Topology-Informed Graph Transformer

Yun Young Choi [† 1] Sun Woo Park [2] Minho Lee [3] Youngho Woo [1]

Editors: *S. Vadgama, E.J. Bekkers, A. Pouplin, S.O. Kaba, H. Lawrence, R. Walters, T. Emerson, H. Kvinge, J.M. Tomczak, S. Jegelka*

Abstract

Transformers, through their self-attention mechanisms, have revolutionized performance in Natural Language Processing and Vision. Recently, there has been increasing interest in integrating Transformers with Graph Neural Networks (GNNs) to enhance analyzing geometric properties of graphs by employing global attention mechanisms. A key challenge in improving graph transformers is enhancing their ability to distinguish between isomorphic graphs, which can potentially boost their predictive performance. To address this challenge, we introduce 'Topology-Informed Graph Transformer (TIGT)', a novel transformer enhancing both discriminative power in detecting graph isomorphisms and the overall performance of Graph Transformers. TIGT consists of four components: (1) a topological positional embedding layer using non-isomorphic universal covers based on cyclic subgraphs of graphs to ensure unique graph representation, (2) a dual-path message-passing layer to explicitly encode topological characteristics throughout the encoder layers, (3) a global attention mechanism, and (4) a graph information layer to recalibrate channel-wise graph features for improved feature representation. TIGT outperforms previous Graph Transformers in classifying synthetic dataset aimed at distinguishing isomorphism classes of graphs. Additionally, mathematical analysis and empirical evaluations highlight our model's competitive edge over state-of-the-art Graph Transformers across various benchmark datasets.

[1]National Institute for Mathematical Sciences [2]University of Wisconsin-Madison [3]VOLTWIN Inc.. Correspondence to: Yun Young Choi <choi930121@gmail.com>.

1. INTRODUCTION

Transformers have achieved remarkable success in domains such as Natural Language Processing (Vaswani et al., 2023) and Computer Vision (Dosovitskiy et al., 2021). Motivated by their prowess, researchers have applied them to the field of Graph Neural Networks (GNNs). They aimed to surmount the limitations of Message-Passing Neural Networks (MPNNs), a subset of GNNs, by attempting to address challenges such as over-smoothing (Oono & Suzuki, 2021), over-squashing (Alon & Yahav, 2021), and restricted expressive power (Xu et al., 2019; Morris et al., 2021). An exemplary application of the integration of Transformers into GNNs is the Graph Transformer. One way to achieve this integration is by applying multi-head attention mechanism of Transformers to each node of a given graph dataset. Such a technique treats the set of all nodes as being fully interconnected or connected by edges. This approach, however, often has limited capabilities in guaranteeing a strong inductive bias [1], hence making it prone to overfitting. To address this drawback, previous studies focused on devising new implementations to blend Graph Transformers with other deep learning techniques including MPNNs, thereby yielding promising empirical results (Yang et al., 2021; Ying et al., 2021; Dwivedi & Bresson, 2021; Chen et al., 2022; Hussain et al., 2022; Zhang et al., 2023; Ma et al., 2023; Rampášek et al., 2023; Kong et al., 2023; Zhang et al., 2022).

While Graph Transformers boast considerable advancements, enhancing them by strengthening their discriminative powers in distinguishing isomorphism classes of graphs still remains a challenge, an approach that can potentially boost their performance in predicting various properties of graph datasets. For instance, studies based on MPNN have enhanced node attributes by utilizing high-dimensional complexes, persistent homological techniques, and recurring subgraph structures (Carrière et al., 2020; Bodnar et al., 2021b; Bouritsas et al., 2021; Wijesinghe & Wang, 2021;

---

[1]For example, unless trained over large scale data, transformers may lack inherent inductive biases in modeling permutation and translational invariance, hierarchical structures, and local data structures (Hahn, 2020; Dosovitskiy et al., 2021; Xu et al., 2021)

Bevilacqua et al., 2022; Park et al., 2022; Horn et al., 2022; Choi et al., 2023; Li et al., 2023; Choi et al., 2024; Zhou et al., 2024). Other approaches addressing these limitations include incorporating positional encoding grounded in random walk strategies, Laplacian Positional Encoding (PE), node degree centrality, and shortest path distance. Furthermore, structure encoding based on substructure similarity has been introduced to amplify the inductive biases inherent in the Transformer.

This paper introduces a Topology-Informed Graph Transformer (TIGT), which embeds topological inductive biases to augment the model's expressive power and predictive efficacy. Prior to the Transformer layer, TIGT augments each node attribute with a topological positional embedding layer based on the differences of universal covers (or unfolding trees) obtained from the original graph and collections of unions of its cyclic subgraphs. These new topological biases allow TIGT to encapsulate the first homological invariants (or cyclic structures)[2] of the given graph datasets. In combination with the novel positional embedding layer, we explicitly encode cyclic subgraphs in the dual-path message passing layer and incorporate channel-wise graph information in the graph information layer. These are combined with global attention across all Graph Transformer layers, a design of which was inspired from (Choi et al., 2023) and (Rampášek et al., 2023). As a result, the TIGT layer can concatenate hidden representations from the dual-path message passing layer, thereby combining information of the original graph dataset and its cyclic subgraphs, global attention layer, and graph information layer. This allows TIGT to preserve both topological information and graph-level information in each layer. Specifically, the dual-path message passing layer enables TIGT to overcome the limitations of positional encoding and structural encoding to increase expressive power when the number of layers increases. We justify the proposed model's expressive power based on the theory of covering spaces. Furthermore, we perform experiments on synthetic datasets to distinguish isomorphism classes of graphs. We also evaluate the proposed model on benchmark datasets to demonstrate its state-of-the-art predictive performance.

Our main contributions can be summarized as follows. **(i)**. We provide a theoretical justification for the expressive power of TIGT. We compare TIGT with other Graph Transformers using a number of results from geometry and probability theory. These tools include the theory of covering

---

[2]Given a graph $G = (V, E)$, the first homology group of G is a free abelian group on a cycle basis of G. A cycle basis of G is a minimal set of cyclic subgraphs of G such that all cyclic subgraphs of G can be expressed as unions or complements of elements in the cycle basis. For a more rigorous treatment on the construction of homology groups of topological spaces in general, we refer to Chapter 2 of (Hatcher, 2002).

spaces (Hatcher, 2002, Chapter 1.3), Euler characteristic formulas of graphs and their subgraphs (Hatcher, 2002, Chapter 2.2), and the geometric rate of convergence of Markov operators over finite graphs to stationary distributions (Lovasz, 1993). **(ii)** We propose a novel positional embedding layer based on the MPNNs and simple architectures to enrich topological information in each Graph Transformer layer. **(iii)**. We demonstrate superior performance in processing synthetic datasets to assess the expressive power of GNNs. **(iv)** We obtain state-of-the-art or competitive results, especially in large graph-level benchmarks.

2. Preliminary

We first introduce a number of background materials which could be helpful for understanding the newly proposed architecture.

Message passing neural networks. MPNNs have demonstrated proficiency in generating vector representations of graphs by exploiting local information based on node connectivity. This capability is shared with other types of GNNs, such as Graph Convolutional Network (GCN), Graph Attention Network (GAT) (Veličković et al., 2018), Graph Isomorphism Network (GIN) (Xu et al., 2019) and Residual Graph ConvNets (GatedGCN) (Bresson & Laurent, 2018). We use the abbreviation MPNN^l to denote an MPNN that has a composition of l neighborhood aggregating layers. Each l-th layer $H^{(l)}$ of the network constructs hidden node attributes of dimension k_l, denoted as $h_v^{(l)}$, using the following composition of functions:

$$\begin{cases} h_v^{(l)} := \text{COMBINE}^{(l)} \left(h_v^{(l-1)}, \\ \qquad \text{AGGREGATE}_v^{(l)} \left(\left\{ \left\{ h_u^{(l-1)} \mid \begin{smallmatrix} u \in V(G), u \neq v \\ (u,v) \in E(G) \end{smallmatrix} \right\} \right\} \right) \right) \\ h_v^{(0)} := X_v \end{cases}$$

where X_v is the initial node attribute at v. Let $M_v^{(l)}$ be the collection of all multisets of k_{l-1}-dimensional real vectors with $\deg v$ elements counting multiplicities. The aggregation function

$$\text{AGGREGATE}_v^{(l)} : M_v^{(l)} \to \mathbb{R}^{k_l'}$$

is a set theoretic function that outputs k_l'-dimensional real vectors, and the combination function

$$\text{COMBINE}^{(l)} : \mathbb{R}^{k_{l-1}+k_l'} \to \mathbb{R}^{k_l}$$

is a set theoretic function combining the attribute h_v^{l-1} and the image of $\text{AGGREGATE}_v^{(l)}$.

