06	72.82	2	Bio
PTC_MR	344	14.29	14.69	2	Mole
COLLAB	5000	74.49	2457.78	3	SN
IMDB-B	1000	19.77	96.53	2	SN
IMDB-M	1500	13.00	65.94	3	SN
DD	1178	284.32	715.66	2	Bio
Peptides-Func	15,535	150.94	307.30	10	Mole
OGBG-MolHIV	41,127	25.50	27.50	2	Mole
OGBG-MolPCBA	437,929	26.00	28.10	128	Mole
EXP	1000	5.00	5.00	10	Synthetic
SR25	500	25.00	50.00	2	Synthetic
CSL	150	10.00	20.00	10	Synthetic
QM9	130831	18.00	19.00	12	Mole
ZINC	12000	23.15	24.99	1	Mole

J.2 Experimental Setting

This section presents the implementation details of each model. We organize the description by task, and for each specific task, we detail the architecture settings, dataset splits, and evaluation protocols.

For **expressiveness evaluation**, we primarily use the BREC dataset. We adopt PPGN, K-GIN, and GraphGPS as backbones, and strictly follow the evaluation protocol for BREC in [51]. For all three models, we set $K = 8$. In addition to BREC, we also evaluate expressiveness on the EXP, CSL, and SR datasets, where we use GIN and K-GIN as backbones. For these datasets, we search for k in $\{1, 2, 3, 4\}$. Specifically, for the CSL and EXP datasets, we repeat each experiment four times and report the average results. For the SR dataset, we run the experiment once and report the best result, following the same protocol used in prior work [19].

For **graph regression**, we primarily use the ZINC dataset. We adopt PPGN, K-GIN, and GraphGPS as backbones, and the architectural hyperparameters are chosen from the original search ranges of the corresponding backbones to ensure a fair comparison. For PPGN, we follow the settings in [42] and tune the subgraph size K over $\{3, 4, 5, 6\}$. For K-GIN, we tune the node context size in $\{8, 16\}$, the number of layers in $\{9, 17\}$, and the hidden dimension in $\{94, 106\}$. For GraphGPS, we tune the context size in $\{3, 4, 5, 6\}$ and the number of sampled neighbors in $\{8, 10, 16\}$. When a node has fewer than the required neighbors, we apply zero-padding to keep the input dimensionality fixed. On ZINC, we run 4 random seeds for K-GIN and PPGN, and 10 random seeds for GraphGPS. For QM9, we use K-GIN as the backbone and set the context size to 16. For the Peptides-Func dataset, we use K-GIN as the backbone. To ensure a fair comparison with existing evaluation protocols on

this dataset, we additionally tune the network depth, searching the number of layers in $\{6, \ldots, 16\}$, and set the context size K in $\{6, \ldots, 16\}$.

For **graph classification**, we primarily conduct evaluations on the TUDataset benchmarks. We first incorporate distributional information into conventional WL graph kernels by using histograms as additional features, i.e., the WL+HistSE. We follow the same experimental setup as the original WL test [46]. We conduct the 10-fold cross-validation, selecting one fold from the training set for validation in each run. The experiments are repeated ten times, and the average performance along with the standard deviation is reported. For TU datasets, we search the number of hop k in $\{1, 2, 3, 4\}$, the hyperparameter C of the C-SVM classifier in $\{1e-7, 1e-5, 1e-3, 1e-1, 1e+1, 1e+3, 1e+5, 1e+7\}$. In addition, we also use GIN and K-GIN as backbones. For GIN+EntSE and GIN+KerSE, we follow the same experimental setup as the original GIN [53]. Specifically, we conduct 10-fold cross-validation and report the average test accuracy across all folds. For TU datasets, we follow the exact hyperparameter search range employed in the original GIN implementation to ensure a fair comparison. For all datasets, the dimensionality of the linear transformation ϕ matches the feature dimension of h in the hidden layer, and is selected from the set $\{16, 32, 64, 128\}$ through a search process. For K-GIN+EntSE and K-GIN+KerSE, we follow the same experimental setup as KP-GIN [19] to ensure a fair comparison. Specifically, we adopt the exact hyperparameter search ranges used in KP-GIN. For the TU datasets and the three synthetic benchmarks, the implementation details are identical to those used in the corresponding GIN-based variants. We evaluate all methods using two context construction strategies, i.e., the shortest path distance and random walk.

