DOES STRUCTURAL INFORMATION HAVE BEING FULLY EXPLOITED IN GRAPH DATA?

Anonymous authors
Paper under double-blind review

ABSTRACT

In real world, graph-structured data is pervasive, operating as an abstraction of data containing nodes and interactions between nodes. There are numerous ways dedicated to excavate structure information explicitly or implicitly, but whether structural information has been adequately exploited remains an unanswered question. This work incorporates a geometric descriptor, Discrete Ricci Curvature (DRC), in order to uncover more structural information. We present a **Curv**ature-based topology-aware Gra**phormer**, termed as **Curvphormer**, that integrates DRC into a powerful graph-based Transformer architecture to build a more expressive graph-based model. This work expands the expressive to use more illuminating geometric descriptors to quantify the connections in graphs in modern models, and to extract desired structural information, such as inherent community structure in graphs with homogeneous information. We conduct extensive experiments on a variety of scaled datasets, including PCQM4M-LSC, ZINC and MolHIV, and obtain remarkable performance gain on various graph-level tasks and finetune tasks. Codes will be released upon acceptance.

1 INTRODUCTION

Graph data include considerable structural information, however existing graph-based algorithms do not fully use the inherent structural information of graphs. Real-word datasets such as citation networks (Sen et al., 2008), molecules (Joh, 2012) and the Internet (Ni et al., 2015) with inherent node-edge structure, can be naturally represented by graphs.

The vast majority of GNNs use a Message Passing (MP) mechanism to explore the graph structure information by aggregating neighborhood information (Kipf & Welling, 2017; Veličković et al., 2018; Hamilton et al., 2017), however they will unavoidably run into over-smoothing and over-squashing issues. Due to MP mechanism, most graph convolution of GNNs may be considered as a special case of Laplacian smoothing (Li et al., 2018). Analogy to random walk on graphs, smoothing operation on graphs will result in the mixing of the personalities of individual nodes. Multiple processes will be taken to smooth the characteristics of individual nodes, culminating in the reduction of variability across nodes from diverse groups. This phenomenon of incapability to classify nodes when the network goes deeper is the most widely discussed defect of GNNs, i.e., over-smoothing (Li et al., 2018; Rong et al., 2020). Another newly discussed problem of GNNs is over-squashing (Alon & Yahav, 2021; Topping et al., 2022), which indicates information flows between long-distant nodes will encounter unavoidable distortion. Over-smoothing and over-squashing are inevitable side effect of MP GNNs. Rong et al. (2020) alleviate over smoothing by randomly drop a percentage of edges in the graph. Alon & Yahav (2021) try to tackle with over-squashing by adding a fully-adjacent layer. However, these approaches cannot totally resolve these issues (Chen et al.).

Graph-based Transformers is another line of recent research. The Transformers are originally proposed as a powerful solver for Natural Language Processing(NLP) tasks (Vaswani et al., 2017), and soon became prevailing in many domains, such as computer vision (Han et al., 2022), time series (Wen et al., 2022) and graph represent learning (Chen et al., 2019; Kim et al., 2022; Dwivedi & Bresson, 2020a). For graph-based transformers, current work mainly focus on how to integrate graph structure into positional encoding(PE) in transformers (Zhang et al., 2020a; Dwivedi & Bresson, 2020a). Since graph data do not have a canonical position like images and sequences, the most widely used PE is the graph Laplacian eigenvectors, which preserve the global structure with

permutation invariance. (Dwivedi et al., 2022). Different from working on different PE methods, Graphormer (Ying et al., 2021a) adds structural encodings to the self-attention module as a structure-aware bias of attention weights. It is experimentally proved that Graphormer is exempted from the problem of over-smoothing. Moreover, because of the self-attention mechanism in the Transformer architecture, each node in the network attends to the others as if they were entirely nearby nodes. Consequently, Transformer-based graph learners can efficiently avoid the issue of overs-quashing. However, current structural descriptors, such as node degrees and shortest path distances(SPD), have limited expressiveness. Rich information in the topology of the graphs still remains unexplored.

Graph-based tasks rely heavily on structural information. The basic distinction between graph data and other data types, such as pictures or sequences, is the non-Euclidean node-edge structure. Graphs can be treated as a discretized manifold (Ni et al., 2019) from the topological view. Based on the homophily assumption of most graphs, the mainstream graph-based tasks, such as node classification, link prediction and graph classification/regression, are in essence tend to strengthen the connection between nodes with the same property, and discriminate nodes with different properties. To describe the geometric relationships of nodes from intra-/inter-communities, we draw inspiration from a recent research focus on developing community detection algorithms (Ni et al., 2019; Sia et al., 2019; Lai et al., 2022) in aid of a geometric notion, i.e., discrete Ricci curvature(DRC) (Ollivier, 2009; Lin et al., 2011).

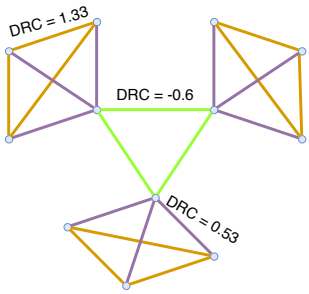

Figure 1: Illustration of DRC on a small graph. Edges in the same color have the same DRC value because of symmetry. Dense connections(yellow edges) are corresponding to positive DRC, while sparse connections(green edges) have negative DRC.

DRC quantifies the intensity of connections between nodes and their neighborhood with regard to the local graph topology. Node pairs being densely connected are associated to positive DRC values, while sparsely connected pairs give rise to negative DRC values. As illustrated in Figure 1, the nodes connected by yellow edges are in the same community and have densely connected/overlapped neighborhoods, while the nodes connected by green edges are from distinct communities with few connections/overlaps between their neighborhoods. Therefore, the DRC value of yellow edges is 1.33, which is obviously larger than -0.6 of the green edges. Purple edges are corresponding to a scenario between two extremes, thus they have a DRC between -0.6 and 1.33. Intuitively, DRC measures the connectiveness of nodes and their neighborhoods, thus it can be integrated to graph Transformers.