Let $M^{(L)}$ be the collection of all multisets of k_L-dimensional vectors with $\#V(G)$ elements. Let

$$\text{READOUT} : M^{(L)} \to \mathbb{R}^K$$

be the graph readout function of K-dimensional real vectors defined over the multiset $M^{(L)}$. Then the K-dimensional vector representation of G, denoted as h_G, is given by

$$h_G := \text{READOUT}\left(\{\{h_v^{(l)} \mid v \in V(G)\}\}\right)$$

Clique adjacency matrix The clique adjacency matrix, proposed by (Choi et al., 2023), is a matrix that represents bases of cycles of a graph in a form analogous to the adjacency matrix, enabling its processing within GNNs. Extracting bases of cycles as in (Paton, 1969), removing edges of the original graph not contained in the cycle bases, and substituting cyclic subgraphs in the cycle bases with cliques result in incorporating topological properties equivalent to first homological invariants (or cyclic substructures) of graphs. The set of cyclic subgraphs of G which forms the basis of the cycle space (or the first homology group) of G is defined as the cycle basis $\mathcal{B}_G$. The clique adjacency matrix, A_C, is the adjacency matrix of the union of $\#\mathcal{B}_G$ complete subgraphs, each obtained from adding all possible edges among the set of nodes of each basis element $B \in \mathcal{B}_G$. Explicitly, the matrix $A_C := \{a_{u,v}^C\}_{u,v \in V(G)}$ is given by

$$a_{u,v}^C := \begin{cases} 1 & \text{if } \exists\, B \in \mathcal{B}_G \text{ cyclic s.t. } u,v \in V(B) \\ 0 & \text{otherwise} \end{cases}$$

We note that it is also possible to construct bounded clique adjacency matrices, analogously obtained from sub-bases of cycles comprised of bounded number of nodes.

3. TOPOLOGY-INFORMED GRAPH TRANSFORMER(TIGT)

In this section, we introduce the overall TIGT architecture. The overall architecture of our model is illustrated in Figure 1. Suppose we are given the graph $G := (V_G, E_G)$, where V_G is the set of nodes and E_G is the set of edges of G. The graph G can be represented by four types of matrices which are used as inputs of TIGT: (1) a node feature matrix $X \in \mathbb{R}^{n \times k_X}$, (2) an adjacency matrix $A \in \mathbb{R}^{n \times n}$, (3) a clique adjacency matrix $A_c \in \mathbb{R}^{n \times n}$, and (4) an edge feature matrix $E \in \mathbb{R}^{n \times k_E}$. Note that n is the number of nodes, k_X is node feature dimension, and k_E is edge feature dimension. The clique adjacency matrices are obtained using the same procedure outlined in previous research (Choi et al., 2023). For clarity and conciseness in our presentation, we leave the details pertaining to the normalization layer and the residual connection in Appendix C.2. We note that some of the mathematical notations used in explaining the model design and details of model structures conform to those shown in (Rampášek et al., 2023).
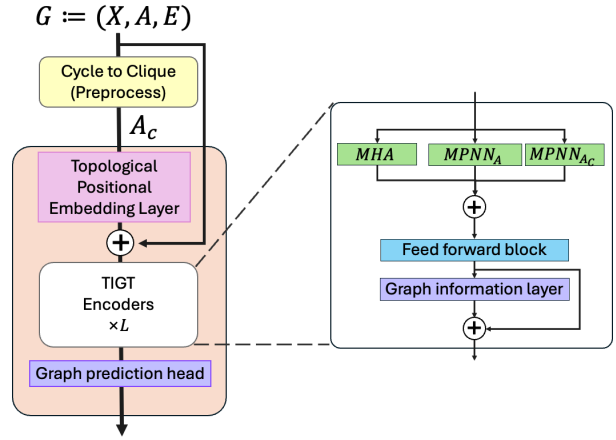

$$G := (X, A, E)$$

Figure 1. Overall Architecture of TIGT.

3.1. Topological positional embedding layer

To adequately capture the nuances of graph structures, most previous work in Graph Transformers uses positional embeddings based on parameters such as node distance, random walk, or structural similarities. Diverging from this typical approach, we propose a novel method for obtaining learnable positional encodings by leveraging MPNNs. This approach aims to enhance their discriminative power with respect to isomorphism classes of graphs, drawing inspiration from Cy2C-GNNs (Choi et al., 2023). First, we use a MPNNs to obtain hidden attributes from the original graph and a new graph structure with clique adjacency matrix as follows:

$$h_A = \text{MPNN}(X, A), \qquad h_A \in \mathbb{R}^{n \times k}$$
$$h_{A_C} = \text{MPNN}(X, A_C), \quad h_{A_C} \in \mathbb{R}^{n \times k}$$
$$h = \begin{bmatrix} h_A & h_{A_C} \end{bmatrix}, \qquad h \in \mathbb{R}^{n \times k \times 2}$$

where X represents the node embedding tensor from the embedding layers, and $[\ ]$ denotes the process of stacking two hidden representations. It's important to note that MPNNs for the adjacency matrix and the clique adjacency matrix share weights. Then the node features are updated along with topological positional attributes, as shown below:

$$X_i^0 = X_i + \text{SUM}(\text{Activation}(h_i \odot \theta_{pe})), \quad X^0 \in \mathbb{R}^{n \times k}$$

where i is the node index of the original graph and $\theta_{pe} \in \mathbb{R}^{1 \times k \times 2}$ represents the learnable parameters that are utilized to integrate features from two universal covers (or unfolding trees) of the two graph structures. The SUM operation performs a sum of the hidden features h_A and h_{A_c} by summing over the last dimensions. For the Activation function, in this study, we use hyperbolic tangent function to bound the value of positional information. Regardless of the presence

or absence of edge attributes, we do not use any existing edge attributes in this layer. The main objective of this layer is to enrich node features by adding topological information obtained from combining the two universal covers. The resulting output X^0 will be subsequently fed into the encoder layers of TIGT.

3.2. Encoder layer of TIGT

The Encoder layer of the TIGT is based on three components: A Dual-path message passing layer, a global attention layer, and a graph information layer. For the input to the Encoder layer, we utilize the transformed input feature X^{l-1} along with A, A_c, and E^{l-1}, where l is encoder layer number in TIGT.

Dual-path MPNNs Hidden representations X^{l-1}, sourced from the preceding layer, are paired with the adjacency matrix and clique adjacency matrix. The paired inputs are then processed through a dual-path message passing layer as follows:

$$X^l_{\text{MPNN},A} = \text{MPNN}_A(X^{l-1}, E^{l-1}, A)$$
$$X^l_{\text{MPNN},A_C} = \text{MPNN}_{A_C}(X^{l-1}, A_C),$$
$$\text{where } X^l_{\text{MPNN},A} \in R^{n \times k} \text{ and } X^l_{\text{MPNN},A_C} \in R^{n \times k}$$

Global attention layer To capture global relationship among nodes of the original graph, we apply the multi-head attention of the vanilla Transformer as follows:

$$X^l_{MHA} = \text{MHA}(X^{l-1}), \quad X^l_{MHA} \in R^{n \times k}$$

where MHA is multi-head attention layer. Then we obtain representation vectors from local neighborhoods, all nodes in graph, and neighborhoods of the same cyclic subgraph. Combining these representations, we obtain the intermediate node representations given by:

$$\bar{X}^l = X^l_{\text{MPNN},A} + X^l_{\text{MPNN},A_C} + X^l_{\text{MHA}}, \ \bar{X}^l \in R^{n \times k}$$

Graph information layer We propose a Graph Information Layer to integrate the pooled graph features with the output of the global attention layer. Inspired by the squeeze-and-excitation block (Hu et al., 2019), this process adaptively recalibrates channel-wise graph features into each node feature as:

$$Y^l_{G,0} = \text{READOUT}\left(\{\bar{X}^l_v \mid v \in V(G)\}\right), X^l_G \in R^{1 \times k}$$
$$Y^l_{G,1} = \text{ReLU}(\text{LIN}_1(Y^l_{G,0})), \qquad Y^l_{G,1} \in R^{1 \times k/N}$$
$$Y^l_{G,2} = \text{Sigmoid}(\text{LIN}_2(Y^l_{G,1})), \qquad Y^l_{G,2} \in R^{1 \times k}$$
$$\bar{X}^l_G = \bar{X}^l \odot Y^l_G, \qquad \bar{X}^l_G \in R^{n \times k}$$

where LIN_1 is linear layer for squeeze feature dimension and LIN_2 is linear layer for excitation feature dimension. Note that N is reduction factor for squeezing feature.

