Substructure Aware Graph Neural Networks

Dingyi Zeng[1*], Wanlong Liu[1*], Wenyu Chen[1], Li Zhou[1], Malu Zhang[1], Hong Qu[1†]

[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
zengdingyi@std.uestc.edu.cn, liuwanlong@std.uestc.edu.cn, cwy@uestc.edu.cn, li_zhou@std.uestc.edu.cn,
maluzhang@uestc.edu.cn, hongqu@uestc.edu.cn

Abstract

Despite the great achievements of Graph Neural Networks (GNNs) in graph learning, conventional GNNs struggle to break through the upper limit of the expressiveness of first-order Weisfeiler-Leman graph isomorphism test algorithm (1-WL) due to the consistency of the propagation paradigm of GNNs with the 1-WL. Based on the fact that it is easier to distinguish the original graph through subgraphs, we propose a novel framework neural network framework called Substructure Aware Graph Neural Networks (SAGNN) to address these issues. We first propose a Cut subgraph which can be obtained from the original graph by continuously and selectively removing edges. Then we extend the random walk encoding paradigm to the return probability of the rooted node on the subgraph to capture the structural information and use it as a node feature to improve the expressiveness of GNNs. We theoretically prove that our framework is more powerful than 1-WL, and is superior in structure perception. Our extensive experiments demonstrate the effectiveness of our framework, achieving state-of-the-art performance on a variety of well-proven graph tasks, and GNNs equipped with our framework perform flawlessly even in 3-WL failed graphs. Specifically, our framework achieves a maximum performance improvement of 83% compared to the base models and 32% compared to the previous state-of-the-art methods.

Introduction

Structured data modeled as graphs has a long history in many fields such as chemo-informatics (Gilmer et al. 2017), bioinformatics (Gainza et al. 2020), physics (Battaglia et al. 2016), traffic (Pan et al. 2023), scene understanding (Liu et al. 2022b) and recommender systems (Wang et al. 2019). The use of Graph Neural Networks (GNNs) to process graph-structured data has also become common, where concise and elegant learning paradigms like Message-passing Neural Networks(MPNNs) (Gilmer et al. 2017) have achieved great success. Although deep learning has greatly enhanced the influence of artificial intelligence in various fields in recent years (Wang et al. 2022, 2023; Zhang et al. 2022; Li et al. 2022; Wang and Chen 2023). With the in-depth researche of GNNs, the foundational problem that limits the expressive power of GNNs is discovered, namely that the paradigm of their message passing limits their ability to perceive structures. The message passing paradigm of GNNs (Xu et al. 2018a) is consistent with the Weisfeiler-Lehman graph isomorphism test algorithm (Weisfeiler and Leman 1968), which leads to the upper limit of GNNs' expressive power within the 1-WL. However, many higher-order substructures are extremely important for specific downstream tasks (Fey, Yuen, and Weichert 2020; Thiede, Zhou, and Kondor 2021; Tahmasebi, Lim, and Jegelka 2020; Kishan et al. 2022), yet most GNNs cannot perceive such substructures due to the upper limit of 1-WL. Directly constructing higher-order GNNs (Maron et al. 2019a; Morris, Rattan, and Mutzel 2020; Morris et al. 2019; Balcilar et al. 2021; Kishan et al. 2022; Li et al. 2021) that exceed 1-WL test becomes the most hopeful choice, bringing problems of scalability and complexity. Due to the high computational consumption of training and inference, the limited computing resources make extensive use of higher-order GNNs a luxury. Thus directly using predefined hand-crafted substructures (Bouritsas et al. 2022; Chen et al. 2020b; Nikolentzos, Dasoulas, and Vazirgiannis 2020; Sandfelder, Vijayan, and Hamilton 2021) as additional features also becomes a highly feasible solution, but at the expense of the generalization ability of the GNNs. More flexible use of substructures (Zhao et al. 2022; Chen et al. 2020b; Nikolentzos, Dasoulas, and Vazirgiannis 2020; Sandfelder, Vijayan, and Hamilton 2021) has been researched, resulting in further improvements. At the same time, inductive coloring methods (You et al. 2021; Vignac, Loukas, and Frossard 2020; Sato, Yamada, and Kashima 2021; Zhang and Chen 2018; Veličković et al. 2019; Xu et al. 2019b) such as random coloring of specific nodes also bring certain performance improvements.

Based on the fact that solving the subgraph isomorphism problem is easier than the original graph isomorphism problem (Bevilacqua et al. 2022), we introduce subgraphs to improve the expressiveness of GNNs. According to different perspectives of subgraph extraction methods, we divide existing subgraph extraction strategies into node-based strategies and graph-based strategies. At the same time, in order to improve the expressiveness of subgraphs for the original graph, we propose a Cut subgraph which can be obtained from the original graph by continuously and selec-

[*]These authors contributed equally.

[†]Corresponding author: Hong Qu

tively removing edges. Unlike most methods using MPNN to encode subgraphs, we extend random walks to the return probabilities in subgraphs to encode the structural information of subgraphs, which reduces time complexity and improves expressiveness. Then we propose a graph neural network framework based on subgraph encoding injection called Substructure Aware Graph Neural Network (SAGNN), which greatly enhances the expressiveness of GNNs without increasing complexity. We further theoretically prove that any 1-WL GNNs equipped with any components of our framework are strictly more powerful than 1-WL. Our extensive experiments validate the state-of-the-art performance of our framework on various base model networks, tasks and datasets, especially on the graph regression task of drug constrained solubility prediction (ZINC-FULL). Our framework achieves a maximum MAE reduction of 83% compared to the base model and a maximum MAE reduction of 32% compared to the previous state-of-the-art model.

In summary, our main contributions are as follows:

- We propose a Cut subgraph which can be obtained from the original graph by continuously and selectively removing edges to help solve graph isomorphism problem. Then we extend random walks to the return probabilities in subgraphs to encode the structural information of subgraphs.
- We propose a GNN framework based on subgraph encoding injection called Substructure Aware Graph Neural Network (SAGNN), which greatly enhances the expressiveness and performance of GNNs.
- Extensive and diverse experiments demonstrate the state-of-the-art performance of our framework on various tasks and datasets[1].

Preliminaries and Related Work

Notations and Background

Let $G = (V, E, \mathbf{X})$ be a simple, undirected, connected graph with a finite set of $nodes$ V and a finite set of $edges$ E, where the node feature $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^\top$.