In this paper, we propose a novel curvature-based topology-aware graph Transformer architecture, namely Curvphormer, to exploit advanced structural information from a topological view. We evaluated the performance of our proposed algorithms on widely used testbeds such as MolHIV, PCQM4M-LSC, and ZINC. Curvphormer exceeds previous benchmarks by a significant margin.

2 METHOD

In this section, we elaborate the formulation of Discrete Ricci Curvature(DRC), and how to incorporate it in Curvphormer. Firstly, the basic settings are stated in Section. 2.1. Then, we carefully identify the Ricci curvature on graphs in Section 2.2. In Section 2.3, we propose the curvature-based topology-aware Curvphormer.

2.1 PRELIMINARIES

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a simple connected graph. $n = |\mathcal{V}|$ and $m = |\mathcal{E}|$ are the number of nodes and edges, respectively. There are two kinds of information from $\mathcal{G}$, i.e.,

- Attribute Information: It means the attribute features carried by the datasets. Such as the signal intensity of a signal tower, which can be abstracted as a node in the network. Actually, not only nodes, but also edges in graphs can contain attributes information. For example, the bonds between molecule pairs can have different types, which can be included in the edge features. We denote the node features by $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^T \in \mathbb{R}^{n \times d}$

and edge features by $\boldsymbol{E} = (\boldsymbol{x}_{e_1}, \ldots, \boldsymbol{x}_{e_m})^T \in \mathbb{R}^{m \times q}$, where d and q are the dimension of node and edge features, respectively.

- Structure Information: It means the positions and interactions of nodes. The positions of nodes is the relative position of nodes with regard to a reference or other nodes. Without loss of generality, position information can be viewed as relations between nodes due to coordinate transformation. Thus, in graphs, structure information is usually encoded by the adjacency matrix of the entire graph or subgraphs. Let $\boldsymbol{A} = \{a_{ij}\} \in \mathbb{R}^{n \times n}$ denote the adjacency matrix, where $a_{ij} = 1$ when $ij \in \mathcal{E}$ and $a_{ij} = 0$ otherwise.

2.2 DISCRETE RICCI CURVATURE

The Ricci curvature is originally a geometric notion, which plays a very important role on Riemannian manifold analysis. It quantifies the degree of space bending. For its discrete counterpart, disreteized Ricci curvature measures the connectiveness of the neighborhood of two nodes. For the discretization of Ricci curvature, there are two mainstream forms, i.e., the Ollivier Ricci curvature (Lin & Yau, 2010; Lin et al., 2011) and the Forman Ricci curvature (Sreejith et al., 2016). Since the Ollivier Ricci curvature has more theoretical foundations and depicts inherent structure more intrinsically (Samal et al., 2018), we apply a limit-free Ollivier Ricci curvature (Bai et al., 2021; Lai et al., 2022) as the definition of DRC.

The Ollivier Ricci curvature is defined on the base of the transportation distance. Firstly, we define the probability distribution of nodes on the graph, which indicates the connections or information flow between one node and others, especially its adjacent neighbors.

Definition 1 ***Probability distribution:*** *For $\forall \alpha \in [0, 1]$ and $\forall x \in \mathcal{V}$, the information flow from node x to other nodes $y \in \mathcal{V}$ can be defined as a probability distribution on $\mathcal{V}$ by*

$$m_x^\alpha(y) := \begin{cases} \alpha, & y = x, \\ (1-\alpha)\frac{\gamma(w_{xy})}{\sum_{z \sim x} w_{xz}}, & y \sim x, \\ 0, & otherwise. \end{cases} \tag{1}$$

where w_{xy} denotes the edge weight on $xy \in \boldsymbol{E}$, and $\gamma(\cdot)$ is an arbitrary non-negative real-valued one-to-one function.

The distance between any two node x and y and their neighborhood can be defined as the transportation distance between two distributions m_x^α and m_y^α.

Definition 2 ***Transportation distance:*** *Let $A(x, y) : \mathcal{V} \times \mathcal{V} \to [0, 1]$ be a coupling satisfying*

$$\sum_{y \in \mathcal{V}} A(x, y) = m_x^\alpha \qquad \textit{and} \qquad \sum_{x \in \mathcal{V}} A(x, y) = m_y^\alpha. \tag{2}$$

Then the transportation distance between two probability distribution m_x^α and m_y^α is defined as

$$W(m_x^\alpha, m_y^\alpha) := \inf_A \sum_{x,y \in \mathcal{V}} A(x, y) d(x, y), \tag{3}$$

where $d(\cdot, \cdot)$ is a distance function.

Here we leverage Dijkstra's Shortest Path Distance as $d(\cdot, \cdot)$ in this work. In order to differentiate topology structures on the base of graph geometry, DRC is defined as below:

Definition 3 ***α-Ricci curvature:***

$$\kappa_\alpha(x, y) = 1 - \frac{W(m_x^\alpha, m_y^\alpha)}{d(x, y)}, \quad \forall \alpha \in [0, 1]. \tag{4}$$

Ollivier Ricci curvature (Lin et al., 2011):

$$\kappa(x, y) = \lim_{\alpha \to 1} \frac{\kappa_\alpha(x, y)}{1 - \alpha}. \tag{5}$$

Note that, in the computation of Ollivier Ricci curvature, when the node pair x and y connect densely, $\kappa(x, y)$ will be larger than the sparsely connected pairs. When computing Ollivier Ricci curvature, in order to avoid limit operation, former works set α to 0.5 (Ni et al., 2015; 2019) and utilize κ_α as an approximation of α. In this work, we leverage another limit-free version of Ollivier Ricci curvature for computation convenience. Let B is a $*$-coupling between two probability distribution m_x^0 and m_y^0 (See A.1).