To culminate the process and ensure channel mixing, the features are passed through an MLP layer as follows:

$$X^l = \text{MLP}(\bar{X}^l_G), \quad X^l \in R^{n \times k}$$

3.3. Mathematical background of TIGT

Now that we have introduced the architectural design of TIGT, we focus on elucidating mathematical insights which enable TIGT to exhibit competitive discriminative power in comparison to other state-of-the-art techniques.

Clique adjacency matrix The motivation for utilizing the clique adjacency matrix in implementing TIGT originates from recent work by previous research (Choi et al., 2023), which establishes a mathematical identification of discerning capabilities of GNNs using the theory of covering spaces of graphs. To summarize their work, conventional GNNs represent two graphs G and H, endowed with node feature functions $X_G : G \to \mathbb{R}^k$ and $X_H : G \to \mathbb{R}^k$, as identical vector representations if and only if the universal covers of G and H are isomorphic, and the pullback of node attributes over the universal covers are identical. We note that universal covers of graphs are infinite graphs containing unfolding trees of the graph rooted at a node as subgraphs. In other words, these universal covers do not contain cyclic subgraphs which may be found in the original given graph as subgraphs. Additional measures to further distinguish cyclic structures of graphs are hence required to boost the distinguishing power of GNNs. Among various techniques to represent cyclic subgraphs, we focus on the following two solutions which can be easily implemented. **(1)** Construct clique adjacency matrices A_C, as utilized in the architectural component of TIGT, which transform the geometry of universal covers themselves. **(2)** Impose additional positional encodings, which alter the pullback of node attributes over the universal covers. The distinguishing power of TIGT can be stated as follows, whose proof follows from the results shown in (Choi et al., 2023).

Theorem 3.1. *Suppose G and H are two graphs with the same number of nodes and edges. Suppose that there exists a cyclic subgraph C that is an element of a cycle basis of G such that satisfies the following two conditions: **(1)** C does not contain any proper cyclic subgraphs, and **(2)** any element of a cycle basis of H is not isomorphic to C. Then TIGT can distinguish G and H as non-isomorphic.*

As a corollary of the above theorem, we obtain the following explicit quantification of discriminative power of TIGT in classifying graph isomorphism classes. We leave the details of the proofs of both theorems in Appendix A.1 and A.2.

Theorem 3.2. *There exists a pair of graphs G and H such that TIGT can distinguish them to be non-isomorphic, whereas 3-Weisfeiler-Lehman (3-WL) test cannot.*

Theorem 3.2 hence shows that TIGT has capability to distinguish pairs of graphs which are not distinguishable by algorithms comparable to 3-WL test, such as the generalized distance Weisfeiler-Lehman test (GD-WL) utilizing either shortest path distance (SPD) or resistance distance (RD) (Zhang et al., 2023).

Graph biconnectivity   Given that TIGT is able to distinguish classes of graphs that 3-WL cannot, it is reasonable to ask whether TIGT can capture topological properties of graphs that other state-of-the-art techniques can encapsulate. One of such properties is bi-connectivity of graphs (Zhang et al., 2023; Ma et al., 2023). We recall that a connected graph G is vertex (or edge) biconnected if there exists a vertex v (or an edge e) such that $G \setminus \{v\}$ (or $G \setminus \{e\}$) has more connected components than G. As these state-of-the-art techniques can demonstrate, TIGT as well is capable to distinguish vertex (or edge) bi-connectivity of graphs. The idea of the proof relies on comparing the Euler characteric formula for graphs G and $G \setminus \{v\}$ (or $G \setminus \{e\}$), the specific details of which are provided in Appendix A.3.

Theorem 3.3. *Suppose G and H are two graphs with the same number of nodes, edges, and connected components such that G is vertex (or edge) biconnected, whereas H is not. Then TIGT can distinguish G and H as non-isomorphic graphs.*

In fact, as shown in Appendix C of (Zhang et al., 2023), there are state-of-the-art techniques which are designed to encapsulate cycle structures or subgraph patterns but cannot distinguish biconnectivity of classes of graphs, such as cellular WL (Bodnar et al., 2021a), simplicial WL (Bodnar et al., 2021b), and GNN-AK (Zhao et al., 2022). These results indicate that TIGT can detect both cyclic structures and bi-connectivity of graphs, thereby addressing the topological features the generalized construction of Weisfeiler-Lehman test aims to accomplish, as well as showing capabilities of improving distinguishing powers in comparison to other pre-existing techniques.

Positional encoding   As aforementioned, the method of imposing additional positional encodings to graph attributes can allow neural networks to distinguish cyclic structures, as shown in various types of Graph Transformers (Ma et al., 2023; Rampášek et al., 2023; Ying et al., 2021). One drawback, however, is that these encodings may not be effective enough to represent classes of topologically non-isomorphic graphs as distinct vectors which are not similar to one another. We present a heuristic argument of the drawback mentioned above. Suppose we utilize a Transformer with finitely many layers to obtain vector representations $X_G^*, X_H^* \in \mathbb{R}^m$ of two graphs G and H with node attribute matrices $X_G, X_H \in \mathbb{R}^{n \times k}$ and positional encoding matrices $POS_G, POS_H \in \mathbb{R}^{n \times k'}$. Suppose further that all layers of the Transformer are comprised of compositions

of Lipschitz continuous functions. This implies that the Transformer can be regarded as a Lipschitz continuous function from $\mathbb{R}^{n \times (k+k')}$ to $\mathbb{R}^m$. Hence, for any $\epsilon > 0$ such that $\|[X_G|POS_G] - [X_H|POS_H(v)]\| < \epsilon$, there exists a fixed constant $K > 0$ such that $\|X_G^* - X_H^*\| < K\epsilon$. This suggests that if the node attributes and the positional encodings of non-isomorphic classes of graphs are similar to one another, say within ϵ-error, then such Transformers will represent these graphs as similar vectors, say within $K\epsilon$-error. Hence, it is crucial to determine whether the given positional encodings effectively perturbs the node attributes to an extent that results in obtaining markedly different vector representations.

In relation to the above observation, we show that the relative random walk probabilities positional encoding (RRWP) suggested in (Ma et al., 2023) may not effectively model K steps of random walks on graphs G containing a cyclic subgraph with odd number of nodes, and may not be distinguishable by 1-WL as K grows arbitrarily large. The proof of the theorem is in Appendix A.4.

Theorem 3.4. *Let $\mathcal{G}$ be any collections of graphs whose elements satisfy the following three conditions: (1) All graphs $G \in \mathcal{G}$ share the same number of nodes and edges, (2) any $G \in \mathcal{G}$ contains a cyclic subgraph with odd number of nodes, and (3) for any number $d \geq 1$, all the graphs $G \in \mathcal{G}$ have identical number of nodes whose degree is equal to d. Fix an integer K, and suppose the node indices for $G \in \mathcal{G}$ are ordered based on its increasing degrees. Let $\mathbf{P}$ be the RRWP positional encoding associated to G defined as $\mathbf{P}_{i,j} := [\boldsymbol{I}, \boldsymbol{M}, \boldsymbol{M}^2, \cdots, \boldsymbol{M}^{K-1}]_{i,j} \in \mathbb{R}^K$, where $\boldsymbol{M} := \boldsymbol{D}^{-1}\boldsymbol{A}$ with $\boldsymbol{A}$ being the adjacency matrix of G, and $\boldsymbol{D}$ the diagonal matrix comprised of node degrees of G. Then there exists a unique vector $\pi \in \mathbb{R}^n$ independent of the choice of elements in $\mathcal{G}$ and a number $0 < \gamma < 1$ such that for any $0 \leq l \leq K-1$, we have $\max_{(i,j)} \|\boldsymbol{M}_{i,j}^l - \pi_j\| < \gamma^l$.*

In particular, the theorem states that the positional encodings which are intended to model K steps of random walks converge at a geometric rate to a fixed encoding $\pi \in \mathbb{R}^K$ regardless of the choice of non-isomorphism classes of graphs $G \in \mathcal{G}$. Hence, such choices of positional encodings may not be effective enough to represent differences in topological structures among such graphs as differences in their vector representations.