All experiments were conducted on a server equipped with four NVIDIA GeForce RTX 3090 GPUs, each with 24 GB of memory.

J.3 Expressiveness Evaluation on EXP, CSL, and SR Datasets

In this subsection, we evaluate expressive power on three widely used benchmarks, EXP, CSL, and SR, using GIN and K-GIN as backbones. We compare our methods against classic expressiveness-enhancing baselines, including GCN-RNI [1], Nested GNN [60], PPGN [32], and GIN-AK+ [61]. Under the same backbones, we further compare two representative positional or structural encoding (PSE) methods with our proposed EntSE and KerSE. The results show that EntSE and KerSE achieve 100% accuracy on all three datasets, indicating that they can perfectly distinguish non-isomorphic graph pairs. Overall, our methods consistently outperform prior expressiveness baselines and PSE approaches, demonstrating stronger structural modeling ability.

Table 10: Evaluation of expressiveness (accuracy).

Method	EXP	CSL	SR
GCN-RNI	98.0 ± 1.85	16.0 ± 0.00	6.67
ID-GNN	100.0 ± 0.00	–	6.67
Nested GNN	99.9 ± 0.26	–	6.67
PPGN	100.0 ± 0.00	–	6.67
GIN-AK+	100.0 ± 0.00	–	6.67
GIN	50.0 ± 0.0	10.0 ± 0.0	6.7
GIN + RWSE	50.0 ± 0.0	32.0 ± 2.8	40
GIN + LapPE	99.6 ± 0.8	98.0 ± 3.2	100.0
GIN + EntSE	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	**100.0**
GIN + KerSE	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	**100.0**
K-GIN	100.0 ± 0.0	92.7 ± 0.0	6.7
K-GIN + RWSE	100.0 ± 0.0	92.0 ± 2.8	20.0
K-GIN + LapPE	100.0 ± 0.0	99.3 ± 2.1	100.0
K-GIN + EntSE	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	**100.0**
K-GIN + KerSE	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	**100.0**

J.4 Comparison with Different Backbone

To more comprehensively evaluate the effectiveness of our proposed framework, we further instantiate both methodologies within the K-GCN and K-GraphSAGE backbones, as reported in Table 11. We compare classification performance against the KP-GCN and KP-GraphSAGE baselines [19] across five datasets. Overall, augmenting either the GCN or GraphSAGE framework with our structural encodings leads to improvements on most datasets, indicating that EntSE and KerSE

Table 11: Comparison under different backbones.

Method	MUTAG	DD	PTC-MR	PROTEINS	IMDBBINARY
KP-GCN	91.7±6.0	79.0±4.7	**67.1±6.3**	75.8±3.5	75.9±3.8
K-GCN+EntSE	**92.2±6.5**	**79.9±4.2**	65.9±8.9	**76.5±3.7**	75.9±4.4
K-GCN+KerSE	**92.8±5.3**	**80.3±3.5**	66.8±6.2	**76.6±4.7**	**76.0±4.3**
KP-GraphSAGE	91.7±6.5	78.1±2.6	66.5±4.0	**76.5±4.6**	76.4±2.7
K-GraphSAGE+EntSE	**92.2±5.5**	**77.6±3.2**	**67.4±7.9**	76.3±5.2	76.1±5.9
K-GraphSAGE+KerSE	**92.8±5.3**	**78.7±3.6**	**67.1±6.6**	75.9±4.5	**76.5±3.0**

provide a generally applicable structural inductive bias. Specifically, the K-GCN+KerSE attains the highest mean accuracy on DD, PROTEINS, and IMDBBINARY. Similarly, the K-GraphSAGE with EntSE and KerSE match or slightly exceed the performance of KP-GraphSAGE [19], with K-GraphSAGE+KerSE achieving the best accuracy on four out of the five datasets. These results indicate that the proposed context distribution can be seamlessly integrated into different message passing backbones while maintaining competitive performance, and they further suggest that kernel-based aggregation is effective in mitigating distributional discrepancies between contexts rooted at neighboring nodes.