Graph Neural Networks

The concept of GNNs is not new (Sperduti and Starita 1997; Baskin, Palyulin, and Zefirov 1997) while the idea of using neural networks to model graph data and extract features has long existed and achieved certain results (Gori, Monfardini, and Scarselli 2005; Scarselli et al. 2008). A more modern version is graph convolutional neural network (GCN) (Kipf and Welling 2017), a graph neural network used for semi-supervised node classification, which laid the foundation for advanced Graph Neural Networks. Besides, researchers also introduce the graph neural networks to the unsupervised scheme like graph contrastive learning and node clustering (Liu et al. 2022c,d,e, 2023; Chen and Kou 2023). Based on the message passing mechanism (Gilmer et al. 2017) similar to GCN, a series of advanced graph neural networks are proposed to solve the problems of over-smoothing and over-squashing (Chen et al. 2020a; Topping et al. 2021; Xu et al. 2018b; Zeng et al. 2022).

Message Passing Neural Networks

MPNNs (Gilmer et al. 2017) provide a methodology to abstract GNNs with a unified view of message passing mechanisms. In the case of this paradigm, node-to-node information is propagated by iteratively aggregating neighboring node information to a central node. Fomally, given a node i in graph G, its hidden representation $\mathbf{h}_i^t$ at t iteration, its neighboring nodes $N(i)$, edge $\mathbf{e}_{ij}$ connecting node i to node j, a iteration of standard message passing paradigm can be expressed as:

$$\mathbf{c}_i^t = \sum_{j \in N(i)} \phi^t \left(\mathbf{h}_i^t, \mathbf{h}_j^t, \mathbf{e}_{ij} \right), \tag{1}$$

$$\mathbf{h}_i^{t+1} = \sigma^t(\mathbf{c}_i^t, \mathbf{h}_i^t), \tag{2}$$

where ϕ^t and σ^t are the $aggregate$ and $update$ function at t iteration.

Weisfeiler-Leman Algorithm

The Weisfeiler-Lehman (WL) algorithm (Weisfeiler and Leman 1968) is a computationally efficient heuristic algorithm for testing graph isomorphism proposed by Weisfeiler and Lehman. The main idea of the WL algorithm is to continuously relabel the root node through the neighbor nodes until the label converges, and to judge whether the two graphs are isomorphic by comparing the labels of the two graphs. Significantly, WL algorithm provides a theoretical basis for many further graph methods (Morris, Rattan, and Mutzel 2020; Bevilacqua et al. 2022) due to its low computational complexity. The Weisfeiler-Lehman subtree kernel (Shervashidze et al. 2011), one of the most successful kernel approaches, utilizes WL algorithm to generate node features through an iterative relabeling. Moreover, based on WL algorithm, a more hierarchical graph isomorphism testing framework (Grohe 2017) is proposed, and higher-order WL tests are also instantiated from the framework.

Limitations of MPNNs

Compared with traditional graph algorithms, GNNs exhibit better adaptability and generalization capabilities, which are mainly reflected in data-driven training methods for classification and regression. The similarity of the message passing mechanism of MPNNs to the WL algorithm makes it regarded as a neural network implementation of the WL algorithm (Gilmer et al. 2017), which brings about the problem of limited expressiveness of MPNNs. Specifically, the upper limit of the expressiveness of MPNNs is 1-WL (Xu et al. 2019a), which makes it cannot distinguish a large class of graphs (Cai, Fürer, and Immerman 1992), and have certain defects in structure perception. Both MPNNs and 2nd-order Invariant Graph Networks (Maron et al. 2019c) cannot count induced subgraphs of any connected pattern of

[1]Our implementation is available at https://github.com/BlackHalo-Drake/SAGNN-Substructure-Aware-Graph-Neural-Networks

3 or more nodes but only star-shaped patterns (Chen et al. 2020b), however such structures have a strong impact on certain downstream tasks such as functional groups in organic chemistry (Lemke 2003).

Beyond 1-WL by inductive coloring Some task-specific inductive node coloring methods (Zhang and Chen 2018; Veličković et al. 2019; Xu et al. 2019b) are proposed to improve the performance of existing GNNs, which are mainly used for tasks such as link prediction and algorithm execution. An inductive node coloring framework (You et al. 2021) is proposed to fill the gaps of coloring methods on graph-level and node-level tasks. Moreover, CLIP (Dasoulas et al. 2020) use colors to disambiguate identical node attributes and is capable of capturing structure characteristics that traditional MPNNs fail to distinguish.

Beyond 1-WL by positional encoding There are absolute positions in text and images, so explicit position encoding for them is efficient. Due to the non-Euclidean spatial characteristic of graph structures, it becomes particularly difficult to explicitly encode graph data, often resulting in loss of information (Liu et al. 2022a). Direct index encoding of all nodes requires $n!$ possible index permutations (Murphy et al. 2019), so it is difficult for neural networks to inductive generalize such encoding methods. Laplacian eigenvectors are also used for positional encoding (Dwivedi et al. 2020; Dwivedi and Bresson 2020) to encode local and global structure information of the graph, but this encoding method does not address the global sign ambiguity problem. The method using random anchor sets of nodes (You, Ying, and Leskovec 2019) has no problem of symbol ambiguity, but its random selection strategy of anchor points limits its inductive generalization ability. A more generalized positional encoding method based on random walks and Laplacian eigenvectors is proposed and instantiated as a framework (Dwivedi et al. 2021), which achieves a great performance.

Beyond 1-WL by substructure perceiving Since specific substructures are extremely important for some graph tasks, directly prior encoding the substructures (Bouritsas et al. 2022; Bodnar et al. 2021a; Barceló et al. 2021) becomes an option, which achieves state-of-the-art performance. Pattern-specific subgraphs encoding (Fey, Yuen, and Weichert 2020; Thiede, Zhou, and Kondor 2021) are also used directly for highly related downstream tasks. Similar hand-crafted substructure count information is useful in global tasks for graph kernels (Zhang et al. 2018b). Theoretical analysis (Tahmasebi, Lim, and Jegelka 2020) verifies the effectiveness of the substructure on graph tasks. G-meta (Huang and Zitnik 2020)conducts message passing on subgraphs to improve the efficiency of message passing in meta-learning. Some works (Chen et al. 2020b; Nikolentzos, Dasoulas, and Vazirgiannis 2020; Sandfelder, Vijayan, and Hamilton 2021) also focus on encoding structure information of k-hop subgraphs and injecting them into nodes for propagation.

Methodology

Generally speaking, MPNNs still cannot exceed 1-WL in terms of expressiveness with sufficient depth and width (Loukas 2020) while distinguishing larger graphs requires stronger expressiveness. Based on the fact that subgraphs are more easily to distinguish than original graphs, we generalize the problem of graph isomorphism to subgraph isomorphism. In this section we introduce our SAGNN framework, which includes (1) Subgraph Extraction Strategies, (2) Subgraph Random Walk Return Probability Encoding, and (3) Subgraph Information Injection.