4. EXPERIMENTS

With the theoretical analysis of TIGT in place, we present the empirical effectiveness of TIGT in discerning various properties of graph datasets.

Dataset   To analyze the effectiveness of TIGT compared to other models in terms of expressive powers, we experiment on the Circular Skip Link(CSL) dataset (Murphy

et al., 2019). CSL dataset comprises of graphs that have different skip lengths $R \in \{2, 3, 4, 5, 6, 9, 11, 12, 13, 16\}$ with 41 nodes that have the same features. Further, we utilize well-known graph-level benchmark datasets to evaluate proposed models compared to other models. We leverage five datasets from the "Benchmarking GNN" studies: MNIST, CIFAR10, PATTERN, and CLUSTER, adopting the same experimental settings as prior research (Dwivedi et al., 2022). Additionally, we use two datasets from the "Long-Range Graph Benchmark" (Dwivedi et al., 2023): Peptides-func and Peptides-struct. Lastly, to further verify the effectiveness of the proposed model on large datasets, we perform experiments on ZINC full dataset (Irwin et al., 2012), which is the full version of the ZINC dataset with 250K graphs and PCQM4Mv2 dataset (Hu et al., 2020) which is large-scale graph regression benchmark with 3.7M graphs. These benchmark encompass binary classification, multi-label classification, and regression tasks across a diverse range of domain characteristics. The details of the aforementioned datasets are summarized in Appendix C.1.

Models   To evaluate the discriminative power of TIGT, we compare a set of previous researches related to expressive power of GNNs on CSL dataset such as Graph Transformers (GraphGPS (Rampášek et al., 2023), GRIT (Ma et al., 2023)) and other message-passing neural networks (GCN (Kipf & Welling, 2017), GIN, Relational Pooling GIN(RP-GIN) (Murphy et al., 2019), Cy2C-GNNs (Choi et al., 2023)). We compare our approach on well-known benchmark datasets to test graph-level tasks with the latest SOTA techniques, widely adopted MPNNs models, and various Graph Transformer-based studies: GRIT (Ma et al., 2023), GraphGPS (Rampášek et al., 2023)), GCN (Kipf & Welling, 2017), GIN (Xu et al., 2019), its variant with edge-features (Hu et al., 2020), GAT (Veličković et al., 2018), GatedGCN (Bresson & Laurent, 2018), GatedGCN-LSPE (Dwivedi et al., 2022), PNA (Corso et al., 2020), Graphormer (Ying et al., 2021), K-Subgraph SAT (Chen et al., 2022), EGT (Hussain et al., 2022), SAN (Kreuzer et al., 2021), Graphormer-URPE (Luo et al., 2022), Graphormer-GD (Zhang et al., 2023), DGN (Beaini et al., 2021), GSN (Bouritsas et al., 2021), CIN (Bodnar et al., 2021b), CRaW1 (Tönshoff et al., 2023), and GIN-AK+ (Zhao et al., 2022).

TIGT Setup   For hyperparameters of models on CSL datasets, we fixed the hidden dimension and batch size to 16, and other hyperparameters were configured similarly to the setting designed for the ZINC dataset. For a fair comparison of the other nine benchmark datasets, we ensured that both the hyperparameter settings closely matched those found in the GraphGPS (Rampášek et al., 2023) and GRIT (Ma et al., 2023) studies. The differences in the number of trainable parameters between TIGT and GraphGPS primarily arise from the additional components introduced

to enrich topological information within the Graph Transformer layers. Further details on hyperparameters, such as the number of layers, hidden dimensions, and the specific type of MPNNs, are elaborated upon in the Appendix C.3.

Performance on the CSL dataset   In order to test the expressive power of the proposed model and state-of-the-art Graph Transformers, we evaluated their performance on the synthetic dataset CSL. The test performance metrics are presented in Table 1. Our analysis found that TIGT, GPS with random-walk structural encoding (RWSE), and GPS with RWSE and Laplacian eigenvectors encodings (LapPE) outperformed other models. However, the recent state-of-the-art model, GRIT with Relative Random Walk Probabilities (RRWP), could not distinguish the CSL classes. Interestingly, TIGT demonstrated resilience in maintaining a near 100% performance rate, irrespective of the number of added Graph Transformer layers. This consistent performance can be attributed to TIGT's unique Dual-path message-passing layer, which ceaselessly infuses topological information across various layers. Conversely, other models, which initially derive benefits from unique node attributes facilitated by positional encoding, showed signs of diminishing influence from this attribution as the number of layers grew. Additionally, we compared our findings with those of GAT and Cy2C-GNN models. Consistent with previous studies (Choi et al., 2023), GAT was unable to perform the classification task on the CSL dataset effectively. In the case of the Cy2C-GNN model, while it demonstrated high accuracy in a single-layer configuration, similar to GPS, we observed a decline in classification performance as the number of layers increased.

Results from benchmark datasets   First, we present the test performance on five datasets from Benchmarking GNNs (Dwivedi et al., 2022) in Table 2. The mean and standard deviations are reported over four runs using different random seeds. It is evident from the results that our model ranks either first or second in performance on three benchmark datasets: ZINC, MNIST, and CIFAR10. However, for the synthetic datasets, PATTERN, and CLUSTER, our performance is found to be inferior compared to recent state-of-the-art models but is on par with the GraphGPS model. Next, we further assess the effectiveness of our current model by evaluating its test performance on four datasets from the "Long-Range Graph Benchmark" (Dwivedi et al., 2023), full ZINC dataset (Irwin et al., 2012), and the PCQM4Mv2 dataset (Hu et al., 2020). In large datasets, such as the full version of the ZINC dataset and the PCQM4Mv2 dataset, TIGT consistently outperforms other models. In particular, on the PCQM4Mv2 dataset, our model demonstrated superior performance with fewer parameters compared to state-of-the-art models. In the "Long-Range Graph Benchmark," our model also presents the second-highest performance compared to other models. Through all these experimental

results, it is evident that by enhancing the discriminative power to differentiate isomorphisms of graphs, we can boost the predictive performance of Graph Transformers. This has enabled us to achieve competitive results in GNN research, surpassing even recent state-of-the-art model on several datasets. In a comparative analysis between Cy2C-GNN and TIGT, we observed a significant increase in performance across all datasets with TIGT. This indicates that the topological non-trivial features of graphs are well-reflected in TIGT, allowing for both a theoretical increase in expressive power and improved performance on benchmark datasets.

5. CONCLUSION

In this paper, we introduce TIGT, a novel Graph Transformer designed to enhance the predictive performance and expressive power of Graph Transformers. This enhancement is achieved by incorporating a topological positional embedding layer, a dual-path message passing layer, a global attention layer, and a graph information layer. Notably, our topological positional embedding layer is learnable and leverages MPNNs. It integrates universal covers drawn from the original graph structure and a modified structure enriched with cyclic subgraphs. This integration aids in detecting isomorphism classes of graphs. Throughout its architecture, TIGT encodes cyclic subgraphs at each layer using the dual-path message passing mechanism, ensuring its expressive power to be maintained as layer depth increases. Despite a modest rise in complexity, TIGT showcases superior performance in experiments on the CSL dataset, surpassing the expressive capabilities of previous GNNs and Graph Transformers. Additionally, both mathematical justifications and empirical evaluations underscore our model's competitive advantage over contemporary Graph Transformers across diverse benchmark datasets.