Theorem 1 *The $*$-coupling based Ricci curvature is formulated by*

$$\kappa^*(x, y) = \frac{1}{d(x, y)} \sup_B \sum_{u,v \in \boldsymbol{V}} B(u, v) d(u, v). \tag{6}$$

Then for any $x, y \in \boldsymbol{V}$, $x \neq y$, the following equation holds:

$$\kappa^*(x, y) = \kappa(x, y). \tag{7}$$

Thus κ^* illustrates the topological characteristic of a graph as Ollivier Ricci curvature, and exempts from limit calculation. In our implementation, we leverage this κ^* curvature when computing DRC, and denote DRC by κ for simplicity. The proof of Theorem 1 can refer to Bai et al. (2021). Algorithm 1 in A.2 formulates the computation of DRC.

2.3 CURVPHORMER

Curvphormer incorporates the advanced geometric information represented by DRC, and encodes it into the Graphormer architecture. The overall architecture of Curvphormer is demonstrated in Figure 2.

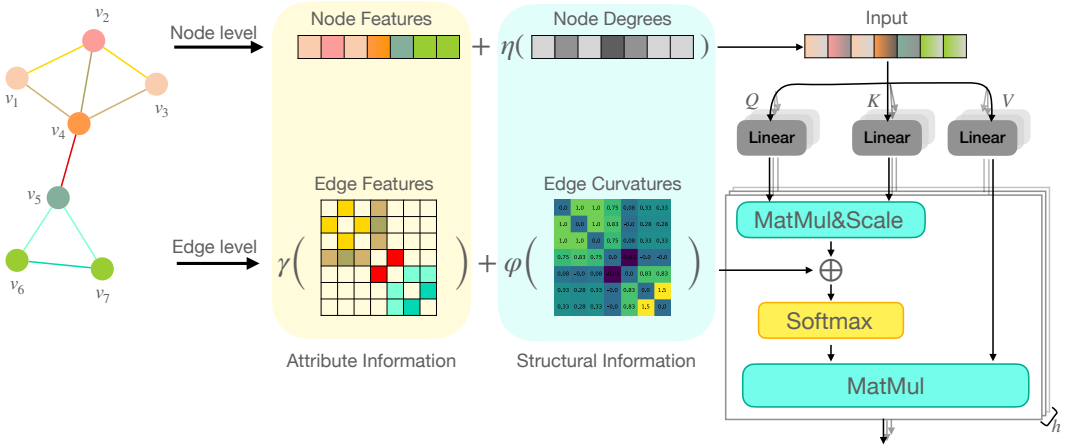

Figure 2: Illustration of Curvphormer with attribute/structure encodings. The input is a combination of two types of node level information, i.e., node features and node degree encoding. Edge level information, i.e., encodings of edge features and curvatures, describes interactions between node pairs, therefore these two encodings are added to the multi-head self-attention module as a bias of the attention weights.

2.3.1 ATTRIBUTE ENCODING

As mentioned before, in graph data, attribute information is the features carried by nodes and edges, describing some specific information in dataset. Node features are the most important information characterizing a dataset. In Curvphormer, we leverage node features without any affine transformation. In many graphs, edges also have attribute features, which is essential for understanding the underlying graph structure. Although edge features are provided by the dataset, they usually indicate the type or intensity of interactions between nodes. Thus, for any node pair (v_i, v_j) in a graph, the correlation between v_i and v_j have to attend to the edges connecting them. Let v_i and v_j are

connected by a shortest path denoted by $v_i \overset{e_1}{\sim} \cdots \overset{e_N}{\sim} v_j$. The correlation between v_i and v_j can be formulated by the mean of the embedded edge features along the path.

$$\gamma(v_i, v_j) = \frac{1}{N} \sum_{k=1}^{N} \text{EdgeEmbedding}_k(\boldsymbol{x}_{e_k}), \tag{8}$$

where $\boldsymbol{x}_{e_k} \in \mathbb{R}^q$ is the edge feature of e_k. $\text{EdgeEmbedding}_k(\boldsymbol{x}_{e_k}) = \boldsymbol{x}_{e_k}^T \cdot \boldsymbol{w}_k$, $\boldsymbol{w}_k \in \mathbb{R}^q$ is a learnable vector.

2.3.2 Structural Encoding

Structural information here refers to the knowledge of the graph that induced by the connectiveness. As demonstrated in Figure 2, we consider two dimensions of structural information. One is the node level information to quantify the importance of nodes in the graph. Take the citation network as an example, the more influential a paper is, the more citations it has, and vice versa. Thus, in an abstract graph, an important node must connect to more neighbors. The node degree is an intuitive choice to describe this node property as in (Ying et al., 2021a). Let $d_i = \sum_{j \in \mathcal{V}} a_{ij}$ be the degree of node v_i. Then we embed d_i to a vector:

$$\eta(v_i) = d_i \cdot \boldsymbol{w}_i, \tag{9}$$

where $\boldsymbol{w}_i \in \mathbb{R}^d$ is a learnable vector. Then incorporate node degree embedding matrix $\boldsymbol{D} = (\eta(v_1), \ldots, \eta(v_n))^T \in \mathbb{R}^{n \times d}$ with the node features as the input of the subsequent module, i.e., $\boldsymbol{H}^{(0)} = \boldsymbol{X} + \boldsymbol{D}$.

The other is the edge level information which can be interpreted by the positional relationship between any node pairs via the edges connecting them. Former works encode position information on graphs by simple Shortest Path Distance (SPD) (Ying et al., 2021a; Chen et al., 2019; Cai & Lam, 2020). However, SPD can only provide a relative distance on graphs. Graphs can be viewed as a discretized manifold in Riemannian spaces. Thus the topology structure of the manifold determines the foundation of graphs. Pure SPD neglects the topology structure of the spaces where graphs embedded in. As we stated in Section 2.2, DRC depicts the connectiveness on the basis of the node's neighborhoods. Nodes with positive DRC connect densely, while negative DRC is related to sparsely connected nodes. By virtue of the expressive power of DRC, we encode the relations of nodes on graph topology with

$$\varphi(v_i, v_j) = \kappa(v_i, v_j) \cdot w_{ij}, \tag{10}$$

where w_{ij} is a learnable scalar.