Subgraph Extraction Strategies

For subgraph-based methods, the subgraph extraction strategy has a crucial impact on the expressiveness of the model. In this section, we classify subgraph extraction strategies into node-based strategies and graph-based strategies.

Node-based Strategies We define the node-based strategy as a single-node-based subgraph extraction strategy that does not require information about the entire graph. Generally speaking, the number of subgraphs of the node-based extraction strategy is equal to the number of nodes of the original graph, and the most common strategy is $EgoNetwork$. Specifically, $N(v)$ denotes the set of neighboring nodes of the root node v and a more generalized notation $N_k(v)$ denotes the set of nodes within k-hop from the root node v. Then the $Ego(v)_k$ is a k-hop $Egonetwork$ rooted at node v and its corresponding nodes are the neighbors within k-hop $N_k(v)$ of the root node v.

Graph-based Strategies Different from the node-based strategy, the graph-based strategy requires the information of the entire graph to obtain subgraphs, and the number of subgraphs obtained is not directly related to the number of nodes in the original graph. In practice, the simplest graph-based strategy is to randomly delete a single edge or delete a single node, which is difficult to stably improve the expressiveness of the model and has a poor performance on high-density strongly regular graphs. To better improve the expressiveness of GNNs, we propose the Cut subgraph, a subgraph obtained from the original graph by continuously and selectively removing edges.

Definition 1. *Formally, we define the block containing node v among b blocks as a $Cut(v)_b$ subgraph obtained by removing edges from the original graph using a specific method. In order to get $Cut(v)_b$ subgraph, we first calculate the Edge Betweenness Centrality (EBC) (Girvan and Newman 2002) of all edges in the original graph, and then continuously remove the edge with the biggest EBC until the original graph is split into b blocks.*

1. $Cut(v)_b$ is equal to the original graph G when $b = 1$. As b increases, both the size of the $Cut(v)_b$ and the level of information it contains decreases.

2. $Cut(v)_b$ is equal to the node i when b is equal to the number of nodes in the graph G and $b = i$.

3. Any $Cut(v)_b$ is a connected graph containing node v.

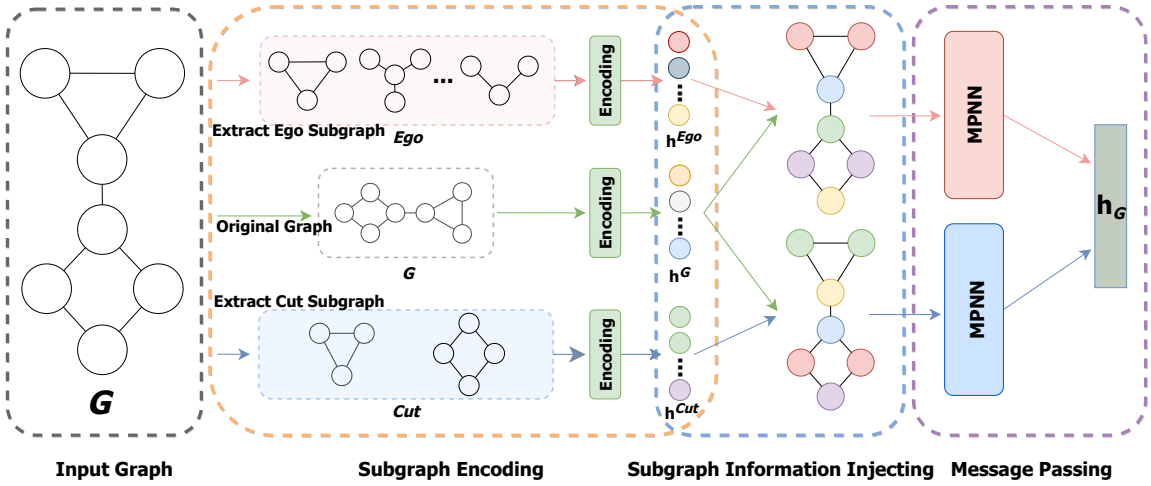

Figure 1: SAGNN's main framework. The main components are (1) Subgraph Encoding (2) Subgraph Information Injection (3) Message Passing.

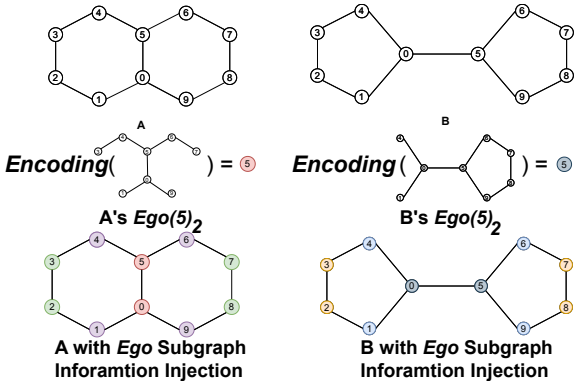

Figure 2: Two non-isomorphic graphs that cannot be distinguished by WL test but can be distinguished by a MPNN with Ego subgraph information injection.

Subgraph Random Walk Return Probability Encoding

The use of random walk for graph encoding is not new, but existing methods (Zhang et al. 2018b; Li et al. 2020; Dwivedi et al. 2021) have shortcomings in complexity or expressiveness. To address these issues, we extend random walk encoding to return probabilities in subgraphs, while reducing time complexity and improving expressiveness. Formally, the random walk return probability encoding of node v in the subgraph G_{sub} is defined as:

$$\boldsymbol{p}_v^{G_{sub}} = \left[\boldsymbol{R}^1_{G_{sub}}(v,v), \boldsymbol{R}^2_{G_{sub}}(v,v), \ldots, \boldsymbol{R}^S_{G_{sub}}(v,v)\right]^T, \tag{3}$$

where $\boldsymbol{R}^s_{G_{sub}}(v,v), s = 1, 2, \ldots, S$, is the return probability of a s-step random walk starting from the root node v in the subgraph G_{sub}. We apply the subgraph encoding to the Ego subgraph and the Cut subgraph and obtain the corresponding return probability $\boldsymbol{p}_v^{Ego}$ and $\boldsymbol{p}_v^{Cut}$.

Then we use a linear layer to encode the subgraph random walk return probabilities into a subgraph hidden representation vector $\mathbf{h}_v^{G_{sub}}$. Similarly, the subgraph hidden representation of node v corresponding to the Ego subgraph [2] and the Cut subgraph is $\mathbf{h}_v^{Ego}$ and $\mathbf{h}_v^{Cut}$.