While TIGT can be successfully applied to graph-level tasks, there remain avenues for future exploration. Firstly, the computational complexity is limited to $O(N^2 + N_E + N_C)$ with the number of nodes N, the number of edges N_E and the number of edge in cyclic subgraphs N_C. Especially, due to the implementation of global attention in the Transformer, computational complexity poses challenges that we are keen to address in subsequent research. Moreover, beyond the realm of graph-level tasks, there is potential to broaden the application of TIGT into areas like node classification and link prediction. Integrating the topological characteristics inherent in TIGT with these domains might uncover more profound insights and enhance predictive accuracy.

References

Alon, U. and Yahav, E. On the bottleneck of graph neural networks and its practical implications, 2021.

Beaini, D., Passaro, S., Létourneau, V., Hamilton, W. L., Corso, G., and Liò, P. Directional graph networks, 2021.

Bevilacqua, B., Frasca, F., Lim, D., Srinivasan, B., Cai, C., Balamurugan, G., Bronstein, M. M., and Maron, H. Equivariant subgraph aggregation networks, 2022.

Bodnar, C., Frasca, F., Otter, N., Wang, Y., Liò, P., Montufar, G. F., and Bronstein, M. Weisfeiler and lehman go cellular: Cw networks. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 2625–2640. Curran Associates, Inc., 2021a.

Bodnar, C., Frasca, F., Wang, Y. G., Otter, N., Montúfar, G., Liò, P., and Bronstein, M. Weisfeiler and lehman go topological: Message passing simplicial networks, 2021b.

Bouritsas, G., Frasca, F., Zafeiriou, S., and Bronstein, M. M. Improving graph neural network expressivity via subgraph isomorphism counting, 2021.

Bresson, X. and Laurent, T. Residual gated graph convnets, 2018.

Carrière, M., Chazal, F., Ike, Y., Lacombe, T., Royer, M., and Umeda, Y. Perslay: A neural network layer for persistence diagrams and new graph topological signatures, 2020.

Chen, D., O'Bray, L., and Borgwardt, K. Structure-aware transformer for graph representation learning, 2022.

Choi, Y. Y., Park, S. W., Woo, Y., and Choi, U. J. Cycle to clique (cy2c) graph neural network: A sight to see beyond neighborhood aggregation. In *The Eleventh International Conference on Learning Representations*, 2023.

Choi, Y. Y., Lee, M., Park, S. W., Lee, S., and Ko, J. A gated mlp architecture for learning topological dependencies in spatio-temporal graphs. *arXiv preprint arXiv:2401.15894*, 2024.

Corso, G., Cavalleri, L., Beaini, D., Liò, P., and Veličković, P. Principal neighbourhood aggregation for graph nets, 2020.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

Dwivedi, V. P. and Bresson, X. A generalization of transformer networks to graphs, 2021.

Table 1. Results of graph classification obtained from CSL dataset (Murphy et al., 2019). Note that the methods written in bolded texts indicate the results obtained from our implementations. All results other than the bolded methods are cited from available results obtained from pre-existing publications. Our results are written as "mean ± standard deviation", which are obtained from 4 runs with different random seeds. Highlighted are the top first, second, and third results.

GNNs	GIN	RP-GIN	GCN	Cy2C-GCN-1
	10.0±0.0	37.6±12.9	10.0±0.0	91.3±1.6
GATs	GAT-1	GAT-2	GAT-5	GAT-10
	10.0±0.0	10.0±0.0	10.0±0.0	10.0±0.0
Cy2C-GNNs	Cy2C-GIN-1	Cy2C-GIN-2	Cy2C-GIN-5	Cy2C-GIN-10
	98.33±3.33	46.67±38.20	9.17±5.69	7.49±3.21
GPS	1 layer	2 layers	5 layers	10 layers
	5.0±3.34	6.67±9.43	3.34±3.85	5.0±3.34
GPS+RWSE	1 layer	2 layers	5 layers	10 layers
	88.33±11.90	93.33±11.55	90.00±11.06	75.0±8.66
GPS+LapPE+RWSE	1 layer	2 layers	5 layers	10 layers
	100±0.0	95±10.0	93.33±13.33	86.67±10.89
GRIT+RRWP	1 layer	2 layers	5 layers	10 layers
	10.0±0.0	10.0±0.0	10.0±0.0	10.0±0.0
TIGT	1 layer	2 layers	5 layers	10 layers
	98.33±3.35	100±0.0	100±0.0	100±0.0

Table 2. Graph classification and regression results obtained from five benchmarks from (Dwivedi et al., 2022). Note that N/A indicate the methods which do not report test results on the given graph data set. The methods written in bolded texts indicate the results obtained from our implementations. All results other than the bolded methods are cited from available results obtained from pre-existing publications. Our results are written as "mean ± standard deviation", which are obtained from 4 runs with different random seeds. Highlighted are the top first, second, and third results.

	ZINC	MNIST	CIFAR10	PATTERN	CLUSTER
Model	MAE↓	Accuracy↑	Accuracy↑	Accuracy↑	Accuracy↑
GCN	0.367±0.011	90.705±0.218	55.710±0.381	71.892±0.334	68.498±0.976
GIN	0.526±0.051	96.485±0.252	55.255±1.527	85.387±0.136	64.716±1.553
GAT	0.384±0.007	95.535±0.205	64.223±0.455	78.271±0.186	73.840±0.326
GatedGCN	0.282±0.015	97.340±0.143	67.312±0.311	85.568±0.088	73.840±0.326
GatedGCN+LSPE	0.090±0.001	N/A	N/A	N/A	N/A
PNA	0.188±0.004	97.94±0.12	70.35±0.63	N/A	N/A
DGN	0.168±0.003	N/A	72.838±0.417	86.680±0.034	N/A
GSN	0.101±0.010	N/A	N/A	N/A	N/A
CIN	0.079±0.006	N/A	N/A	N/A	N/A
CRaW1	0.085±0.004	97.944±0.050	69.013±0.259	N/A	N/A
GIN-AK+	0.080±0.001	N/A	72.19±0.13	86.850±0.057	N/A
SAN	0.139±0.006	N/A	N/A	86.581±0.037	76.691±0.65
Graphormer	0.122±0.006	N/A	N/A	N/A	N/A
K-Subgraph SAT	0.094±0.008	N/A	N/A	86.848±0.037	77.856±0.104
EGT	0.108±0.009	98.173±0.087	68.702±0.409	86.821±0.020	79.232±0.348
Graphormer-GD	0.081±0.009	N/A	N/A	N/A	N/A
GPS	0.070±0.004	98.051±0.126	72.298±0.356	86.685±0.059	78.016±0.180
GRIT	0.059±0.002	98.108±0.111	76.468±0.881	87.196±0.076	80.026±0.277
Cy2C-GNNs	0.102±0.002	97.772±0.001	64.285±0.005	86.048±0.005	64.932±0.003
TIGT	0.057±0.002	98.230±0.133	73.955±0.360	86.680±0.056	78.033±0.218

Table 3. Graph-level task results obtained from two long-range graph benchmarks (Dwivedi et al., 2023) , ZINC-full dataset (Irwin et al., 2012) and PCQM4Mv2 (Hu et al., 2020). Note that N/A indicate the methods which do not report test results on the given graph data set. The methods written in bolded texts indicate the results obtained from our implementations. All results other than the bolded methods are cited from available results obtained from pre-existing publications. Our results are written as "mean ± standard deviation", which are obtained from 4 runs with different random seeds. Highlighted are the top first, second, and third results.