2.3.3 Self-Attention Mechanism

Self-attention module is the main part of the Transformer architecture, which captures the global information by connecting all positions (Vaswani et al., 2017; Ying et al., 2021a). It computes the weighted sum of values, where the weights of values is obtained by a query-key function. Let $\boldsymbol{H} = (\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n)^T \in \mathbb{R}^{n \times d}$ be the input of the module. In Curvphormer, when a node attends other nodes in the graph, the edge attribute information $\boldsymbol{\Gamma} = \{\gamma(v_i, v_j)\}$ as well as the DRC-based structural information $\boldsymbol{\Phi} = \{\varphi(v_i, v_j)\}$ are added to the attention weights to provide more topology-aware ability. Therefore, the self-attention can be formulated by

$$\text{Attention}(\boldsymbol{H}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_K}} + \boldsymbol{\Gamma} + \boldsymbol{\Phi}\right)\boldsymbol{V}, \tag{11}$$

where $\boldsymbol{Q} = \boldsymbol{H}\boldsymbol{W}_Q, \boldsymbol{K} = \boldsymbol{H}\boldsymbol{W}_K, \boldsymbol{V} = \boldsymbol{H}\boldsymbol{W}_V$, and $\boldsymbol{W}_Q, \boldsymbol{W}_K \in \mathbb{R}^{d \times d_K}, \boldsymbol{W}_V \in \mathbb{R}^{d \times d_V}$. Thus the corrlation between nodes v_i and v_j is

$$A_{ij} = \text{softmax}\left(\frac{(\boldsymbol{h}_i\boldsymbol{W}_Q)(\boldsymbol{h}_j\boldsymbol{W}_K)^T}{\sqrt{d_K}} + \gamma(v_i, v_j) + \varphi(v_i, v_j)\right)\boldsymbol{V}. \tag{12}$$

For multi-head self-attention is obtained by

$$\text{MHA}(\boldsymbol{H}) = \text{Concat}\left(\text{Attention}_1(\boldsymbol{H}), \ldots, \text{Attention}_h(\boldsymbol{H})\right)\boldsymbol{W}_O, \tag{13}$$

where $\boldsymbol{W}_O \in \mathbb{R}^{hd \times d_{\text{model}}}$.

2.3.4 CURVPHORMER STRUCTURE

Curvphormer follows the basic architecture of Graphormer (Ying et al., 2021a), which is a variant of the vanilla Transformer encoder (Vaswani et al., 2017). Each layer of Curvphormer is consist of a multi-head attention module(MHA) and a feed-forward network(FFN) module. The detailed implementation of a Curvphormer layer is formulated as

$$\widehat{\boldsymbol{H}}^{(l+1)} = \text{MHA}(\text{LayerNorm}(\boldsymbol{H}^{(l)})) + \boldsymbol{H}^{(l)} \tag{14}$$

$$\boldsymbol{H}^{(l+1)} = \text{FFN}(\text{LayerNorm}(\widehat{\boldsymbol{H}}^{(l+1)})) + \widehat{\boldsymbol{H}}^{(l+1)} \tag{15}$$

Besides, in order to enhance the ability of Curvphormer to capture the representation of the entire graph, like in (Ying et al., 2021a), a virtual node is applied, which is connected to all nodes in the graph by virtual edges, and the corresponding structural encodings are set to distinct learnable variables.

The training procedure of Curvphormer is mainly based on a Transformer encoding module. The self-attention mechanism has a complexity of $\mathcal{O}(n^2 \cdot d)$ per layer, where n is the number of nodes, and d is the dimension of node features. Before training, Curvphormer computes DRC as the input of structural encoding. The computing complexity of DRC is $\mathcal{O}(m \cdot \bar{d}^3)$, where m is the number of edges, and $\bar{d}$ is the average degree of nodes. It is time consuming for DRC computing on very large graphs, thus we utilize this valuable structural information as a preprocess of graphs before training.

3 EXPERIMENTS

In this section, we conduct three experiments to intuitively clarify the motivation as well as effectiveness of Curvphormer. Firstly, we illustrate the importance of the topology information in Section 3.1 on a small dataset, i.e., the Zachary's Karate Club Network (Zachary, 1977), indicating the importance of our inclusion of curvature as a factor. Then, we intuitively show the expressiveness of DRC on graph structures comparing with the widely used graph structure descriptor SPD in Section 3.2. Finally, we perform experiments on three different scaled real-world datasets to test the performance of Curvphormer in Section 3.3.

3.1 STRUCTURAL INFORMATION IS CRUCIAL IN GRAPH-BASED TASKS

To illustrate the importance of graph structure information, we devise a binary node classification experiment on the small Karate Club Network(Karate). Karate is composed of two communities with 34 nodes(members of the club). The edges between nodes indicate interactions between club members. We apply a simple two-layer GCN model (Kipf & Welling, 2016) to learn the underlying graph structure. And the node features is provided by three designed cases, i.e., random numbers, SPD and DRC, for testing the influence of different kinds of information in a simple NN-based model.

The accuracy of these three scenarios is shown in Table 1(best performance in 10 runs). For random features, even though they cannot provide any useful information, the classification accuracy is still better than random guess because of the utilization of adjacency matrix in the model. Notice that when more structure information is provided, the performance of the model improves remarkably. Moreover, DRC outperforms SPD in this experiment setting. It indicates that advanced topology information can excavate more effective structural information than simple distance information.

Feature type	**Feature Description**	**Accuracy(%)**
Random numbers	no useful information.	78
SPD	provides distance information for nodes.	95
DRC	provides an advanced topology information.	**97**

Table 1: Test different types of structure information on the Karate dataset with a 2-layer GCN. Structural information yields better results, and advanced topological DRC outperforms SPD.