Message Passing with Subgraph Information Injection

We concatenate our subgraph hidden representation with the initial features of nodes to obtain Ego subgraph information injection feature $\mathbf{h}_v^{E,0}$ and Cut subgraph information injection feature $\mathbf{h}_v^{C,0}$. In order to capture the global structural information, we also concatenate $\mathbf{h}_v^G$, which is the structural hidden representation of node v corresponding to the original graph G. Formally, $\mathbf{h}_v^{E,0}$ and $\mathbf{h}_v^{C,0}$ are defined as follows:

$$\mathbf{h}_v^{E,0} = [\mathbf{x}_v, \mathbf{h}_v^{Ego}, \mathbf{h}_v^G], \ \mathbf{h}_v^{C,0} = [\mathbf{x}_v, \mathbf{h}_v^{Cut}, \mathbf{h}_v^G]. \tag{4}$$

We adopt two message passing channels: Ego channel and Cut channel defined as follows,

$$\mathbf{c}_v^{E,t} = \sum_{u \in N(v)} \phi_E^t \left(\mathbf{h}_v^{E,t}, \mathbf{h}_u^{E,t}, \mathbf{e}_{vu}\right), \tag{5}$$

$$\mathbf{c}_v^{C,t} = \sum_{u \in N(v)} \phi_C^t \left(\mathbf{h}_v^{C,t}, \mathbf{h}_u^{C,t}, \mathbf{e}_{vu}\right), \tag{6}$$

$$\mathbf{h}_v^{E,t+1} = \sigma_E^t(\mathbf{c}_v^{E,t}, \mathbf{h}_v^{E,t}), \ \mathbf{h}_v^{C,t+1} = \sigma_C^t(\mathbf{c}_v^{C,t}, \mathbf{h}_v^{C,t}), \tag{7}$$

where ϕ_E^t, ϕ_C^t and σ_E^t, σ_C^t are the $aggregate$ and $update$ functions for two channels respectively at t iteration, $t = 0, \ldots, L\text{-}1$. Finally, the whole graph representation $\mathbf{h}_G$ is obtained through pooling operation defined in detail as follows:

$$\mathbf{h}_G = \mathrm{POOL}\left(\left\{\left[\mathbf{h}_v^{E,L}, \mathbf{h}_v^{C,L}\right] \mid v \in V\right\}\right), \tag{8}$$

where POOL is a global pooling function for all nodes.

[2] In our implementation, the Ego encoding of a single node is the aggregation of all Ego subgraph encodings that contain this node.

Method	MUTAG	PTC	PROTEINS	NCI1	IMDB-B
RWK (Gärtner, Flach, and Wrobel 2003)	79.2±2.1	55.9±0.3	59.6±0.1	>3 days	N/A
GK (k = 3) (Shervashidze et al. 2009)	81.4±1.7	55.7±0.5	71.4±0.3	62.5±0.3	N/A
PK (Neumann et al. 2016)	76.0±2.7	59.5±2.4	73.7±0.7	82.5±0.5	N/A
WL kernel (Shervashidze et al. 2011)	90.4±5.7	59.9±4.3	75.0±3.1	**86.0±1.8**	73.8±3.9
DCNN (Atwood and Towsley 2016)	N/A	N/A	61.3±1.6	56.6±1.0	49.1±1.4
DGCNN (Zhang et al. 2018a)	85.8±1.8	58.6±2.5	75.5±0.9	74.4±0.5	70.0±0.9
IGN (Maron et al. 2019b)	83.9±13.0	58.5±6.9	76.6±5.5	74.3±2.7	72.0±5.5
GIN (Xu et al. 2018a)	89.4±5.6	64.6±7.0	76.2±2.8	82.7±1.7	75.1±5.1
PPGNs (Maron et al. 2019a)	90.6±8.7	66.2±6.6	77.2±4.7	83.2±1.1	73.0±5.8
Natural GN (de Haan, Cohen, and Welling 2020)	89.4±1.6	66.8±1.7	71.7±1.0	82.4±1.3	73.5±2.0
GSN (Bouritsas et al. 2022)	92.2±7.5	68.2 ± 7.2	76.6±5.0	83.5±2.0	**77.8±3.3**
SIN (Bodnar et al. 2021b)	N/A	N/A	76.4±3.3	82.7±2.1	75.6±3.2
CIN (Bodnar et al. 2021a)	92.7±6.1	68.2±5.6	77.0±4.3	83.6±1.4	75.6±3.7
GIN-AK+ (Zhao et al. 2022)	91.3±7.0	67.7±8.8	77.1±5.7	85.0±2.0	75.0±4.2
ESAN-GIN (Bevilacqua et al. 2022)	91.0±7.1	69.2±6.5	77.1±4.6	83.8±2.4	77.1±3.0
DropGIN (Papp et al. 2021)	90.4±7.0	66.3±8.6	76.3±6.1	N/A	75.7±4.2
SAGIN(Ours)	**95.2±3.0**	**72.1±7.6**	**79.8±3.8**	85.3±1.7	75.9±3.8
SAPNA*(Ours)	92.0±6.8	70.3±6.5	79.3±3.6	85.0±1.1	77.4±3.7

Table 1: Test results for TUDatasets. The first section of the table includes the results of graph kernel methods, while the second includes the results of GNNs, and the third part includes the results of the GNNs boosted by our framework. The top three are highlighted by red, green, and blue.

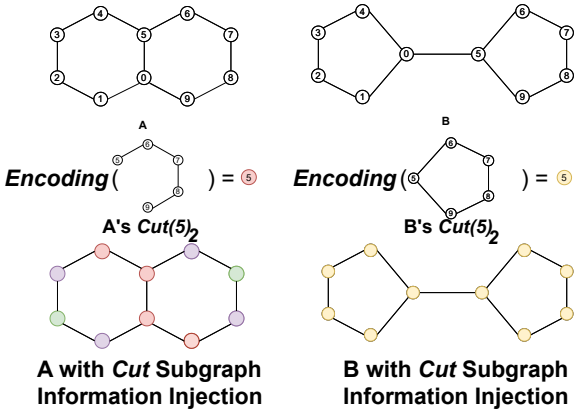

Figure 3: Two non-isomorphic graphs that cannot be distinguished by WL test but can be distinguished by a MPNN with Cut subgraph information injection.

Expressiveness Analysis

Proposition 1. *With Ego subgraph information injection, an 1-WL MPNN is strictly more powerful than 1-WL Test.*

Proof. This proof for Proposition 1 mainly consists of two parts. We first prove that if two graphs are judged to be isomorphic by the 1-WL MPNN with Ego subgraph information injection, then the 1-WL test also must judge the two graphs to be isomorphic, which proves that 1-WL MPNN with Ego subgraph information injection is at least as powerful as 1-WL. Then we give two graphs that are judged to be isomorphic by the 1-WL, but not isomorphic from the 1-WL MPNN with Ego subgraph information injection thus proving the 1-WL MPNN with Ego subgraph information injection is more powerful than 1-WL.