| | Long-range graph benchmark | | ZINC-full | | PCQM4Mv2 | | |
| | Peptides-func | Peptides-struct | Model | MAE↓ | Model | MAE(Valid)↓ | # Param |
Model	AP↑	MAE↓					
GCN	0.5930±0.0023	0.3496±0.0013	GCN	0.113±0.002	GCN	0.1379	2.0M
GINE	0.5498±0.0079	0.3547±0.0045	GIN	0.088±0.002	GIN	0.1195	3.8M
GatedGCN	0.5864±0.0035	0.3420±0.0013	GAT	0.111±0.002	GCN-virtual	0.1195	4.9M
GatedGCN+RWSE	0.6069±0.0035	0.3357±0.0006	SignNet	0.024±0.003	GIN-virtual	0.1083	6.7M
Transformer+LapPE	0.6326±0.0126	0.2529±0.016	Graphormer	0.052±0.005	Graphormer	0.0864	48.3M
SAN+LapPE	0.6384±0.0121	0.2683±0.0043	Graphormer-URPE	0.028±0.002	GRPE	0.0890	46.2M
SAN+RWSE	0.6439±0.0075	0.2545±0.0012	Graphormer-GD	0.025±0.004	TokenGT (Lap)	0.0910	48.5M
GPS	0.6535±0.0041	0.2500±0.0012	GPS	N/A	GPS-medium	0.0858	19.4M
GRIT	0.6988±0.0082	0.2460±0.0012	GRIT	0.023±0.001	GRIT	0.0859	16.6M
Cy2C-GNNs	0.5193±0.0025	0.2521±0.0012	**Cy2C-GNNs**	0.042±0.001	**Cy2C-GNNs**	0.0956	4M
TIGT	0.6679±0.0074	0.2485±0.0015	**TIGT**	0.014±0.001	**TIGT**	0.0826	13.0M

Dwivedi, V. P., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. Graph neural networks with learnable structural and positional representations, 2022.

Dwivedi, V. P., Rampášek, L., Galkin, M., Parviz, A., Wolf, G., Luu, A. T., and Beaini, D. Long range graph benchmark, 2023.

Fey, M. and Lenssen, J. E. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Hahn, M. Theoretical Limitations of Self-Attention in Neural Sequence Models. *Transactions of the Association for Computational Linguistics*, 8:156–171, 2020.

Hatcher, A. *Algebraic Topology*. Cambridge University Press, 2002.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Horn, M., Brouwer, E. D., Moor, M., Moreau, Y., Rieck, B., and Borgwardt, K. Topological graph neural networks, 2022.

Hu, J., Shen, L., Albanie, S., Sun, G., and Wu, E. Squeeze-and-excitation networks, 2019.

Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., and Leskovec, J. Strategies for pre-training graph neural networks, 2020.

Hussain, M. S., Zaki, M. J., and Subramanian, D. Global self-attention as a replacement for graph convolution. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, aug 2022. doi: 10.1145/3534678.3539296. URL https://doi.org/10.1145%2F3534678.3539296.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Irwin, J. J., Sterling, T., Mysinger, M. M., Bolstad, E. S., and Coleman, R. G. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768, 2012.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks, 2017.

Kong, K., Chen, J., Kirchenbauer, J., Ni, R., Bruss, C. B., and Goldstein, T. Goat: A global transformer on large-scale graphs. In *International Conference on Machine Learning*, pp. 17375–17390. PMLR, 2023.

Kreuzer, D., Beaini, D., Hamilton, W. L., Létourneau, V., and Tossou, P. Rethinking graph transformers with spectral attention, 2021.

Li, C., Xia, L., Ren, X., Ye, Y., Xu, Y., and Huang, C. Graph transformer for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1680–1689, 2023.

Lovasz, L. Random walks on graphs: a survey. *Combinatorics*, pp. 1–46, 1993.

Luo, S., Li, S., Zheng, S., Liu, T.-Y., Wang, L., and He, D. Your transformer may not be as powerful as you expect, 2022.

Ma, L., Lin, C., Lim, D., Romero-Soriano, A., Dokania, P. K., Coates, M., Torr, P., and Lim, S.-N. Graph inductive biases in transformers without message passing, 2023.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. Weisfeiler and leman go neural: Higher-order graph neural networks, 2021.

Murphy, R., Srinivasan, B., Rao, V., and Ribeiro, B. Relational pooling for graph representations. In *International Conference on Machine Learning*, pp. 4663–4673. PMLR, 2019.

Oono, K. and Suzuki, T. Graph neural networks exponentially lose expressive power for node classification, 2021.

Park, S. W., Choi, Y. Y., Joe, D., Choi, U. J., and Woo, Y. The pwlr graph representation: A persistent weisfeiler-lehman scheme with random walks for graph classification. In *Topological, Algebraic and Geometric Learning Workshops 2022*, pp. 287–297. PMLR, 2022.

Paton, K. An algorithm for finding a fundamental set of cycles of a graph. *Communications of the ACM*, 12(9): 514–518, 1969.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a general, powerful, scalable graph transformer, 2023.

Tönshoff, J., Ritzert, M., Wolf, H., and Grohe, M. Walking out of the weisfeiler leman hierarchy: Graph learning beyond message passing, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need, 2023.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks, 2018.

Wijesinghe, A. and Wang, Q. A new perspective on" how graph neural networks go beyond weisfeiler-lehman?". In *International Conference on Learning Representations*, 2021.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks?, 2019.

Xu, Y., Zhang, Q., Zhang, J., and Tao, D. Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in Neural Information Processing Systems*, 34, 2021.

Yang, J., Liu, Z., Xiao, S., Li, C., Lian, D., Agrawal, S., Singh, A., Sun, G., and Xie, X. Graphformers: Gnn-nested transformers for representation learning on textual graph, 2021.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. Do transformers really perform bad for graph representation?, 2021.

Zhang, B., Luo, S., Wang, L., and He, D. Rethinking the expressive power of gnns via graph biconnectivity, 2023.

Zhang, Z., Liu, Q., Hu, Q., and Lee, C.-K. Hierarchical graph transformer with adaptive node sampling. *Advances in Neural Information Processing Systems*, 35: 21171–21183, 2022.

Zhao, L., Jin, W., Akoglu, L., and Shah, N. From stars to subgraphs: Uplifting any gnn with local structure awareness, 2022.

Zhou, C., Yu, R., and Wang, Y. On the theoretical expressive power and the design space of higher-order graph transformers. In *International Conference on Artificial Intelligence and Statistics*, pp. 2179–2187. PMLR, 2024.

A. Mathematical Proofs

This subsection focuses on listing the mathematical background required for proving a series of theorems outlined in the main text of the paper. Throughout this subsection, we regard a graph $G := (V, E)$ as a 1-dimensional topological space endowed with the closure-finiteness weak topology (CW topology), the details of which are written in (Hatcher, 2002)[Chapter 0, Appendix]. In particular, we may regard G as a 1-dimensional CW complex, the set of nodes of G corresponds to the 0-skeleton of G, and the set of edges of G corresponds to the 1-skeleton of G.

By regarding G as a 1-dimensional CW complex, we are able to reinterpret the infinite unfolding tree of G rooted at any choice of a node $v \in G$ as a contractible infinite 1-dimensional CW complex, also known as the universal cover of G.

Definition A.1. Given any topological space X, the universal cover $\pi_X : \tilde{X} \to X$ is a contractible topological space such that for any point $x \in X$, there exists an open neighborhood U containing x such that $\pi_X^{-1}(U)$ is a disjoint union of open neighborhoods, each of which is homeomorphic to U.

A.1. Proof of Theorem 3.1

The proof follows immediately from the fact that TIGT utilizes clique adjacency matrix A_C (or bounded clique adjacency matrix), whose mathematical importance was explored in Theorem 3.3, Lemma 4.1, and Theorem 4.3 of (Choi et al., 2023). We provide an exposition of the key ideas of the proof of the above three theorems here.

Let G and H be two graphs endowed with node attribute functions $f_G : V(G) \to \mathbb{R}^k$ and $f_H : V(H) \to \mathbb{R}^k$. Theorem 3.3 of (Choi et al., 2023) implies that conventional GNNs can represent two graphs G and H as identical vector representations if and only if the following two conditions hold:

- There exists an isomorphism $\varphi : \tilde{G} \to \tilde{H}$ between two universal covers of G and H.

- There exists an equality of pullback of node attributes $f_G \circ \pi_G = f_H \circ \pi_H \circ \varphi$.

In particular, even if G and H have different cycle bases whose elements consist of cyclic subgraphs not containing any other proper cyclic subgraphs, if the universal covers of G and H are isomorphic, then conventional GNNs cannot distinguish G and H as non-isomorphic.

