Seminar report: *"Equivariant Subgraph Aggregation Networks"**

Supreet Sharma[1]

[1]RWTH Aachen University

*The following report is based on [Beatrice Bevilacqua, 2021].

Contents

0.1 Introduction

0.1.1 Definitions

Definition 1 (Graphs)**.** Graphs with n vertices are defined as $G = (A, X)$. Here, $A \in \mathbb{R}^{n \times n}$ is the graph's Adjacency matrix and $X \in \mathbb{R}^{n \times d}$ is node feature matrix (each node feature vector is of dimension d).

Definition 2 (Neighbourhood)**.** The neighbourhood of a vertex V, in a Graph G is a subgraph containing nodes that are adjacent to V or that are connected to V directly and all the edges connecting them.

Definition 3 (Mutliset or Bags of subgraphs)**.** A Bag or Multiset of subgraphs is defined by $S_G = \{\!\{G_1, \ldots, G_m\}\!\}$. Here G is the original graph and it can be represented by a multiset (S_G) of its constituent subgraphs $G_1, ..., G_m$.

Definition 4 (Symmetric Group)**.** This report will be introducing two type of Symmetric group, *Subgraph Symmetry*, S_n and *Set Symmetry*, S_m. S_n in the is a multiset of all the permutations of nodes in a graph. And S_m is the permutation of all subgraphs in a multiset or bags of Subgraphs (S_G - See above definition).

Definition 5 (Isomorphic Graphs)**.** Two Graphs G_1 and G_2 are isomorphic if,

1. The number of edges and vertices in both graphs are same.

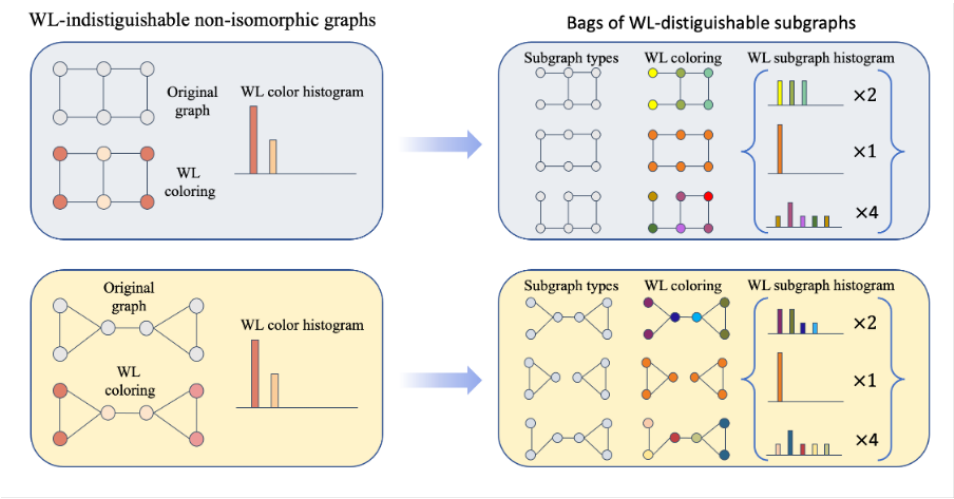

Figure 1: **Left:** WL Graph isomorphism incapable of distinguishing a basic pair of graphs. **Right:** Bags of Sub-graphs computed using Edge-Deleted policy, which can distinguish the two graphs successfully. From Beatrice Bevilacqua [2021].

2. The edge connectivity of both graphs is same.

Definition 6 (Weisfeiler-Lehman Test)**.** The Weisfeiler-Lehman Isomorphism Test (WL) [Weisfeiler and Leman, 1968] is a hierarchy of polynomial-time algorithms for determining if graphs are isomorphic. The k-WL test iteratively recolors k-tuples of vertices of a graph at each step as per some neighborhood aggregation rules. It stops once we reach a stable coloring.