3.2 WHY DRC DEPICTS STRUCTURAL INFORMATION BETTER THAN SPD?

Now we intuitively show the expressiveness of DRC comparing with SPD by a small graph with two small communities bridging by an edge as shown in Figure 3. Though both SPD and DRC have the ability to know there are two communities, DRC depicts more in-depth structure information than SPD. Note the interactions between nodes $1, 3$ and nodes $1, 5$. Node 1 and 3 are from the same community, while node 1 and 5 are from different communities. The relationships of these two pairs are different, while $\text{SPD}_{13} = \text{SPD}_{15} = 2$ (highlighted by orange circles in Figure $3(c)$). Besides, edge e_{45} is the only bridge edge connecting two communities. However, $\text{SPD}_{45} = 1$ (red dotted circle in Figure $3(c)$) can not differentiate e_{45} from other 1-hop pairs. SPD is incapable to describe these differences in structure. Fortunately, DRC can amend these defect because it attends to the nodes' neighborhoods. The tightly interacted pairs are tend to have larger DRC than sparsely interacted pairs. $\text{DRC}_{13} = 1$ is apparently larger than $\text{DRC}_{15} = 0.08$ for the first case. Meanwhile, $\text{DRC}_{45} = -0.83$ highlights the difference of this edge from others.

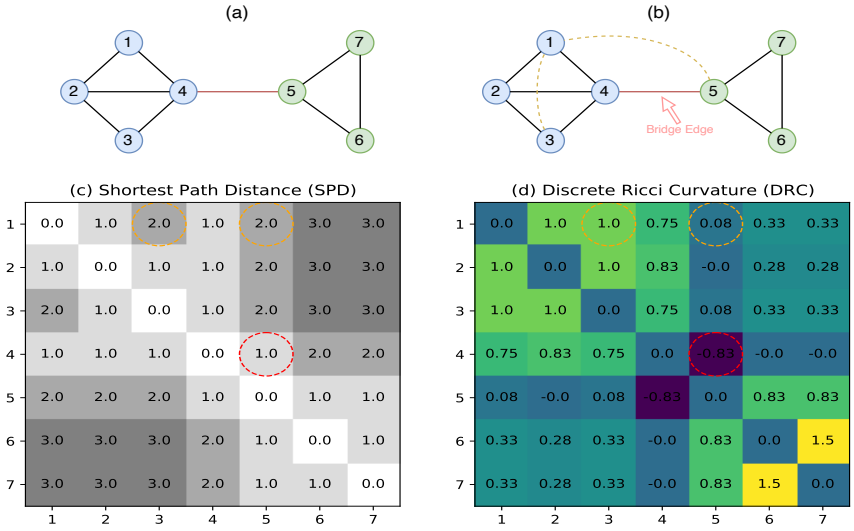

Figure 3: A small graph demonstrates the structural expresiveness of SPD v.s. DRC. The difference between (1) inter-/intra-community relations, i.e., $1\&3$ and $1\&5$, (2) the bridge edge e_{45} and other 1-hop pairs, can not be captured by SPD, but well described by DRC.

3.3 EXPERIMENTS ON REAL-WORD DATASETS

In this part, we build up our experiments on three different scaled datasets, i.e., MolHIV(small), ZINC(medium) and PCQM4M-LSC(large). Refer to A.3 for details about the datasets.

3.3.1 EXPERIMENTAL SET-UP

We benchmark Curvphormer with the non-topology-aware Graphormer baseline Ying et al. (2021a). Basic setting of Curvphormer follows (Ying et al., 2021b) but modified some parameters for model finetune. The number of attention heads and the dimension of node/edge features are set to 16. We use AdamW as the optimizer, and set the hyper-parameter Adam-ϵ to 1e-8 and Adam-(β_1, β_2) to $(0.99, 0.999)$. The learning rate is set to 2e-4 with a lower-bound 1e-9. The batch size is set to 512. All models and tasks are trained on 8 NVIDIA 3080ti GPUs for about three days. Other settings are the same as the baseline. We train Curvphormer on PCQM4M-LSC and ZINC from scratch and To test the finetune the pre-trained model on ZINC with the small dataset MolHIV to test the transferable ability of Curvphormer. In addition, in order to test if Curvphormer can effectively resist the performance drop caused by over-smoothing, we test Curvphormer on MolHIV dataset with varying number of layers up to 20.

Datasets	Scale	Task	Model	#Layers	#Param	validMAE
PCQM4M-LSC	Large	Regression	GCN (Brossard et al., 2020)	12	2.0M	0.1691
			GIN (Xu et al., 2018)	12	3.8M	0.1537
			DeeperGCN (Li et al., 2020)	12	25.5M	0.1398
			GT (Dwivedi & Bresson, 2020b)	12	0.6M	0.1400
			Graphormer$_{\text{SMALL}}$ (Ying et al., 2021b)	12	12.5M	0.1264
			Graphormer (Ying et al., 2021b)	12	47.1M	0.1234
			Curvphormer	8	34.1M	**0.1024**
			Model	#Layers	#Param	testMAE
ZINC	Medium	Regression	GIN (Xu et al., 2018)	2	510K	0.526
			GraphSage (Hamilton et al., 2017)	2	505K	0.398
			GAT (Veličković et al., 2017)	2	531K	0.384
			GCN (Brossard et al., 2020)	2	505K	0.367
			GatedGCN-PE (Bresson & Laurent, 2017)	2	505K	0.367
			PNA (Corso et al., 2020)	16	387K	0.214
			Graphormer$_{\text{SLIM}}$ (Ying et al., 2021b)	12	47.0M	0.122
			Curvphormer	8	34.1M	**0.080**
			Model	#Layers	#Param	AUC(%)
MolHIV	Small	Classification	GCN-$_{\text{GraphNorm}}$ (Brossard et al., 2020)	12	526K	78.83
			PNA (Corso et al., 2020)	12	326K	79.05
			PHC-GNN (Le et al., 2021)	12	111K	79.34
			DeeperGCN–$_{\text{FLAG}}$ (Li et al., 2020)	12	532K	79.42
			DGN (Beaini et al., 2021)	12	114K	79.70
			Graphormer-$_{\text{FLAG}}$ (Ying et al., 2021b)	12	47.0M	80.51
			Curvphormer	12	47.1M	**83.93**

Table 2: Results on the PCQM4M-LSC, ZINC and MolHIV datasets. Performance metric for regression task on PCQM4M-LSC and ZINC is MAE, and for classification task on MolHIV is AUC. Curvphormer outperforms the benchmarks on all these datasets.