To address this problem, one can include additional edges to cyclic subgraphs of G and H to alter universal covers of G and H to be not isomorphic to each other. This is the key insight in Lemma 4.1 of (Choi et al., 2023). Any two cyclic graphs without proper cyclic subgraphs have isomorphic universal covers, both of which are homeomorphic to the real line $\mathbb{R}^1$. however, when two cyclic graphs are transformed into cliques (meaning that all the nodes lying on the cyclic graphs are connected by edges), then as long as the number of nodes forming the cyclic graphs are different, the universal covers of the two cliques are not isomorphic to one another.

The task of adjoining additional edges connecting nodes lying on a cyclic graph is executed by utilizing the clique adjacency matrix A_C, the matrix of which is also constructed in (Choi et al., 2023). Hence, Theorem 4.3 of (Choi et al., 2023) uses Lemma 4.1 to conclude that by utilizing the clique adjacency matrix A_C (or the bounded clique adjacency matrix), one can add suitable edges to cyclic subgraphs of G and H which do not contain any proper cyclic subgraphs, thereby constructing non-isomorphic universal covers of G and H which allow conventional GNNs to represent G and H as non-identical vectors. In a similar vein, TIGT also utilizes clique adjacency matrices A_C as an input data, the data of which allows one to add suitable edges to cyclic subgraphs of any classes of graphs to ensure constructions of their non-isomorphic universal covers.

A.2. Proof of Theorem 3.2

We now prove that TIGT is capable of distinguishing a pair of graphs G and H which are not distinguishable by 3-WL. The graphs of our interest are non-isomorphic families of strongly regular graphs $SR(16, 6, 2, 2)$, in particular the 4×4 rook's graph and the Shrikhande graph. Both graphs are proven to be not distinguishable by 3-Weisfeiler-Lehman test (Bodnar et al., 2021b)[Lemma 28], but possess different cycle bases whose elements comprise of cyclic graphs which does not contain any proper cyclic subgraphs (Bodnar et al., 2021a)[Theorem 16]. Theorem 3.1 hence implies that TIGT is capable of distinguishing the 4×4 rook's graph and the Shrikhande graph.

We note that these types of strongly regular graphs are also utilized to demonstrate the superiority of a proposed GNN to 3-WL test, such as graph inductive bias Transformers (GRIT) (Ma et al., 2023) or cellular Weisfeiler-Lehman test

(CWL) (Bodnar et al., 2021a).

A.3. Proof of Theorem 3.3

Next, we demonstrate that TIGT is also capable of distinguishing biconnectivity of pairs of graphs G and H. Recall that the Euler characteristic formula (Hatcher, 2002)[Theorem 2.44] for graphs imply that

$$\#E(G) - \#V(G) = \# \text{ Connected components of } G - \# \text{ cycle basis of } G$$

where the term "# cycle basis of G" is the number of elements of a cycle basis of G. This number is well-defined regardless of the choice of a cycle basis, because its number is equal to the dimension of the first homology group of G with rational coefficients, one of the topological invariants of G.

Without loss of generality, assume that G is vertex-biconnected whereas H is not. Then there exists a vertex $v \in V(G)$ such that $G \setminus \{v\}$ has more connected components than G and H. This implies that given any choice of bijection $\phi : V(G) \to V(H)$ between the set of nodes of G and H, the graphs $G \setminus \{v\}$ and $H \setminus \phi(\{v\})$ satisfy the following series of equations:

$$\begin{aligned}
& \# \text{ Connected components of } H \setminus \phi(\{v\}) - \# \text{ cycle basis of } H \setminus \phi(\{v\}) \\
&= \#E(H \setminus \phi(\{v\})) - \#V(H \setminus \phi(\{v\})) \\
&= \#E(H) - \#V(H) + 1 \\
&= \#E(G) - \#V(G) + 1 \\
&= \#E(G \setminus \{v\}) - \#V(G \setminus \{v\}) \\
&= \# \text{ Connected components of } G \setminus \{v\} - \# \text{ cycle basis of } G \setminus \{v\}
\end{aligned}$$

By the condition that G is vertex-biconnected whereas H is not, it follows that the number of cycle basis of $G \setminus \{v\}$ and the number of cycle basis of $H \setminus \{\phi(v)\}$ are different. Because the above equations hold for any choice of cycle bases G and H, we can further assume that both cycle bases G and H satisfy the condition that all elements do not contain proper cyclic subgraphs. But because the number of edges and vertices of the two graphs $G \setminus \{v\}$ and $H \setminus \{\phi(v)\}$ are identical, it follows that there exists a number $c > 0$ such that the number of elements of cycle bases of $G \setminus \{v\}$ and $H \setminus \phi(\{v\})$ whose number of nodes is equal to c are different. Hence, the two graphs G and H can be distinguished by TIGT via the utilization of clique adjacency matrices of $G \setminus \{(v)\}$ and $H \setminus \phi(\{v\})$, i.e. applying Theorem 3.1 to two graphs $G \setminus \{(v)\}$ and $H \setminus \phi(\{v\})$.

In fact, the theorem can be generalized to distinguish any pairs of graphs G and H with the same number of edges, nodes, and connected components, whose number of components after removing a single vertex or an edge become different. We omit the proof of the corollary, as the proof is a direct generalization of the proof of Theorem 3.3.

Corollary A.2. *Let G and H be two graphs with the same number of nodes, edges, and connected components. Suppose there exists a pair of nodes $v \in V(G)$ and $w \in V(H)$ (or likewise a pair of edges $e_1 \in E(G)$ and $e_2 \in E(H)$) such that the number of connected components of $G \setminus \{v\}$ and $H \setminus \{w\}$ are different (and likewise for $G \setminus \{e_1\}$ and $H \setminus \{e_2\}$). Then TIGT can distinguish G and H as non-isomorphic graphs.*

A.4. Proof of Theorem 3.4

The idea of the proof follows from focuses on reinterpreting the probability matrix $\mathbf{M} := \mathbf{D}^{-1}\mathbf{A}$ as a Markov chain defined over a graph $G \in \mathcal{G}$.

Let's recall the three conditions applied to the classes of graphs inside our collection $\mathcal{G}$:

- All graphs $G \in \mathcal{G}$ share the same number of nodes and edges

- Any $G \in \mathcal{G}$ contains a cyclic subgraph with odd number of nodes

- For any number $d \geq 1$, all graphs $G \in \mathcal{G}$ have identical number of nodes whose degree is equal to d.

Denote by n the number of nodes of any graph $G \in \mathcal{G}$. The second condition implies that any graph $G \in \mathcal{G}$ is non-bipartite, hence the probability matrix $\mathbf{M}$ is an irreducible aperiodic Markov chain over the graph G. In particular, this shows that the

Table 4. Ablation study to analyze the effectiveness of the component of TIGT on the ZINC dataset. (Dwivedi et al., 2022).

ZINC	MAE↓
TIGT	0.057±0.002
w/o graph information	0.059±0.005
w/o topological positional embedding	0.060±0.003
not share weight in topological positional embedding	0.061±0.003
Transformer → Performer in global attention	0.063±0.001
w/o global attention	0.063±0.003
Tanh → ReLU in topological positional embedding	0.063±0.004
Dual-path MPNNs → Single-path MPNNs	0.064±0.003
Sum → Mean readout in graph information	0.069±0.001
Cy2C-GNNs	0.102±0.002
Cy2C-GNNs(Large)	0.121±0.003
GraphGPS	0.070±0.004

Markov chain $\mathbf{M}$ has a unique stationary distribution $\pi \in \mathbb{R}^n$ such that the component of π at the j-th node of G satisfies

$$\pi_j = \frac{d(j)}{2\#E(G)}$$

where $d(j)$ is the degree of the node j (Lovasz, 1993)[Section 1]. The first condition implies that regardless of the choice of the graph $G \in \mathcal{G}$, the stationary distributions of π obtained from such Markov chains associated to each G are all identical up to re-ordering of node indices based on their node degrees. The geometric ergodicity of Markov chains, as stated in (Lovasz, 1993)[Theorem 5.1, Corollary 5.2], show that for any initial probability distribution $\delta \in \mathbb{R}^n$ over the graph G, there exists a fixed constant $C > 0$ such that for any $l \geq 0$,

$$\max_j |(\delta^T \mathbf{M}^l)_j - \pi_j| < C \times \gamma^l$$

The geometric rate of convergence γ satisfies the inequality $0 < \gamma < 1$. We note that the value of γ is determined from eigenvalues of the matrix $N := D^{-1/2}\mathbf{M}D^{1/2}$, all of whose eigenvalues excluding the largest eigenvalue is known to have absolute values between 0 and 1 for non-bipartite graphs G (Lovasz, 1993)[Section 3]. To obtain the statement of the theorem, we apply above equation with probability distributions δ whose i-th component is 1, and all other components are equal to 0.