J.5 Evaluation on OGB Datasets

As shown in Table 12, we further evaluate our structural encodings on the molecular property prediction benchmarks MolHIV and MolPCBA using GraphGPS as the backbone. Overall, our methods achieve competitive performance compared with existing positional and structural encodings [10, 18, 27]. On MolHIV, GPS+KerSE obtains an AUROC of 0.7833 ± 0.0221, ranking second among all methods and outperforming LapPE, LapPE+RWSE, AllPSE, and GPSE. On MolPCBA, GPS+EntSE achieves an AP of 0.2928 ± 0.0011, also ranking second and approaching the best result obtained by GPS+LapPE. These results suggest that our proposed EntSE and KerSE can provide useful structural information for molecular graph representation learning.

Table 12: Performance comparison of structural encodings on MolHIV and MolPCBA. The best result in each column is highlighted in bold, and the second-best result is underlined.

Method	MolHIV (AUROC ↑)	MolPCBA (AP ↑)
GPS+rand	0.6210 ± 0.0444	0.0753 ± 0.0045
GPS+none	0.7798 ± 0.0077	0.2869 ± 0.0012
GPS+LapPE	0.7736 ± 0.0097	$\mathbf{0.2939 \pm 0.0016}$
GPS+RWSE	$\mathbf{0.7880 \pm 0.0101}$	0.2907 ± 0.0028
GPS+LapPE+RWSE	0.7719 ± 0.0129	0.2854 ± 0.0029
GPS+AllPSE	0.7645 ± 0.0236	0.2826 ± 0.0001
GPS+GPSE	0.7815 ± 0.0133	0.2911 ± 0.0036
GPS+EntSE (ours)	0.7763 ± 0.0174	$\underline{0.2928 \pm 0.0011}$
GPS+KerSE (ours)	$\underline{0.7833 \pm 0.0221}$	0.2888 ± 0.0017

J.6 Detailed Time Analysis

In the runtime comparison, we select three datasets with different sparsity levels: PROTEINS as a sparse dataset, DD as a medium-density dataset, and COLLAB as a dense dataset. Table 13 reports their basic statistics, including the number of graphs, average numbers of nodes and edges, average and maximum degrees, clustering coefficient, maximum diameter.

Table 13: Statistics of graph classification datasets.

Dataset	#Graphs	Avg. $\lvert V\rvert$	Avg. $\lvert E\rvert$	Avg. Degree	Max Degree	Avg. Clustering	Avg. Diameter	Max Diameter
PROTEINS	1,113	39.1	72.8	3.73	25	0.5142	11.56	61
DD	1,178	284.3	715.7	5.03	19	0.4794	19.29	75
COLLAB	5,000	74.5	2,457.8	65.99	491	0.8908	1.86	2

These statistics help explain the preprocessing cost under different graph structures. Taking COLLAB as an example, its maximum diameter is only 2, which means that the ego-subgraph already covers

the entire graph when $k = 2$ and does not further expand for larger k. Therefore, for dense or highly clustered graphs, a small k is usually sufficient to capture the local structural context. This also explains why the preprocessing cost remains acceptable in both time and memory.

As shown in Table 14, we compare the time costs of KP-GNN and our methods during the preprocessing and training stages on three graph classification datasets. During preprocessing, KP-GNN incurs an extremely high time cost, taking more than one day on the COLLAB dataset. In contrast, K-GIN+EntSE and K-GIN+KerSE substantially reduce the preprocessing time. Our methods require only 29s and 31s on PROTEINS, 605s and 614s on DD, and remain within 8k seconds on COLLAB. During training, the time differences among the three methods are relatively small. However, K-GIN+EntSE and K-GIN+KerSE still maintain high efficiency. Overall, these results demonstrate that our structure-encoding-based approach offers a clear advantage in computational efficiency.