Definition 7 (Graph neural networks)**.** Graph Neural Networks (GNN), are a class of neural networks that process graph data. Message Passing Neural Network (MPNN) [Justin Gilmer, 2017] is one of the popular architecture under GNN. MPNN are defined by several message passing layers, which is expressed in [Christopher Morris and Grohe, 2019] as,

$$x_v^t = W_1^t x_v^{t\text{-}1} + W_2^t \sum_{u \sim v} x_u^{t\text{-}1} + b^t$$

Here, x_v^t is the feature of node v after the application of t^{th} layer, W_i, b are the learn-able parameters and v are all the nodes in the neighbourhood of u.

Definition 8 (Expressive power of GNNs)**.** It defines the ability of a GNN model to distinguish non-isomorphic graphs.

Definition 9 (Symmetry Group and Group actions)**.** Michael M. Bronstein [2021] introduced the concept of *Geometic Priors* according to which an input, for example an image or a graph, is a signal in some *domain* Ω, a 2-Dimensional Grid (See Figure 2). The *symmetry group* $\mathfrak{G}$, describes the structure of the domain, Ω, that is all the permutations of the input. For example a set of all 2D translations for an input image. The elements of the group, is referred as Group actions ($\mathfrak{g} \in \mathfrak{G}$), for example a group action is one of the element in a set of all 2D translations of an input image. *Symmetry group* is defined by *Group representation*, $\rho(\mathfrak{G})$ in signal space $\mathcal{X}(\Omega)$.

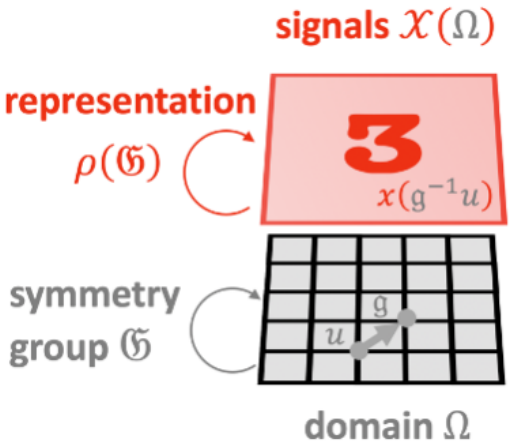

Figure 2: Illustrations of Geometric Deep Learning: *signal space* $\mathcal{X}(\Omega)$, *domain* Ω, Symmetries of the domain Ω, given by group $\mathfrak{G}$, acts on signals $x \in \mathcal{X}(\Omega)$ through group representations $\rho(\mathfrak{g})$, restricts the structure of hypothesis class. From Michael M. Bronstein [2021].

Definition 10 (Invariance and equivariance)**. Invariant** functions are the ones that are not affected by the action of a group, i.e., $f(\rho(\mathfrak{g})x) = f(x)$ for any $g \in \mathfrak{G}$. Contrastingly, in **Equivariant** functions the output is transformed in the same way as the input, i.e., $f(\rho(\mathfrak{g})x) = \rho(\mathfrak{g})f(x)$ for any $g \in \mathfrak{G}$.

0.1.2 Limitations of MPNNs

[Christopher Morris and Grohe, 2019] and [Keyulu Xu and Jegelka, 2019] presented analysis on the expressive power of MPNNs. The study concluded that MPNNs despite all thier populairty are at most as expressive as the 1-WL Graph isomorphism test. Moreover, the study states that, WL test cannot distinguish simple pair of graphs as shown in Figure 1. This inspired a quest for lot of new GNN algorithms with improved expressive power.

0.1.3 Other Powerful Architectures

There have been several recent works that proposed some powerful architectures with better expressiveness then MPNNs.
One of the method is k-WL tests ([Christopher Morris and Grohe, 2019], [Christopher Morris, 2020]) which is compromise between expressivity and complexity. However, it gets really difficult to implement anything above 2nd order networks. Other possible approaches involves using structural encoding for nodes and edges with standard MPNN method. Other alternative approaches includes MPNNs with structural encodings for nodes and edges (for example, clique or cycle counts) [Giorgos Bouritsas and Bronstein, 2022] or lift graphs into cell complexes [Cristian Bodnar and Bronstein, 2021a]. Though, there is a pre-computation step for all these approaches which may get very expensive.