Assuming that there are graphs A and B that are judged to be isomorphic by 1-WL MPNN with Ego subgraph information injection. The sets of A and B generated by 1-WL MPNN with Ego subgraph information injection after t iterations $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{h}_i^{h,(0)}))|\mathbf{h}_i^{h,(0)} \in \mathcal{V}_A\right\}$ and $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{h}_i^{h,(0)}))|\mathbf{h}_i^{h,(0)} \in \mathcal{V}_B\right\}$ are identical. So there exists an ordering of nodes $\mathbf{h}_1^{h,(0),A}, \mathbf{h}_2^{h,(0),A}, \cdots, \mathbf{h}_n^{h,(0),A}$ and $\mathbf{h}_1^{h,(0),B}, \mathbf{h}_2^{h,(0),B}, \cdots, \mathbf{h}_n^{h,(0),B}$, such that for any subgraphs order $i = 1, 2, \cdots, n$, $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{h}_i^{h,(0),A}))\right\}$ and $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{h}_i^{h,(0),B}))\right\}$ are identical. If two node feature with Ego subgraph information injection are identical, their original feature are identical, there exists an ordering of nodes $\mathbf{x}_1^A, \mathbf{x}_2^A, \cdots, \mathbf{x}_n^A$ and $\mathbf{x}_1^B, \mathbf{x}_2^B, \cdots, \mathbf{x}_n^B$, such that for any node order $i = 1, 2, \cdots, n$, $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{x}_v^A))\right\}$ and $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{x}_v^A))\right\}$ are identical. Finally it can be deduced that the sets $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{x}_v^A))|v \in \mathcal{V}_A\right\}$ and $\left\{\mathbf{HASH}^t(\mathcal{N}(\mathbf{x}_v^A))|v \in \mathcal{V}_B\right\}$ are identical which means 1-WL MPNN with Ego subgraph information injection is at least as powerful as 1-WL test.

In Figure 2, two graphs are judged to be isomorphic by the 1-WL test but not isomorphic by the 1-WL MPNN with Ego subgraph information injection, which demonstrates 1-WL MPNN with Ego subgraph information injection is more powerful than 1-WL test.

Proposition 2. *With Cut subgraph information injection, an 1-WL MPNN is strictly more powerful than 1-WL Test.*

Proof. Similarly to previous proof, this proof for Proposi-

Method	EXP (ACC)	SR25 (ACC)	ZINC (MAE)	ZINC-FULL (MAE)	MolPCBA (AP)	MolHIV (ROC)
HIMP (Fey, Yuen, and Weichert 2020)	N/A	N/A	0.151±0.006	0.036±0.002	N/A	78.80±0.82
PNA (Corso et al. 2020)	N/A	N/A	0.188±0.004	N/A	28.38±0.35	79.05±1.32
GSN (Bouritsas et al. 2022)	N/A	N/A	0.108±0.018	N/A	N/A	77.99±1.00
CIN (Bodnar et al. 2021a)	N/A	N/A	0.079±0.006	0.022±0.002	N/A	**80.94±0.57**
GIN (Xu et al. 2018a)	50%	6.67%	0.163±0.004	0.088±0.002	26.82±0.06	78.81±1.19
GIN-AK+ (Zhao et al. 2022)	**100%**	6.67%	0.080±0.001	N/A	**29.30±0.44**	79.61±1.19
ESAN-GIN (Bevilacqua et al. 2022)	**100%**	N/A	0.102±0.003	N/A	N/A	78.00±1.42
SAGIN(Ours)	**100%**	**100%**	**0.072±0.001**	**0.016±0.002**	28.53±0.30	80.64±0.42
PNA* (Corso et al. 2020)	50%	6.67%	0.140±0.006	N/A	27.37±0.09	79.05±1.02
PNA*-AK+ (Zhao et al. 2022)	**100%**	6.67%	0.085±0.003	N/A	28.85±0.06	78.80±1.53
SAPNA*(Ours)	**100%**	**100%**	0.073±0.001	0.016±0.003	27.84±0.03	79.44±1.44

Table 2: Test results for expressiveness datasets and large scale datasets. The first section of the table includes the results of some specific methods, while the other sections include results of base model and that have been boosted by different methods. The top three are highlighted by red, green, and blue.

Dataset	ZINC (MAE)		SR25 (ACC)
Layers	Valid	Test	Test
2	0.1006±0.0024	0.0822±0.0063	100%
4	0.0974±0.0045	0.0778±0.0001	100%
6	0.0825±0.0023	0.0721±0.0025	100%
8	0.0897±0.0023	0.0782±0.0038	100%
10	0.0945±0.0020	0.0790±0.0057	100%

Table 3: Hyperparameter study of layers on ZINC and SR25 datasets.

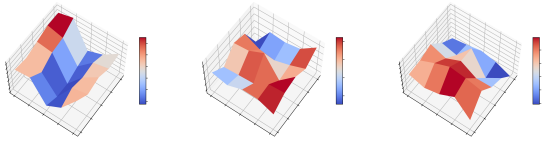

Figure 4: Hyperparameter study on ZINC, PROTEINS and MOLHIV.

tion 2 mainly consists of two parts. It is easy to prove that if two graphs are judged to be isomorphic by the 1-WL MPNN with Cut subgraph information injection, then the 1-WL test also must judge the two graphs to be isomorphic, which proves that 1-WL MPNN with Cut subgraph information injection is at least as powerful as 1-WL. And in Figure 3, two graphs are judged to be isomorphic by the 1-WL test but not isomorphic by the 1-WL MPNN with Cut subgraph information injection, which demonstrates 1-WL MPNN with Cut subgraph information injection is more powerful than 1-WL test.

Experiments and Discussion

Experimental Setup

Datasets and tasks We use EXP (Abboud et al. 2021) and SR25 (Balcilar et al. 2021) datasets to verify the expressiveness improvement of our framework for MPNNs, where EXP contains 600 pairs of 1-WL failed graphs and SR25 contains 15 3-WL failed strongly regular graphs. For performance on real-world tasks, we use three kinds of datasets of different scales for validation. The first kind is small-scale real-world datasets TUDataset (Morris et al. 2020), which includes MUTAG, PTC, PROTEINS, NCI1 and IMDB from biology, chemistry and social networks. The second is a large-scale molecular benchmark from the zinc database from the ZINC database (Sterling and Irwin 2015), which includes ZINC (12K graphs) (Dwivedi et al. 2020) and ZINC-FULL (250k graphs) (Gómez-Bombarelli et al. 2018; Jin, Barzilay, and Jaakkola 2018; Zhang et al. 2018a). Specifically, ZINC and ZINC-FULL are two graph regression task datasets for drug constrained solubility prediction. The last is a molecular large-scale dataset from the Open Graph Benchmark (Hu et al. 2020, 2021) which includes OGBG-MOLHIV (41k graphs) and OGBG-PCBA (437k graphs). Experiments are repeated 10 times on expressiveness datasets and 3 times on large scale datasets to calculate mean and standard derivation. For TUDataset, we perform 10-fold cross-validation and report the average and standard deviation of validation accuracy across the 10 folds within the cross-validation.