B. Abalation study

To understand the significance of each component in our deep learning model, we performed multiple ablation studies using the ZINC dataset (Dwivedi et al., 2022). The results are presented in Table 4. The influence of the graph information and the topological positional embedding layer is relatively marginal compared to other layers. The choice of weight-sharing within the topological positional embedding layer, as well as the selection between the hyperbolic tangent and ReLU activation functions, play a significant role in the model's performance. Likewise, opting for Single-path MPNNs, excluding the adjacency matrix instead of the proposed Dual-path in each TIGT layer, results in a considerable performance drop. Within the graph information layer, it's evident that employing a sum-based readout function, akin to graph pooling, is crucial for extracting comprehensive graph information and ensuring optimal results. Additionally, we experimented with applying the Performer, which utilizes a kernel trick to replace the quadratic complexity of the transformer's global attention with linear complexity, in our TIGT model. However, we found that this resulted in performance similar to models that did not use global attention. This suggests further research on effectively addressing the issue of quadratic complexity inherent in TIGT. In a similar setting, we conducted experiments with Cy2C-GNN, which has fewer parameters (114,433) compared to TIGT, and observed poorer performance. We also tested a larger version of Cy2C-GNN, named Cy2C-GNN(Large), with 1,766,401 parameters—approximately three times more than TIGT's 539,873—only to find that this resulted in a worse mean absolute error (MAE).

Table 5. Summary of the statistics of dataset in overall experiments (Dwivedi et al., 2022; 2023; Irwin et al., 2012; Hu et al., 2020).

Dataset	ZINC/ZINC-full	MNIST	CIFAR10	PATTERN	CLUSTER	Peptides-func	Peptides-struct	PCQM4Mv2
# Graphs	12,000/250,000	70,000	60,000	14,000	12,000	15,535	15,535	3,746,620
Average # nodes	23.2	70.6	117.6	118.9	117.2	150.9	150.9	14.1
Average # edges	24.9	564.5	941.1	3,039.3	2,150.9	307.3	307.3	14.6
Directed	No	Yes	Yes	No	No	No	No	No
Prediction level	Graph	Graph	Graph	Inductive node	Inductive node	Graph	Graph	Graph
Task	Regression	10-class classfi.	10-class classfi.	Binary classif.	6-class classif.	10-task classif.	11-task regression	Regression
Metric	Mean Abs. Error	Accuracy	Accuracy	Weighted Accuracy	Accuracy	Avg. Precision	Mean Abs. Error	Mean Abs. Error
Average # H1 cycles	2.8/2.8	212.1	352.5	2921.4	2034	3.7	3.7	1.4
Average magnitude # cycles	5.6/5.6	4.4	5.1	3.6	4.1	6.7	6.7	4.9
# graph w/o cycles	66/1109	0	0	0	0	1408	1408	444736

Table 6. Hyperparameters for ten datasets from BenchmarkingGNNs (Dwivedi et al., 2022), ZINC-full (Irwin et al., 2012), the Long-range Graph Benchmark (Dwivedi et al., 2023) and PCQM4Mv2 (Hu et al., 2020).

Layer		ZINC/ZINC-full	MNIST	CIFAR10	PATTERN	CLUSTER	Peptide-func	Peptides-struct	PCQM4Mv2
Topological P.E	MPNNs	GIN	GatedGCN	GAT	GatedGCN	GIN	GIN	GIN	GIN
	Weights of MPNNs	Share	Not share	Share	Not share	Not share	Not share	Not share	Not share
	Activation	Tanh	Tanh	Tanh	ReLU	Tanh	Tanh	ReLU	ReLU
	Normalize	Batch	Batch	Batch	Batch	Batch	Batch	Batch	Batch
	Self-loop	False	False	False	False	False	False	False	True
Dual-path MPNNs	MPNNs	GIN	GatedGCN	GAT	GatedGCN	GatedGCN	GatedGCN	GIN	GatedGCN
	Weights of MPNNs	Not share	Not share	Single-path	Single-path	Single-path	Single-path	Single-path	Not share
	Dropout	0.0	0.0	0.05	0.05	0.05	0.0	0.05	0.05
Global attention	# Layers	10	3	3	4	6	4	4	10
	Hidden dim	64	52	52	64	48	96	96	256
	# Heads	4	4	4	8	8	4	4	8
	Attention dropout	0.5	0.5	0.8	0.2	0.8	0.5	0.5	0.2
Graph information	Residual connection	True(In)	False	True(In)	True(In)	True(In)	True	True	True
	Pooling	Sum	Mean	Mean	Sum	Mean	Sum	Sum	Mean
	Reduction factor	4	4	4	4	4	4	4	4
Graph pooling		Sum	Mean	Mean	-	-	Mean	Mean	Mean
Train	Batch size	32/256	16	16	32	16	32	32	256
	Learning rate	0.001	0.001	0.001	0.0005	0.001	0.0003	0.0003	0.0005
	# Epochs	2000	200	100	100	100	200	200	250
	# Weight decay	1e-5	1e-5	1e-5	1e-5	1e-5	0.0	0.0	0.0
# Parameters		539873	190473	98381	279489	533814	565066	574475	13.0M

C. Implementation details

C.1. Datasets

A detail of statistical properties of benchmark datasets are summarized in Table 5. We perform the experiments on GraphGPS (Rampášek et al., 2023) framework.

C.2. Additional notes on model design

TIGT includes batch normalization layer, proposed from (Ioffe & Szegedy, 2015), and residual connection, as outlined in (He et al., 2016).

C.3. Hyperparameters

For all models we tested on the CSL dataset, we consistently set the hidden dimension to 32 and the batch size to 5. Other hyperparameters were kept consistent with those used for the models evaluated on the zinc dataset.

To ensure a fair comparison, we followed the hyperparameter settings of GraphGPS (Rampášek et al., 2023) as outlined in their benchmark datasets. It's worth noting that, due to the intrinsic nature of the TIGT architecture, the number of model parameters varies. Details regarding these hyperparameters are provided in Table 6.

Table 7. Hyperparameters for ten datasets from CSL dataset.

Layer		Cy2C-GNNs	GPS+LapPE+RWSE	GRIT+RRWP	TIGT
	Type of MPNNs	GIN	GIN	-	GIN
	Type of Attention layer	-	Transformer	GRIT	Transformer
Encoder	Hidden dim	64	64	64	64
	# Heads	-	4	4	4
	Dropout	0.0	0.0	0.0	0.0
	Batch size	4	4	4	4
	Learning rate	0.001	0.001	0.001	0.001
Train	# Epochs	200	200	200	200
	# Weight decay	1e-5	1e-5	1e-5	1e-5
	# layer (prediction head)	1	1	1	1
Preprocessing time		0.24s	0.30s	0.11s	0.24s
# Parameters/Computation time(per epoch)					
# Layers	1	36634/3.0s	45502/4.4s	50458/5.37s	64490/4.2s
# Layers	2	45082/3.3s	87422/5.7s	97626/5.73s	117114/5.8s
# Layers	5	70426/3.8s	213182/9.4s	236506/9.9s	274986/10.8s
# Layers	10	112666/5s	422782/15.5s	474970/13.7s	538106/18.3s

C.4. Implementation detail of Experiment on CSL dataset

The CSL dataset (Murphy et al., 2019) was obtained using the 'GNNBenchmarkDataset' option from the PyTorch Geometric library (Fey & Lenssen, 2019). We partitioned the dataset into training, validation, and test sets with proportions of 0.6, 0.2, and 0.2, respectively. Detailed descriptions of the hyperparameters are presented in Table 7. Hyperparameters for the CSL dataset that are not specified here are consistent with those used in the ZINC dataset experiment (Rampášek et al., 2023; Ma et al., 2023).