Table 14: Preprocessing and training time (in seconds) on PROTEINS, DD, and COLLAB datasets.

Stage	Method	PROTEINS	DD	COLLAB
Preprocess	KP-GNN	119	1933	>> 1 day
	K-GIN+EntSE	**14**	**298**	**826**
	K-GIN+KerSE	**18**	**329**	**854**
Training	KP-GNN	1841	17163	—
	K-GIN+EntSE	1910	**6532**	**19213**
	K-GIN+KerSE	2001	**6290**	**18360**

J.7 Empirical Validation of Entropy Compression

To verify the reasonableness of EntSE's scalar compression on real-world datasets, we computed both the eigenvalue distribution and the entropy for the extracted contexts, and measured the proportion of cases where different distributions yield the same entropy across various datasets, as shown in the table below. We found that such cases are extremely rare. Therefore, the entropy can serve as a direct measure for characterizing the contextual distribution.

Table 15: Collision analysis of entropy compression. A collision pair denotes two extracted contexts with different eigenvalue distributions but the same entropy value.

Dataset	Collision Pairs	Total Pairs	Collision Rate
ZINC	5,587	23,694,672,171	2.36×10^{-7}
QM9	600,397	1,894,773,964,327	3.17×10^{-7}
PEPTIDES-FUNC	2,237,986,137	2,474,675,819,811	9.04×10^{-4}
COLLAB	87,684,579	1,036,323,664	8.46×10^{-2}

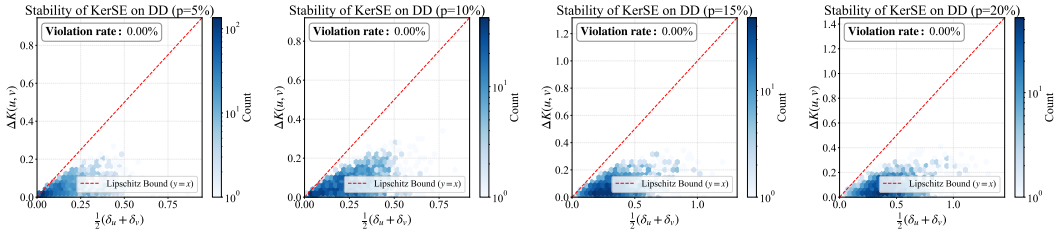
Furthermore, we analyze how the entropy collision rate varies with K on BREC. The results are reported in Table 16. The EntSE exhibits no collisions on the Basic, Extension, and 4-Vertex Condition categories and only a 1% collision rate on Regular graphs. These results indicate that scalar entropy retains strong discriminative information for most of these synthetic graph families. The results are also consistent with the distinguishing results reported in Table 1, supporting our interpretation that the additional distinguishing signal is primarily introduced by EntSE.

Table 16: Entropy collisions on BREC across different values of K. Each entry reports the number of colliding pairs over the total number of pairs, followed by the collision rate.

BREC Category	Number of Pairs	$K = 2$	$K = 4$	$K = 8$
Basic	60	0/60 (0.00%)	0/60 (0.00%)	0/60 (0.00%)
Regular	100	1/100 (1.00%)	1/100 (1.00%)	1/100 (1.00%)
Extension	100	0/100 (0.00%)	0/100 (0.00%)	0/100 (0.00%)
CFI	100	100/100 (100.00%)	94/100 (94.00%)	87/100 (87.00%)
4-Vertex Condition	20	0/20 (0.00%)	0/20 (0.00%)	0/20 (0.00%)
Distance-Regular	20	20/20 (100.00%)	20/20 (100.00%)	20/20 (100.00%)