Baselines We equip our framework on GIN (Xu et al. 2018a) and PNA*[3] (Corso et al. 2020) models to verify the effectiveness of our framework. For some small real-world datasets: MUTAG, PTC, PROTEINS, NCI1 and IMDB from TUDatset, we directly reference the existing experiment results for RWK (Gärtner, Flach, and Wrobel 2003), GK (Shervashidze et al. 2009), PK (Neumann et al. 2016), WL kernel (Shervashidze et al. 2011), DCNN (Atwood and Towsley 2016), DGCNN (Zhang et al. 2018a), IGN (Maron et al. 2019b), GIN (Xu et al. 2018a), PPGNs (Maron et al. 2019a), Natural GN (de Haan, Cohen, and Welling 2020), GSN (Bouritsas et al. 2022), SIN (Bodnar et al. 2021b), CIN (Bodnar et al. 2021a)

[3]PNA* is a variant of PNA that uses degree to scale embeddings to encoding degree and concatenate to node embeddings .

Methods	MUTAG	PTC	PROTEINS	NCI1	IMDB-B
SAGIN	95.23±2.99	72.07±7.64	79.79±3.75	85.28±1.68	75.90±3.84
SAGIN w/o Ego	94.68±4.38	71.21±6.95	78.53±2.49	84.94±1.53	75.30±3.97
SAGIN w/o Cut	94.12±3.09	70.91±7.47	78.98±2.44	84.90±1.37	75.50±3.89
SAGIN w/o Global	93.10±5.62	72.38±5.82	79.54±3.67	84.60±1.46	75.40±3.75

Table 4: Ablation study on small scale datasets.

Methods	EXP (ACC)	SR25 (ACC)	ZINC (MAE)	ZINC-FULL (MAE)	MOLPCBA (AP)	MOLHIV (ROC)
SAGIN	100%	100%	0.072±0.002	0.016±0.002	28.53±0.30	80.64±1.28
SAGIN w/o Ego	100%	100%	0.094±0.002	0.028±0.001	27.14±0.62	76.97±2.26
SAGIN w/o Cut	100%	100%	0.081±0.002	0.031±0.001	27.57±0.05	77.78±1.92
SAGIN w/o Global	100%	100%	0.079±0.001	0.019±0.001	28.34±0.16	80.13±0.68

Table 5: Ablation study on expressiveness datasets and large scale datasets.

models from (Bodnar et al. 2021a), ESAN (Bevilacqua et al. 2022), k-Reconstruction GNNs (Cotta, Morris, and Ribeiro 2021) and DropGNN (Papp et al. 2021). We reference other real world datasets results for PNA* (Corso et al. 2020), CIN (Bodnar et al. 2021a), GNN-AK+ (Zhao et al. 2022), GraphSNN (Wijesinghe and Wang 2022), MPGNNs-LSPE (Dwivedi et al. 2021), GatedGCN (Dwivedi et al. 2020), HIMP (Fey, Yuen, and Weichert 2020) in their literature for comparison.

Results and Discussion

Small real-world datasets Table 1 presents the results of our framework and other competitive models on small real-world datasets. Our framework achieves state-of-the-art performance on all TUDatset datasets, demonstrating the great advantage of our framework in structural awareness.

Expressiveness datasets Table 2 presents the performance of our framework on two expressiveness datasets, where our framework achieves 100% accuracy on both the EXP and SR25 datasets with different base models. Compared with other methods, our framework shows a superior boosting effect, which can obtain more than 3-WL capabilities with the base models of less than 1-WL capabilities.

Large scale datasets We further validate the performance of our framework on large-scale datasets and achieve significant performance improvements with different base models, which can be viewed in Table 2. Our framework achieves a maximum MAE reduction of 83% compared to the base model and a maximum MAE reduction of 32% compared to the previous state-of-the-art model on the graph regression task of drug constrained solubility prediction (ZINC-FULL). On other relative structure-insensitive graph-level tasks, our framework still outperforms other frameworks and competitive models on multiple datasets.

Hyperparameter effect We first conduct an ablation study to analyze the impact of the values of Ego and Cut on model performance on different tasks and different data schema. Figure 4 visualizes the results of our hyperparameter experiments on three datasets. All three datasets show the same performance trend, that is, the best performance in the case of Ego=3, Cut=4, which shows our model's great hyperparameter stability. Our models perform optimally with relatively similar hyperparameters under different data structures and different tasks, which allows for less hyperparameter tuning when deploying our model on real world tasks. Then we investigate the impact of the number of layers on SR25 and ZINC. As is shown in Table 3, the accuracy achieves 100% when the number of layers is only 2 and remains 100% with the increase of the number of layers, which illustrates that the expressiveness of our model does not depend on model depth. But on real-world dataset (ZINC), our model needs proper layer numbers to achieve good performance.

Ablation study To better demonstrate the capabilities of our components, we conduct ablation study on all datasets of our experiments. Table 4 and Table 5 present all results of our ablation experiments, where different datasets exhibit different dependencies on different components. For example, the ZINC dataset pays more attention to low-level local structure information, so the performance loss of ablating Cut component is not as large as ablating Ego component.

Conclusion

In this paper, we first propose a Cut subgraph which can be obtained from the original graph by continuously and selectively removing edges to help solving graph isomorphism problem. Then we further propose a GNN framework called Substructure Aware Graph Neural Network, which enhances the expressiveness and performance of GNNs by encoding subgraphs at different levels and injecting information into nodes. Our extensive and diverse experiments demonstrate the state-of-the-art performance of our framework on various tasks and datasets.

Acknowledgements

This work was supported by the National Science Foundation of China under Grant 61976043, and in part by the Science and technology support program of Sichuan Province under Grant 2022YFG0313.

References

Abboud, R.; Ceylan, I. I.; Grohe, M.; and Lukasiewicz, T. 2021. The Surprising Power of Graph Neural Networks with Random Node Initialization. In *Proc. of IJCAI*.

Atwood, J.; and Towsley, D. 2016. Diffusion-convolutional neural networks. *Proc. of NeurIPS*.

Balcilar, M.; Héroux, P.; Gauzere, B.; Vasseur, P.; Adam, S.; and Honeine, P. 2021. Breaking the limits of message passing graph neural networks. In *Proc. of ICML*.

Barceló, P.; Geerts, F.; Reutter, J.; and Ryschkov, M. 2021. Graph neural networks with local graph parameters. *Proc. of NeurIPS*.