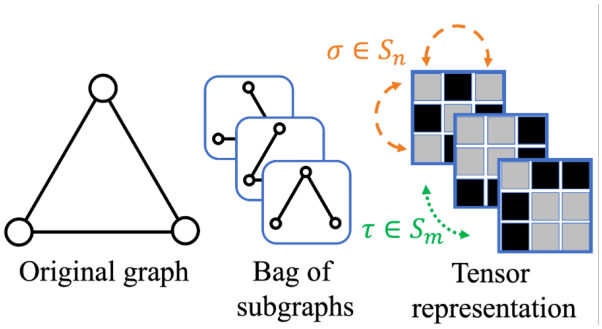

Figure 3: Symmetry Structure of subgraphs using edge deletion policy and m = 3. The subgraphs are represented by $m \times n \times n$ tensor $\mathcal{A}$. $\sigma \in S_m$ permutes on nodes of subgraphs and $\tau \in S_n$ permutes on the subgraphs. From Michael M. Bronstein [2021].

0.2 ESAN Framework

In contrast to MPNN, [Beatrice Bevilacqua, 2021] proposes *Equivariant Subgraph Aggregation Networks* (ESAN), which has better expressive power. Finding distinguishable subgraphs between two distinct graphs is at the core of this approach. Similar to WL test which encodes multiset of *node colors*, ESAN opts for encoding bags of subgraphs. Each graph is represented as a bag or multiset of its subgraph. The subgraphs are chosen by some predefined policy, such as removing an edge from the graph, as illustrated in Figure 1.
The framework constitutes of the following,

1. Architecture to process multi-set of sub-graphs
2. Sub-graph Selection Policies

0.2.1 Bag of Graphs Encoder Architecture

This section outlines a Neural Network Architecture for processing multi-sets of subgraphs to produce a final graph representation. [Beatrice Bevilacqua, 2021] presents an equivariant architecture called DSS-GNN as well its variant DS-GNN.

0.2.1.1 Symmetry group for sets of graphs

The bags of sub-graphs of G, S_G (with n nodes and m subgraphs) can be represented as $(\mathcal{A}, \mathcal{X}) \in \mathbb{R}^{n\times n\times m} \times \mathbb{R}^{n\times d\times m}$, here $\mathcal{A} \in \mathbb{R}^{n\times n\times m}$ corresponds to m Adjacency matrices and $\mathcal{X} \in \mathbb{R}^{n\times d\times m}$ is a set of m feature matrices for m subgraphs and d is the length of node feature vector.
Subgraph symmetries which is the permutation of nodes in the original graph, is represented by symmetric group S_n. And it acts on (A,X) as following,

$$(\sigma \cdot A)_{ij} = A_{\sigma^{-1}(i)\sigma^{-1}(j)} \text{ and } (\sigma \cdot X)_{il} = X_{\sigma^{-1}(i)l},\ \sigma \in S_n$$

The permutation of Subgraphs in S_G, which is *set symmetries*, given by symmetric group S_m and it acts on $(\mathcal{A}, \mathcal{X})$ as,

$$(\tau \cdot \mathcal{A})_{ijk} = \mathcal{A}_{ij\tau^{-1}(k)} \text{ and } (\tau \cdot \mathcal{X})_{ilk} = \mathcal{X}_{il\tau^{-1}(k)},\ \tau \in S_m$$