3.3.2 RESULTS

Table 2 summarizes the performance of Curphormer and other baselines on PCQM4M-LSC, ZINC and MolHIV. The metrics are mean absolute error(MAE) for regression task and AUC for classification task. Curvphormer achieves the best results and noticeably surpasses the previous state-of-art GNNs as well the recent graph-Transformer model GT (Dwivedi & Bresson, 2020a) and Graphormer (Ying et al., 2021a).

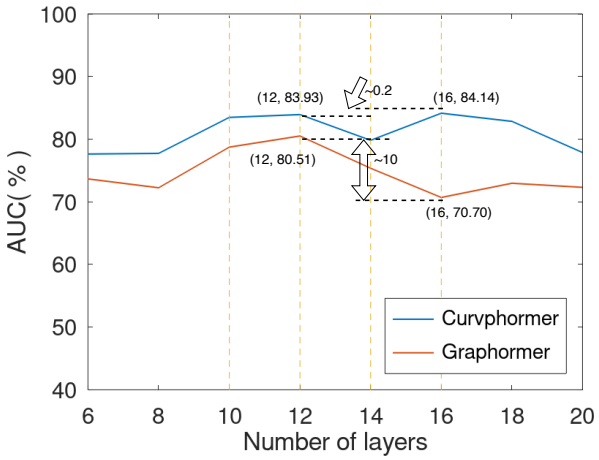

Figure 4: Testing the performance of Curvphormer on MolHIV for different number of layers. Curvphormer surpasses the baseline Graphormer by a significant margin, and attains stable satisfactory performance for varying number of layers.

Next, we test deeper Curvphormer's performance on the MolHIV dataset comparing with the baseline Graphormer. Figure 4 shows that both models are capable of resisting over-smoothing. Meanwhile, Curvphormer surpasses Graphormer by a noticeable margin for all layer configurations. it is noteworthy that when the model layer changes from 12 to 16, the performance of Graphormer drops from 80.51 to 70.70. In contrast, Curvphormer achieves a comparable result after a slight drop.

4 RELATED WORK

In this section, we highlight the most recent approaches on NN-based models working on demystify the structural information of graph data. And then we give prominence to some related applications of DRC in finding the underlying structure of graphs.

4.1 STRUCTURAL ENCODINGS

On MP-GNNs GNN methods to processing graph data have natural merits for the theoretical basis. Most GNNs follow the scheme of MP mechanism, and leverage random walk algorithms to explore the underlying structure of graphs in aid of stochastic process (Li et al., 2018; Gasteiger et al., 2019). Some other GNN methods try to incorporate local structure information by utilizing a local k-hop subgraph as the structural fingerprint of its central node (Zhang et al., 2020b; Wang et al., 2021). Moreover, some methods propose to introduce some additional structural information encoded by geometric notions such as DRC to GNNs explicitly or implicitly (Ye et al., 2020; Li et al., 2022). However, due to the inevitable over-smoothing and over-squashing problem and limited expressiveness of GNNs, the increment of structural information does not yield much improvement in performance.

On graph-based Transformers The challenge of powerful Transformer architecture in graph representation is how to properly encode structural information into a positional encoding module (Dwivedi & Bresson, 2020a) or the self-attention module (Ying et al., 2021a). (Dwivedi & Bresson, 2020a) exploit graph structure by pre-computing Laplacian eigenvectors of the adjacency matrix as positional encoding(PE) in the vanilla Transformer architecture to provide a distance-aware information. Graph-BERT (Zhang et al., 2020a) operates on sampled linkless subgraphs for local structural information and enhance the capability on extremely large graphs. What's more, Graph-BERT introduces three PE embeddings to take in positional information on local subgraphs. Specifically, a Weisfeiler-Lehman(WL) absolute PE is leveraged to capture the global information, an intimacy-based and a hop-based relative PE are introduced to extract the local information in subgraphs. It is notable that TokenGT (Kim et al., 2022) puts forward that pure Transformers can attain impressive performance on graphs by a orthonormal node identifier and a type identifier. It suggests that the Transformer architecture itself has the potential to fit in the graph structure. Further involving an advanced geometric descriptor into the Transformer architecture is a promising direction.

4.2 DRC IN FINDING GRAPH STRUCTURE

In light of the property of Ricci curvature in Riemannian geometry, the discrete version of Ricci curvature is an instinct choice as a topological descriptor. (Ni et al., 2015) leverages DRC to analyze the Internet topologies. (Sia et al., 2019) constructs a community detection algorithm by removing negative curved edges step-by-step. (Ni et al., 2019; Lai et al., 2022) leverage a DRC-based Ricci flow to deform a graph, then intra-community nodes get closer and inter-community nodes disperse. DRC is capable of finding the underlying relationship between nodes, characterizing them to clusters with identical or distinct properties.

5 CONCLUSION AND DISCUSSION

This work introduces Curvphormer, a topology-aware graph Transformer that incorporate an advanced structural information into expressive Graphormer architecture. DRC effectively differentiate topology structure of graphs with homophily property, and helps our model achieve remarkable performance improvements on different scaled datasets in graph classification/regression tasks. It shows that applying more geometric descriptors to expressive graph models is rewarding. Meanwhile, the exploration of graph structural information is still challenging. For example, discovering the topology information of heterogeneous graphs still needs future endeavor. Moreover, the computation complexity of DRC restricts its application in large dynamic systems. Curvphormer brings a inspiration on a better understanding of graph structure, and encourages more future works.