Baskin, I. I.; Palyulin, V. A.; and Zefirov, N. S. 1997. A neural device for searching direct correlations between structures and properties of chemical compounds. *Journal of chemical information and computer sciences*.

Battaglia, P.; Pascanu, R.; Lai, M.; Jimenez Rezende, D.; et al. 2016. Interaction networks for learning about objects, relations and physics. *Proc. of NeurIPS*.

Bevilacqua, B.; Frasca, F.; Lim, D.; Srinivasan, B.; Cai, C.; Balamurugan, G.; Bronstein, M. M.; and Maron, H. 2022. Equivariant subgraph aggregation networks. In *Proc. of ICLR*.

Bodnar, C.; Frasca, F.; Otter, N.; Wang, Y. G.; Liò, P.; Montufar, G. F.; and Bronstein, M. 2021a. Weisfeiler and lehman go cellular: Cw networks. *Proc. of NeurIPS*.

Bodnar, C.; Frasca, F.; Wang, Y.; Otter, N.; Montufar, G. F.; Lio, P.; and Bronstein, M. 2021b. Weisfeiler and lehman go topological: Message passing simplicial networks. In *Proc. of ICML*.

Bouritsas, G.; Frasca, F.; Zafeiriou, S. P.; and Bronstein, M. 2022. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Cai, J.-Y.; Fürer, M.; and Immerman, N. 1992. An optimal lower bound on the number of variables for graph identification. *Combinatorica*.

Chen, J.; and Kou, G. 2023. Attribute and Structure preserving Graph Contrastive Learning. In *Proc. of AAAI*.

Chen, M.; Wei, Z.; Huang, Z.; Ding, B.; and Li, Y. 2020a. Simple and Deep Graph Convolutional Networks. In *Proc. of ICML*.

Chen, Z.; Chen, L.; Villar, S.; and Bruna, J. 2020b. Can graph neural networks count substructures? *Proc. of NeurIPS*.

Corso, G.; Cavalleri, L.; Beaini, D.; Liò, P.; and Veličković, P. 2020. Principal neighbourhood aggregation for graph nets. *Proc. of NeurIPS*.

Cotta, L.; Morris, C.; and Ribeiro, B. 2021. Reconstruction for powerful graph representations. *Proc. of NeurIPS*.

Dasoulas, G.; Santos, L. D.; Scaman, K.; and Virmaux, A. 2020. Coloring Graph Neural Networks for Node Disambiguation. In *Proc. of IJCAI*.

de Haan, P.; Cohen, T. S.; and Welling, M. 2020. Natural graph networks. *Proc. of NeurIPS*.

Dwivedi, V. P.; and Bresson, X. 2020. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*.

Dwivedi, V. P.; Joshi, C. K.; Laurent, T.; Bengio, Y.; and Bresson, X. 2020. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*.

Dwivedi, V. P.; Luu, A. T.; Laurent, T.; Bengio, Y.; and Bresson, X. 2021. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*.

Fey, M.; Yuen, J.-G.; and Weichert, F. 2020. Hierarchical inter-message passing for learning on molecular graphs. *arXiv preprint arXiv:2006.12179*.

Gainza, P.; Sverrisson, F.; Monti, F.; Rodola, E.; Boscaini, D.; Bronstein, M.; and Correia, B. 2020. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*.

Gärtner, T.; Flach, P.; and Wrobel, S. 2003. On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *Proc. of ICML*.

Girvan, M.; and Newman, M. E. 2002. Community structure in social and biological networks. *Proceedings of the national academy of sciences*.

Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; and Aspuru-Guzik, A. 2018. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*.

Gori, M.; Monfardini, G.; and Scarselli, F. 2005. A new model for learning in graph domains. In *Proc. of IJCNN*.

Grohe, M. 2017. *Descriptive complexity, canonisation, and definable graph structure theory*. Cambridge University Press.

Hu, W.; Fey, M.; Ren, H.; Nakata, M.; Dong, Y.; and Leskovec, J. 2021. OGB-LSC: A Large-Scale Challenge for Machine Learning on Graphs. In *Proc. of NeurIPS*.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Proc. of NeurIPS*.

Huang, K.; and Zitnik, M. 2020. Graph meta learning via local subgraphs. *Proc. of NeurIPS*.

Jin, W.; Barzilay, R.; and Jaakkola, T. 2018. Junction tree variational autoencoder for molecular graph generation. In *Proc. of ICML*.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proc. of ICLR*.

Kishan, K.; Li, R.; Cui, F.; and Haake, A. R. 2022. Predicting biomedical interactions with higher-order graph convolutional networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Lemke, T. L. 2003. *Review of organic functional groups: introduction to medicinal organic chemistry*. Lippincott Williams & Wilkins.

Li, H.; Zhang, L.; Zhang, D.; Fu, L.; Yang, P.; and Zhang, J. 2022. TransVLAD: Focusing on Locally Aggregated Descriptors for Few-Shot Learning. In *Proc. of ECCV*.

Li, J.; Peng, H.; Cao, Y.; Dou, Y.; Zhang, H.; Yu, P.; and He, L. 2021. Higher-order attribute-enhancing heterogeneous graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*.

Li, P.; Wang, Y.; Wang, H.; and Leskovec, J. 2020. Distance encoding: Design provably more powerful neural networks for graph representation learning. *Proc. of NeurIPS*.

Liu, C.; Yang, Y.; Ding, Y.; and Lu, H. 2022a. EDEN: A Plug-in Equivariant Distance Encoding to Beyond the 1-WL Test.

Liu, Y.; Long, C.; Zhang, Z.; Liu, B.; Zhang, Q.; Yin, B.; and Yang, X. 2022b. Explore Contextual Information for 3D Scene Graph Generation. *IEEE Transactions on Visualization and Computer Graphics*.

Liu, Y.; Tu, W.; Zhou, S.; Liu, X.; Song, L.; Yang, X.; and Zhu, E. 2022c. Deep Graph Clustering via Dual Correlation Reduction. In *Proc. of AAAI*, volume 36, 7603–7611.

Liu, Y.; Xia, J.; Zhou, S.; Wang, S.; Guo, X.; Yang, X.; Liang, K.; Tu, W.; Li, Z. S.; and Liu, X. 2022d. A Survey of Deep Graph Clustering: Taxonomy, Challenge, and Application. *arXiv preprint arXiv:2211.12875*.

Liu, Y.; Yang, X.; Zhou, S.; and Liu, X. 2022e. Simple Contrastive Graph Clustering. *arXiv preprint arXiv:2205.07865*.

Liu, Y.; Yang, X.; Zhou, S.; Liu, X.; Wang, Z.; Liang, K.; Tu, W.; Li, L.; Duan, J.; and Chen, C. 2023. Hard Sample Aware Network for Contrastive Deep Graph Clustering. In *Proc. of AAAI*.