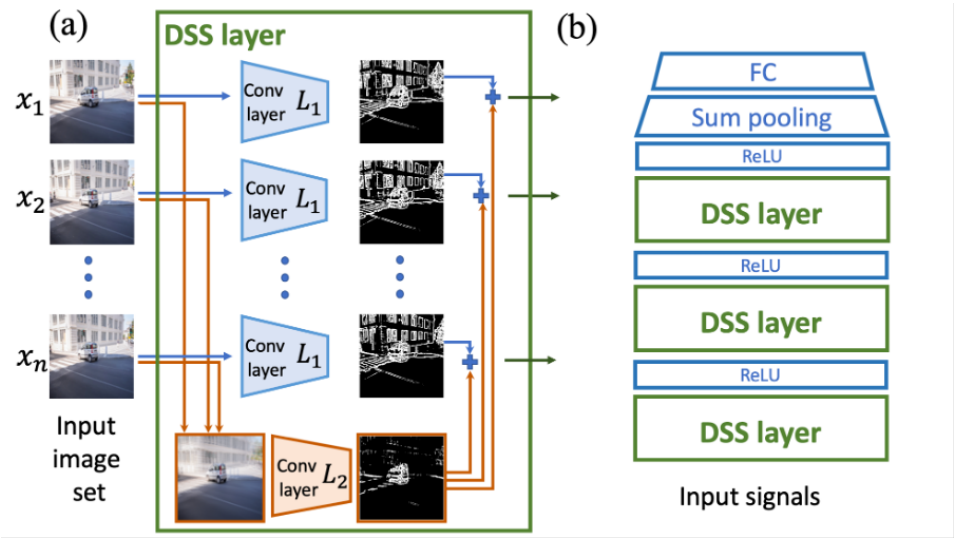

Figure 4: A DSS layer is composed of a Siamese Component (blue) that is applied to each component individually and an aggregation module (orange), that sums up all the images. The output of aggregation module is added to the Siamese output part. From Haggai Maron and Fetaya [2020].

Subgraph Symmetry and *Set Symmetry* are combined into a single Group, $H = (S_n \times S_m)$ as in Figure 3. It acts on $(\mathcal{A}, \mathcal{X})$ as following,

$$((\tau, \sigma) \cdot \mathcal{A})_{ijk} = \mathcal{A}_{\sigma^{-1}(i)\sigma^{-1}(j)\tau^{-1}(k)}, \quad ((\tau, \sigma) \cdot \mathcal{X})_{ilk} = \mathcal{X}_{\sigma^{-1}(i)l\tau^{-1}(k)}$$

To put it simply, σ permutes the nodes and τ permutes the subgraphs. Moreover, same permutation τ is applied over all subgraphs, thus nodes are ordered consistently over all subgraphs.

0.2.1.2 H-equivariant Layers

Equivariant architectures process subgraph bags in accordance with their natural symmetry (Product of *set* and *subgraph* symmetry). Building blocks of such an architecture are H-equivariant layers. [Haggai Maron and Fetaya, 2020] presented a method for learning sets of symmetric elements. The approach involves defining the symmetric group for the sets of symmetric elements, then defining the linear layer space that corresponds to it. It is composed of a Siamese component applied to each element individually, and an information sharing module that aggregates all elements. A simple DSS layer on a set of images is illustrated in Figure 4.
Motivated by this approach, Beatrice Bevilacqua [2021] adopted the same layer structure (Figure 5) composed of a Siamese and an information sharing component and parameterized it using an equivariant GNN layer like MPNN. The Layer $(L : \mathbb{R}^{n\times n\times m} \times \mathbb{R}^{n\times d\times m} \to \mathbb{R}^{n\times n\times m} \times \mathbb{R}^{n\times d'\times m})$ takes in a bag of subgraphs and outputs another bag of subgraphs.

$$(L(\mathcal{A}, \mathcal{X}))_i = L^1(A_i, X_i) + L^2\left(\sum_{j=1}^m A_j, \sum_{j=1}^m X_j\right)$$

Here, $(L(\mathcal{A}, \mathcal{X}))_i$ is the output of i-th subgraph when applied to the layer, $L^1, L^2 : \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times d'}$ represents two base graph encoders. By parametrizing L^1, L^2 with