REFERENCES

Zinc: A free tool to discover chemistry for biology. Journal of Chemical Information and Modeling, 52(7):1757–1768, 2012. doi: 10.1021/CI3001277.

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In International Conference on Learning Representations, 2021. URL https://openreview.net/forum?id=i80OPhOCVH2.

Shuliang Bai, An Huang, Linyuan Lu, and Shing-Tung Yau. On the sum of ricci-curvatures for weighted graphs. Pure and Applied Mathematics Quarterly, 17(5):1599 – 1617, 2021.

Dominique Beaini, Saro Passaro, Vincent Létourneau, Will Hamilton, Gabriele Corso, and Pietro Liò. Directional graph networks. In International Conference on Machine Learning, pp. 748–758. PMLR, 2021.

Xavier Bresson and Thomas Laurent. Residual gated graph convnets. arXiv preprint arXiv:1711.07553, 2017.

Rémy Brossard, Oriel Frigo, and David Dehaene. Graph convolutions that can finally model local structure. arXiv preprint arXiv:2011.15069, 2020.

Deng Cai and Wai Lam. Graph transformer for graph-to-sequence learning. In Proceedings of The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI), 2020.

Benson Chen, Regina Barzilay, and Tommi S. Jaakkola. Path-augmented graph transformer network. CoRR, abs/1905.12712, 2019.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In Proceedings of the 37th International Conference on Machine Learning, volume 119 of Proceedings of Machine Learning Research, pp. 1725–1735. URL https://proceedings.mlr.press/v119/chen20v.html.

Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. Advances in Neural Information Processing Systems, 33:13260–13271, 2020.

Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. ArXiv, abs/2012.09699, 2020a.

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

Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. arXiv preprint arXiv:2003.00982, 2020.

Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. In International Conference on Learning Representations, 2022. URL https://openreview.net/forum?id=wTTjnvGphYj.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In International Conference on Learning Representations (ICLR), 2019.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In NIPS, 2017.

Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A survey on vision transformer. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1–1, 2022. doi: 10.1109/tpami.2022.3152247. URL https://doi.org/10.1109%2Ftpami.2022.3152247.

Jinwoo Kim, Tien Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. arXiv, abs/2207.02505, 2022. URL https://arxiv.org/abs/2207.02505.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations (ICLR), 2017.

Xin Lai, Shuliang Bai, and Yong Lin. Normalized discrete ricci flow used in community detection. Physica A: Statistical Mechanics and its Applications, 597:127251, 2022. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2022.127251. URL https://www.sciencedirect.com/science/article/pii/S0378437122002242.

Tuan Le, Marco Bertolini, Frank Noé, and Djork-Arné Clevert. Parameterized hypercomplex graph neural networks for graph classification. In International Conference on Artificial Neural Networks, pp. 204–216. Springer, 2021.

Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. Deepergcn: All you need to train deeper gcns. arXiv preprint arXiv:2006.07739, 2020.

Haifeng Li, Jun Cao, Jiawei Zhu, Yu Liu, Qing Zhu, and Guohua Wu. Curvature graph neural network. Information Sciences, 592:50–66, 2022. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2021.12.077. URL https://www.sciencedirect.com/science/article/pii/S0020025521012986.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In AAAI, pp. 3538–3545. AAAI Press, 2018.

Y. Lin and S. T. Yau. Ricci curvature and eigenvalue estimate on locally finite graphs. Mathematical Research Letters, 17(2):343–356, 2010. ISSN 1073-2780. doi: 10.4310/MRL.2010.v17.n2.a13.

Yong Lin, Linyuan Lu, and Shing-Tung Yau. Ricci curvature of graphs. Tohoku Mathematical Journal, 63(4):605–627, 2011. ISSN 0040-8735. doi: 10.2748/tmj/1325886283.

C. C. Ni, Y. Y. Lin, F. Luo, and J. Gao. Community detection on networks with ricci flow. Scientific Reports, 9(1):9984, 2019. ISSN 2045-2322 (Electronic)2045-2322 (Linking). doi: 10.1038/s41598-019-46380-9.

Chien-Chun Ni, Yu-Yao Lin, Jie Gao, Xianfeng David Gu, and Emil Saucan. Ricci curvature of the internet topology. In 2015 IEEE Conference on Computer Communications (INFOCOM), pp. 2758–2766, 2015. doi: 10.1109/INFOCOM.2015.7218668.

Y. Ollivier. Ricci curvature of markov chains on metric spaces. Journal of Functional Analysis, 256 (3):810–864, 2009. ISSN 0022-1236. doi: 10.1016/j.jfa.2008.11.001.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. DropEdge: Towards deep graph convolutional networks on node classification. In International Conference on Learning Representations, 2020. URL https://openreview.net/forum?id=Hkx1qkrKPr.

Areejit Samal, R. P. Sreejith, Jiao Gu, Shiping Liu, Emil Saucan, and Jürgen Jost. Comparative analysis of two discretizations of ricci curvature for complex networks. Scientific Reports, 8(1): 1–16, Jun 2018. ISSN 2045-2322. doi: 10.1038/s41598-018-27001-3.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. AI Magazine, 29(3):93, Sep. 2008. doi: 10.1609/aimag.v29i3.2157. URL https://ojs.aaai.org/index.php/aimagazine/article/view/2157.

J. Sia, E. Jonckheere, and P. Bogdan. Ollivier-ricci curvature-based method to community detection in complex networks. Scientific Reports, 9(1):9800, 2019. ISSN 2045-2322 (Electronic) 2045-2322 (Linking). doi: 10.1038/s41598-019-46079-x.