Loukas, A. 2020. What graph neural networks cannot learn: depth vs width. In *Proc. of ICLR*.

Maron, H.; Ben-Hamu, H.; Serviansky, H.; and Lipman, Y. 2019a. Provably powerful graph networks. *Proc. of NeurIPS*.

Maron, H.; Ben-Hamu, H.; Shamir, N.; and Lipman, Y. 2019b. Invariant and Equivariant Graph Networks. In *Proc. of ICLR*.

Maron, H.; Fetaya, E.; Segol, N.; and Lipman, Y. 2019c. On the universality of invariant networks. In *Proc. of ICML*.

Morris, C.; Kriege, N. M.; Bause, F.; Kersting, K.; Mutzel, P.; and Neumann, M. 2020. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*.

Morris, C.; Rattan, G.; and Mutzel, P. 2020. Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings. *Proc. of NeurIPS*.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2019. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proc. of AAAI*.

Murphy, R.; Srinivasan, B.; Rao, V.; and Ribeiro, B. 2019. Relational pooling for graph representations. In *Proc. of ICML*.

Neumann, M.; Garnett, R.; Bauckhage, C.; and Kersting, K. 2016. Propagation kernels: efficient graph kernels from propagated information. *Machine Learning*.

Nikolentzos, G.; Dasoulas, G.; and Vazirgiannis, M. 2020. k-hop graph neural networks. *Neural Networks*.

Pan, Z.; Sharma, A.; Hu, J. Y.-C.; Liu, Z.; Li, A.; Liu, H.; Huang, M.; and Geng, T. T. 2023. Ising-Traffic: Using Ising Machine Learning to Predict Traffic Congestion under Uncertainty. In *Proc. of AAAI*.

Papp, P. A.; Martinkus, K.; Faber, L.; and Wattenhofer, R. 2021. Dropgnn: random dropouts increase the expressiveness of graph neural networks. *Proc. of NeurIPS*.

Sandfelder, D.; Vijayan, P.; and Hamilton, W. L. 2021. Ego-gnns: Exploiting ego structures in graph neural networks. In *Proc. of ICASSP*.

Sato, R.; Yamada, M.; and Kashima, H. 2021. Random features strengthen graph neural networks. In *Proc. of SDM*.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *IEEE transactions on neural networks*.

Shervashidze, N.; Schweitzer, P.; Van Leeuwen, E. J.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*.

Shervashidze, N.; Vishwanathan, S.; Petri, T.; Mehlhorn, K.; and Borgwardt, K. 2009. Efficient graphlet kernels for large graph comparison. In *Proc. of AISTATS*.

Sperduti, A.; and Starita, A. 1997. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*.

Sterling, T.; and Irwin, J. J. 2015. ZINC 15–ligand discovery for everyone. *Journal of chemical information and modeling*.

Tahmasebi, B.; Lim, D.; and Jegelka, S. 2020. Counting substructures with higher-order graph neural networks: Possibility and impossibility results. *arXiv preprint arXiv:2012.03174*.

Thiede, E.; Zhou, W.; and Kondor, R. 2021. Autobahn: Automorphism-based graph neural nets. *Proc. of NeurIPS*.

Topping, J.; Giovanni, F. D.; Chamberlain, B. P.; Dong, X.; and Bronstein, M. M. 2021. Understanding over-squashing and bottlenecks on graphs via curvature. *CoRR*.

Veličković, P.; Ying, R.; Padovano, M.; Hadsell, R.; and Blundell, C. 2019. Neural Execution of Graph Algorithms. In *Proc. of ICLR*.

Vignac, C.; Loukas, A.; and Frossard, P. 2020. Building powerful and equivariant graph neural networks with structural message-passing. *Proc. of NeurIPS*.

Wang, H.; Zhao, M.; Xie, X.; Li, W.; and Guo, M. 2019. Knowledge graph convolutional networks for recommender systems. In *Proc. of WWW*.

Wang, L.; and Chen, L. 2023. FTSO: Effective NAS via First Topology Second Operator. *Preprints*.

Wang, L.; Gong, Y.; Ma, X.; Wang, Q.; Zhou, K.; and Chen, L. 2022. IS-MVSNet: Importance Sampling-Based MVSNet. In *Proc. of ECCV*, 668–683. Springer.

Wang, L.; Gong, Y.; Wang, Q.; Zhou, K.; and Chen, L. 2023. Flora: dual-Frequency LOss-compensated ReAl-time monocular 3D video reconstruction. In *Proc. of AAAI*.

Weisfeiler, B.; and Leman, A. 1968. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series*.

Wijesinghe, A.; and Wang, Q. 2022. A New Perspective on" How Graph Neural Networks Go Beyond Weisfeiler-Lehman?". In *Proc. of ICLR*.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018a. How Powerful are Graph Neural Networks? In *Proc. of ICLR*.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019a. How Powerful are Graph Neural Networks? In *Proc. of ICLR*.

Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.; and Jegelka, S. 2018b. Representation Learning on Graphs with Jumping Knowledge Networks. In *Proc. of ICML*.

Xu, K.; Li, J.; Zhang, M.; Du, S. S.; Kawarabayashi, K.-i.; and Jegelka, S. 2019b. What Can Neural Networks Reason About? In *Proc. of ICLR*.

You, J.; Gomes-Selman, J. M.; Ying, R.; and Leskovec, J. 2021. Identity-aware Graph Neural Networks. In *Proc. of AAAI*.

You, J.; Ying, R.; and Leskovec, J. 2019. Position-aware graph neural networks. In *Proc. of ICML*.

Zeng, D.; Zhou, L.; Liu, W.; Qu, H.; and Chen, W. 2022. A Simple Graph Neural Network via Layer Sniffer. In *Proc. of ICASSP*.

Zhang, D.; Li, C.; Li, H.; Huang, W.; Huang, L.; and Zhang, J. 2022. Rethinking Alignment and Uniformity in Unsupervised Image Semantic Segmentation. *arXiv preprint arXiv:2211.12875*.

Zhang, M.; and Chen, Y. 2018. Link prediction based on graph neural networks. *Proc. of NeurIPS*.

Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018a. An end-to-end deep learning architecture for graph classification. In *Proc. of AAAI*.

Zhang, Z.; Wang, M.; Xiang, Y.; Huang, Y.; and Nehorai, A. 2018b. Retgk: Graph kernels based on return probabilities of random walks. *Proc. of NeurIPS*.

Zhao, L.; Jin, W.; Akoglu, L.; and Shah, N. 2022. From Stars to Subgraphs: Uplifting Any GNN with Local Structure Awareness. In *Proc. of ICLR*.