J.8 Results of Stability

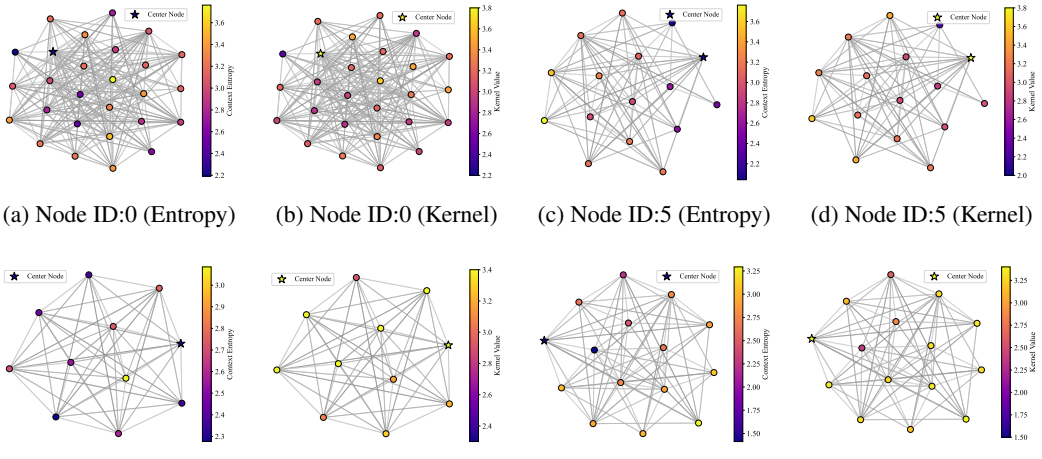
In this subsection, we present additional stability verification results under perturbation ratios of 5%, 10%, 15%, and 20% as Figure 5. The y-axis measures the change in kernel-based similarity, i.e., $y = \Delta K(u, v) = \left|\mathrm{KerSE}(u, v) - \mathrm{KerSE}(\hat{u}, \hat{v})\right|$, and the x-axis corresponds to the average entropy variation of the two nodes, i.e., $x = \frac{1}{2}\left(|\delta_u| + |\delta_v|\right)$. Each point represents a sampled node pair (u, v), with the color intensity reflecting sample density. We further report the *violation rate*, defined as the fraction of node pairs whose kernel variation exceeds the entropy variation. This indicates that the variation of KerSE is consistently smaller than the corresponding entropy variation, suggesting that kernel-based similarity effectively mitigates the amplification of structural perturbations. Overall, these results demonstrate that the KerSE exhibits strong robustness and stability even under substantial structural noise.



(a) $p = 5\%$ (b) $p = 10\%$ (c) $p = 15\%$ (d) $p = 20\%$

Figure 5: Stability of KerSE on DD under different perturbation ratios.

Visualization of Node Representations. In addition, to further investigate why the stability of KerSE leads to performance improvements, we take the DD dataset as a case study. We visualize the context distributions of selected nodes from the same graph in the DD dataset. Specifically, we sample nodes with IDs {0, 5, 10, 15} as the central node, and for each center node, we plot the context representations of the center and its 14 neighboring nodes. As shown in Figure 6, the entropy-based representations exhibit larger scale variation across these nodes, reflected by more pronounced color differences among neighbors. In contrast, the kernel-based representations appear more concentrated, indicating reduced inter-node discrepancy. This observation supports our claim that the reproducing kernel smooths and normalizes structural context, thereby enabling more stable message passing in GNNs. It also helps explain why KerSE outperforms EntSE on certain datasets.



(a) Node ID:0 (Entropy) (b) Node ID:0 (Kernel) (c) Node ID:5 (Entropy) (d) Node ID:5 (Kernel)

(e) Node ID:10 (Entropy) (f) Node ID:10 (Kernel) (g) Node ID:15 (Entropy) (h) Node ID:15 (Kernel)

Figure 6: Additional visualizations of entropy-based and kernel-based node context distributions.

Table 17: Accuracy (%) on synthetic datasets with varying context size K.