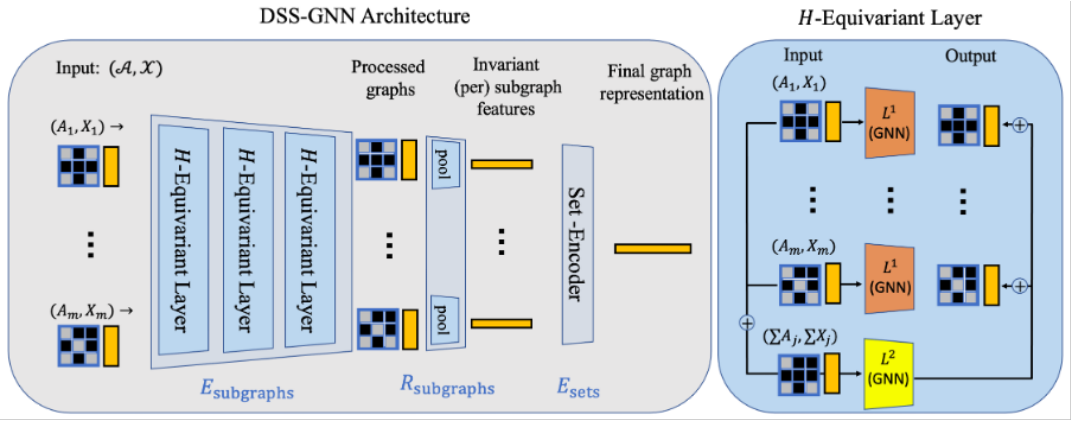

Figure 5: **Left**: DSS-GNN architecture, composed of a Feature Encoder, a Readout Layer and a Set Encoder. **Right**: An H-equivariant Layer with a Siamese component (Orange) and an information-sharing (yellow). From Beatrice Bevilacqua [2021].

MPNN the adjacency matrix of the processed subgraphs remain same, i.e., H-equivariany layer outputs the same adjacency matrix along with the processed nodes features.
While L^1 treats each graph independently, L^2 allows information to be shared across all subgraphs. DSS presented in [Haggai Maron and Fetaya, 2020], uses a sum aggregator, given by $\left(\sum_{j=1}^{m} A_j \text{ and } \sum_{j=1}^{m} X_j\right)$. Although other aggregator can be used such as mean, max or aggregator that without the applied subgraph, i.e., $\left(\sum_{j\neq i}^{m} A_j \text{ and } \sum_{j\neq i}^{m} X_j\right)$. Eventually, the S_n equivariance of base encoder and S_m equivariance of the aggregator gives H-equivariance (see Figure 5) of the layer.

0.2.1.3 DSS-GNN

The DSS-GNN (illustrated in Figure 5) is an architecture based on the DSS-GNN layer. It comprises of three parts and they are put together as follows,

$$F_{\text{DSS-GNN}} = E_{\text{sets}} \circ R_{\text{subgraphs}} \circ E_{\text{subgraphs}} \tag{0.1}$$

- *Equivariant feature encoder ($E_{subgraphs}$)* : $\mathbb{R}^{n\times n\times m} \times \mathbb{R}^{n\times d\times m} \rightarrow \mathbb{R}^{n\times n\times m} \times \mathbb{R}^{n\times d'\times m}$
  It consist of several H-equivariant layers and it learns useful features for the nodes in all of the applied sugraphs.
- *Subgraph Readout Layer $R_{subgraphs}$* : $\mathbb{R}^{n\times n\times m} \times \mathbb{R}^{n\times d'\times m} \rightarrow \mathbb{R}^{d'\times m}$
  This part creates a invariant feature vector by aggregating node and edge features independently for each input subgraph.
- *Universal Set Encoder E_{sets}* : $\mathbb{R}^{d'\times m} \rightarrow \mathbb{R}^{d''}$
  It outputs a final vector graph representation using a universal set encoder such as DeepSets [Manzil Zaheer and Smola, 2017] or PointNet [Charles R Qi and Guibas, 2017].

($\text{E}_{\text{subgraphs}} \circ \text{R}_{\text{subgraphs}}$) encodes a subgraph to a S_n-invariant representation and E_{sets} produces a single H-invariant representation corresponding to the original Graph, G from a set of S_n-invariant subgraphs.