R. P. Sreejith, K. Mohanraj, J. Jost, E. Saucan, and A. Samal. Forman curvature for complex networks. Journal of Statistical Mechanics Theory and Experiment, 2016(6):063206, 2016. doi: 10.1088/1742-5468/2016/06/063206.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In International Conference on Learning Representations, 2022. URL https://openreview.net/forum?id=7UmjRGzp-A.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. arXiv preprint arXiv:1710.10903, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. International Conference on Learning Representations, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec. Multi-hop attention graph neural networks. In IJCAI, 2021.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. arXiv preprint arXiv:2202.07125, 2022.

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

Ze Ye, Kin Sum Liu, Tengfei Ma, Jie Gao, and Chao Chen. Curvature graph network. In International Conference on Learning Representations, 2020. URL https://openreview.net/forum?id=BylEqnVFDB.

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? In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), Advances in Neural Information Processing Systems, 2021a. URL https://openreview.net/forum?id=OeWooOxFwDa.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? Advances in Neural Information Processing Systems, 34:28877–28888, 2021b.

Wayne Zachary. An information flow model for conflict and fission in small groups. Journal of Anthropological Research, 33(4):452–473, 1977. doi: 10.1086/jar.33.4.3629752.

Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. Graph-bert: Only attention is needed for learning graph representations. arXiv preprint arXiv:2001.05140, 2020a.

Kai Zhang, Yaokang Zhu, Jun Wang, and Jie Zhang. Adaptive structural fingerprints for graph attention networks. In International Conference on Learning Representations, 2020b. URL https://openreview.net/forum?id=BJxWx0NYPr.

A APPENDIX

A.1 A LIMIT-FREE OLLIVIER RICCI CURVATURE (BAI ET AL., 2021)

Definition 4 *Let $B : V \times V \to \mathbb{R}$ be a coupling function. We simply denote μ_x^0 as μ_x. For any $x, y \in V$, if B satisfies:*

1. $B(x, y) > 0$, *while* $B(u, v) \leq 0$ *for* $u \neq x$ *or* $v \neq y$;
2. $\sum_{u,v \in V} B(x, y) = 0$;
3. $\sum_{v \in V} B(u, v) = -\mu_x(u)$ *for all* $u \neq x$;
4. $\sum_{u \in V} B(u, v) = -\mu_y(v)$ *for all* $v \neq y$.

Then we call B as a $*$-coupling between μ_x and μ_y.

A.2 DISCRETE RICCI CURVATURE(DRC) ALGORITHM

Algorithm 1: Computation of Discrete Ricci Curvature(DRC).

Input: A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
Output: A weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w, \kappa)$, where w and κ are the weights and discrete Ricci curvature on edges, respectively.

Initialization. Edge weights $w_e = 1, \forall e \in \mathcal{E}$;
Compute the Shortest Path Distance(SPD) of each pair of nodes, i.e., $d(u, v) \forall u, v \in \mathcal{V}$;
for $e = (x, y) \in \mathcal{E}$ **do**
    Compute the discrete Ricci curvature. $\kappa_e = \frac{1}{d(x,y)} \sup_B \sum_{u,v \in \mathcal{V}} B(u, v) d(u, v)$;
end

A.3 DETAILS OF DATASETS

We summarize the datasets used in this work in Table 1, Table 2 and Figure 4.

DATASETS	Scale	#Graphs	#Nodes	#Edges	Task Type
Karate	Very small	1	33	78	Binary classification
ZINC(sub-set)	Small	12,000	277,920	597,960	Regression
MolHIV	Medium	41,127	1,048,738	1,130,993	Binary classification
PCQM4M-LSC	Large	3,803,453	53,814,542	55,399,880	Regression

Table 3: Statistics of the datasets.

Next we state detailed information of the four datasets we used, including their features and the reasons we choose them.

- The Karate Club complex network is a network commonly used for community detection studies in complex networks. The network has 34 nodes and 78 edges, where 34 nodes represent 34 members of a karate club and the edges between nodes represent two members who know each other, and the dataset is a real dataset that corresponds to a study of the relationships of people in a karate club in the United States. This dataset is a real dataset that corresponds to a study of the relationships of people in a karate club in the United States. It is of extraordinary interest for community discovery studies in complex networks 3.

- The Open Graph Benchmark (OGB[1]) is a collection of realistic, large-scale, and diverse benchmark datasets for machine learning on graphs. OGB datasets are automatically downloaded, processed, and split using the OGB Data Loader. The model performance can be evaluated using the OGB Evaluator in a unified manner. OGB is a community-driven initiative in active development.
- ZINC is a free database of commercially-available compounds for virtual screening. ZINC contains over 230 million purchasable compounds in ready-to-dock, 3D formats. ZINC also contains over 750 million purchasable compounds that can be searched for analogs.
- OGB Large-Scale Challenge (OGB-LSC[2]) is a collection of three real-world datasets for advancing the state-of-the-art in large-scale graph ML. OGB-LSC provides graph datasets that are orders of magnitude larger than existing ones and covers three core graph learning tasks – link prediction, graph regression, and node classification.

The task of Karate is to distinguish the type of community a person belongs to. And we employ another popular leaderboard, i.e., benchmarking-gnn (Dwivedi et al., 2020). We use the ZINC datasets, which is the most popular real-world molecular dataset to predict graph property regression for contrained solubility, an important chemical property for designing generative GNNs for molecules. Different from the scaffold spliting in OGB, uniform sampling is adopted in ZINC for data splitting. In addition, the task of PCQM4M-LSC is to predict DFT(density functional theory)-calculated HOMO-LUMO energy gap of molecules given their 2D molecular graphs, which is one of the most practically-relevant quantum chemical properties of molecule science. PCQM4M-LSC is unprecedentedly large in scale comparing to other labeled graph-level prediction datasets, which contains more than 3.8M graphs.

[1] https://ogb.stanford.edu/

[2] https://ogb.stanford.edu/docs/lsc/