Method	EXP (ACC)				SR (ACC)				CSL (ACC)			
	K=1	K=2	K=3	K=4	K=1	K=2	K=3	K=4	K=1	K=2	K=3	K=4
K-GIN	50	50	100	100	6.67	6.67	6.67	6.67	12	22.7	42	62.7
K-GIN+EntSE	**50**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**22**	**52**	**90**	**100**
K-GIN+KerSE	**50**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**12**	**32**	**62**	**100**

J.9 Impact of Context Size on Expressive Power and Graph Classification

We also conduct an analysis to investigate how varying the context size k influences the expressive power of the models. Specifically, we vary k from $\{1, 2, 3, 4\}$ and report the classification accuracy on EXP, SR, and CSL datasets. We select the K-GIN backbone, and the results are presented in Table 17. Overall, both K-GIN+EntSE and K-GIN+KerSE consistently improve upon the original backbone for all choices of k. As k increases, the models become progressively more expressive, and when $k = 4$, both variants achieve perfect accuracy on all three datasets. Notably, even with $k = 1$, our models can already distinguish all non-isomorphic graphs on SR, further highlighting the effectiveness of von Neumann entropy as a strong structural signal.

Table 18: Performance comparison with varying context size k.

Metric	Method	PROTEINS			DD		
		$k=2$	$k=3$	$k=4$	$k=2$	$k=3$	$k=4$
Avg. ego-size	–	9.03	15.15	21.52	13.71	27.34	46.68
Accuracy	GNN-AK+	76.5±3.4	76.0±3.9	75.5±4.8	OOM	OOM	OOM
	KP-GIN	75.9±4.2	75.3±4.7	75.8±5.3	78.6±2.9	78.0±4.4	**80.0±5.3**
	K-GIN+EntSE	**76.8±3.4**	**76.6±5.2**	**77.0±3.1**	79.7±3.0	79.5±3.5	78.9±3.9
	K-GIN+KerSE	76.7±6.5	75.8±4.7	75.9±4.9	**80.4±4.9**	**79.7±5.2**	79.6±5.6
Training Time (s/epoch)	GNN-AK+	0.789	0.791	0.786	OOM	OOM	OOM
	KP-GIN	0.461	0.646	0.589	**0.944**	**1.037**	**1.283**
	K-GIN+EntSE	**0.421**	**0.468**	0.501	1.176	1.237	1.561
	K-GIN+KerSE	0.447	0.482	**0.494**	1.094	1.312	1.358

Furthermore, we study the effect of the context size on graph classification by varying $k \in \{2, 3, 4\}$ on the DD and PROTEINS datasets, as summarized in Table 18. Overall, both K-GIN+EntSE and K-GIN+KerSE present strong and stable performance across different k. On the PROTEINS dataset, K-GIN+EntSE consistently achieves the best accuracy for all three settings, indicating that EntSE is robust to the choice of context size. On the DD dataset, K-GIN+KerSE attains the highest accuracy for all k, outperforming both the KP-GIN baseline and K-GIN+EntSE, which highlights the advantage of kernelized relational encoding on structurally complex graphs. In terms of efficiency, the GNN-AK+ baseline runs out of memory on DD for all k, whereas our methods remain feasible and achieve better accuracy, demonstrating a favorable trade-off between effectiveness and scalability.

We also evaluate how the preprocessing time varies with the contextual radius k. We select the datasets cover substantially different structural regimes, ranging from relatively small and sparse molecular graphs to large graphs and highly dense collaboration networks.

Table 19: Preprocessing time (in seconds) across datasets for different values of K.

Dataset	$K=1$	$K=2$	$K=3$	$K=4$
PROTEINS	9.535	15.075	21.250	27.663
DD	71.390	123.876	195.750	308.465
COLLAB	372.990	867.230	1378.580	1886.330

As expected, the preprocessing time increases with k because larger radii produce larger local contexts. Nevertheless, context caching substantially reduces redundant entropy computation, particularly when repeated local structures occur.

J.10 Analysis on Context Construction Strategy

In this section, we compare the expressiveness of K-GIN+EntSE and K-GIN+KerSE under different context construction strategies, including shortest-path distance (SPD) and random walk (RW). As shown in Table 20, both variants achieve perfect accuracy on EXP and SR regardless of the context type, indicating that their expressiveness on these datasets is insensitive to the specific context

construction. Additionally, using the SPD strategy to construct the context also achieves 100% accuracy on the CSL dataset.

Table 20: Accuracy (%) with different context construction strategies.

Method	EXP		SR		CSL	
	SPD	RW	SPD	RW	SPD	RW
K-GIN+EntSE	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0
K-GIN+KerSE	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	62.00 ± 0.0

Table 21: Comparison of K-GIN+KerSE and K-GIN+KerSE without sampling on benchmark datasets.

Metric	Method	MUTAG	PTC	PROTEINS	IMDBBINARY	DD
Test Accuracy	K-GIN+KerSE	**92.8±7.0**	**67.6±6.4**	**76.7±6.5**	**76.6±4.1**	**80.4±4.9**
	K-GIN+KerSE w/o sample	92.2±5.4	65.9±8.0	76.3±5.3	76.2±4.0	79.0±3.5
Time (s)	K-GIN+KerSE	**0.179**	**0.265**	**1.014**	**0.889**	**4.919**
	K-GIN+KerSE w/o sample	0.185	0.325	1.170	1.015	126.980

J.11 Analysis on Neighbor Sampling Mechanism.

As our KerSE method employs an efficient neighbor sampling mechanism, we perform a detailed ablation study on this component, as presented in Table 21. We evaluate the test accuracy and the inference time, respectively. Removing neighbor sampling from K-GIN+KerSE leads to lower accuracy on most datasets. More importantly, incorporating sampling yields a substantial gain in computational efficiency while maintaining strong predictive performance. For example, on DD, the runtime is reduced from 126.980s to 4.919s. Overall, these results validate the proposed sampling mechanism as a key component of KerSE, enabling a favorable trade-off between effectiveness and efficiency, especially on larger graphs.

To validate the proposed sampling strategy, we compare several neighbor sampling variants on graph classification benchmarks, including sampling high-entropy neighbors, low-entropy neighbors, and uniform random sampling. As shown in Table 22, sampling high-entropy neighbors consistently yields the best performance on both datasets, with particularly notable gains on DD. This suggests that high-entropy nodes tend to provide richer and more discriminative structural signals. By prioritizing such neighbors, the model focuses message passing on the most informative local contexts, leading to improved downstream performance.

Table 22: Performance of different sampling strategies.

Dataset	High-entropy	Low-entropy	Random Sampling
IMDB-BINARY	**76.6 ± 4.1**	75.4 ± 3.3	75.1 ± 5.0
DD	**80.4 ± 4.9**	78.7 ± 5.3	77.3 ± 6.9

Furthermore, we conduct a sensitivity analysis on two representative benchmarks to study the impact of the number of sampled neighbors m. The results on DD and IMDB-BINARY are reported in Table 23. Overall, the performance remains stable as m varies from 2 to 8, indicating that our method is not sensitive to the sampling size. In particular, $m = 4$ achieves the best accuracy on both datasets, while larger m does not lead to consistent further improvements. These results suggest that sampling only a small number of neighbors is sufficient to capture the key structural information while maintaining strong accuracy and efficiency.

J.12 Comparisons with Different Entropy Choices

As mentioned earlier in Appendix D, we provide theoretical evidence that von Neumann entropy is a more suitable structural descriptor than structural Shannon entropy. In this section, we further validate the practical benefits of using von Neumann entropy through empirical studies. First, on the EXP, CSL, and SR benchmarks, Table 24 shows that the variant using von Neumann entropy consistently outperforms the one using structural Shannon entropy, indicating stronger expressive power under our framework. Second, we compare the two entropy choices on graph classification tasks. As shown in Table 25, both von Neumann entropy and structural Shannon entropy serve as effective components within our framework and achieve comparable performance across datasets.

Table 23: Accuracy (%) with different sampling number m.

Dataset	m=2	m=4	m=6	m=8
DD	79.3±5.6	79.6±3.3	79.5±3.3	78.6±5.4
IMDB-BINARY	75.3±2.9	76.8±2.9	75.9±3.3	76.4±3.8

Notably, von Neumann entropy exhibits a slight advantage in terms of higher mean accuracy and often smaller variance, suggesting more reliable performance in practice.

Table 24: Comparison of different entropy on expressive power.

Method	EXP	CSL	SR
Structural Shannon Entropy	100.0	26.67	96.7
von Neumann entropy	**100.0**	**100.0**	**100.0**

J.13 Comparisons with Different Kernel Functions

For the KerSE method, we adopt a reproducing kernel for the kernel aggregation module. To validate its effectiveness, we compare it against the context Jensen-Shannon kernel [3], a widely used baseline for measuring distributional similarity. The comparison results are summarized in Table 25. We can observe that the reproducing kernel consistently outperforms the Jensen-Shannon kernel across all five graph classification datasets. This result indicates that the reproducing kernel more effectively alleviates the discrepancy between context entropies rooted at neighboring nodes, leading to more stable and informative message aggregation, which is consistent with our theoretical analysis.

Table 25: Comparison of different entropy and kernel strategies on five datasets.

Category	Method	MUTAG	D&D	PTC-MR	PROTEINS	IMDBBINARY
Entropy	Structural Shannon Entropy	92.8±4.6	79.0±3.5	66.8±9.4	76.4±3.4	75.5±3.3
	von Neumann entropy	**93.9±5.5**	**79.7±3.0**	**67.4±9.4**	**77.0±3.1**	**76.3±4.3**
Kernel	Jensen Shannon kernel	92.8±7.0	79.5±5.8	65.6±6.9	76.2±4.0	75.9±4.3
	Reproduce kernel	**92.8±4.6**	**80.4±4.9**	**65.9±8.0**	**76.7±6.5**	**76.6±4.1**

J.14 Limitation and Future Work

The GNN field is currently undergoing a paradigm shift from solely pursuing higher expressiveness to balancing expressiveness and generalization. Compared with existing methods that enhance expressiveness, our EntSE and KerSE can be viewed as a lightweight inductive bias. From the perspective of VC dimension, the model parameters barely increase, so the set of representable functions is hardly enlarged, and the increase in VC dimension is minimal. A limitation of our work is that we only provide a preliminary generalization argument. A rigorous generalization bound analysis for expressiveness has not yet been established. A promising future direction is to establish a theoretical framework that analyzes the extent to which the improvement in expressiveness is beneficial. Our lightweight design provides a valuable starting point for exploring this question.

J.15 Impact Statements

This paper proposes a structural encoding framework based on contextual distributions to improve the expressive power and structural inductive bias of graph representation learning models, including Message Passing Neural Networks and Graph Transformers. There are no specific negative societal consequences or ethical concerns associated with this theoretical advancement that we feel must be highlighted here.

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 main made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We have discussed the limitations in Section J.14

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: We provide proof of Proposition 4.2 in the Appendix B, the proof of Theorem 4.4 in the Appendix C and the proof of Theorem 4.5 in the Appendix G. The remaining corollaries are self-evident and do not require formal proof.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide network architecture and implementation details in the Appendix J.2.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We provide the code in the experimental section.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide the training and test details in the Appendix J.2

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report standard deviations over multiple runs for all experiments in the expressiveness analysis, graph classification, and graph regression tasks, as detailed in the experimental section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide the details of compute resources in the Appendix J.2.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper discuss both potential positive societal impacts and negative societal impacts of the work performed in the Appendix J.15.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cited the original papers that produce the code package or dataset.

    Guidelines:

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

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not contain new assets.

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs.

Guidelines:

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