0.2.1.4 DS-GNN

It is a variant of DSS-GNN, in which information sharing component of each *H-equivariant layer of* $E_{subgraphs}$ is set to off. We set $L^2 = 0$ in equation 0.1 for each of the layer in $\mathrm{E}_{\mathrm{subgraphs}}$. Thus, there is no information sharing component, each subgraph is encoded independently. In this arrangement, all the subgraphs are aligned, that is they have the same node permutation.

0.2.2 Subgraph Selection Policies

In this section we discuss the various subgraph selection policies that effect the expressive power of our architecture.
Some basic notation,

- G: Set of all graphs with n or less nodes.
- $\mathbb{P}(G)$: Set of all subsets, $S \subseteq G$.
- π: Its the Subgraph selection policy, it assigns a set of subgraphs for each Graph, G. The subgraph selection should be invariant of the node permutation.

We use this notation $\pi(G) \coloneqq S_G^\pi$. Here, the order of subgraph is arbitrary but the order of nodes within each subgraph is same.
The paper defined following subgraph selection policies,

- **Node-Deleted Policy**: Set of sub-graphs obtained by removing an edge from the original Graph.
- **Edge-Deleted Policy**: Removing a single edge.
- **EGO-Networks Policy**: Maps original graph to a set of ego-networks of some depth k, one for each node in the graph.
- **EGO+ Network Policy**: Variant of EGO where the root node has a identifying feature.
- **Augmented Policy** $\hat{\pi}$: Adding original graph to bag of sub-graphs obtained by using any selection policy π, to ensure at least the expressive power of the basic graph encoder. We donate any of the above subgraph selection policy as augmented policy by introducing a hat over the policy name, for example an Augmented EGO-Network policy is denoted by $\widehat{EGO}$.

0.2.2.1 Stochastic Sampling

Using entire bag of subgraphs might prove expensive for larger graphs. We use stochastic sampling as solution to this. It selects a small subset, $|\bar{S_G^\pi}| \subset S_G^\pi$ uniformly and randomly for every training epoch.

0.3 Computational Complexity

In this section we analyse the complexity of ESAN, table 1 summarizes the results for ESAN and some other known models.

Model	Time	Space
PPGN	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$
3-IGN	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$
3-GNN	$\mathcal{O}(n^4)$	$\mathcal{O}(n^3)$
GSN	$\mathcal{O}(n\Delta_{max})$	$\mathcal{O}(n + n\Delta_{\max})$
CWN	$\mathcal{O}\left(\sum_{p=1}^{2} n_p \left(B_p + 2\binom{B_p}{2}\right)\right)$	$\mathcal{O}\left(n + \sum_{p=1}^{2} n_p \left(1 + B_p + 2\binom{B_p}{2}\right)\right)$
ESN	$\mathcal{O}(\lvert S\rvert n\Delta_{\max})$	$\mathcal{O}(\lvert S\rvert(n + n\Delta_{\max}))$

Table 1: Complexity of Graph Networks. Δ_{max} denotes maximum degree over all nodes. From Beatrice Bevilacqua [2021]

0.3.1 Time Complexity

ESAN requires $\mathcal{O}(|S|n^2)$ time, here each subgraph is processed using MPNN which has a time complexity of $\mathcal{O}(n^2)$. For node-deleted and ego-net subgraph selection policies, $|S| = \mathcal{O}(n)$, that results in a total time complexity of $\mathcal{O}(n^3)$, which is equal to PPGN [Haggai Maron and Lipman, 2019a] and 3-IGN [Haggai Maron and Lipman, 2019b]. Furthermore, for edge-deleted graph selection policies, $|S| = \mathcal{O}(n^2)$, that gives a total time complexity of $\mathcal{O}(n^4)$, which is comparable to 3-GNN [Christopher Morris and Grohe, 2019].
For sparse Graphs the time complexity of ESAN improves to $\mathcal{O}(|S|n\Delta_{max})$, here Δ_{max} is maximum node degree. Relative to PPGN and k-IGN this is a major benefit for ESAN, as the former two do not improve for sparse graphs. For sparse graph $\Delta_{max} = 1$.
Table 1 also present two sparse architechtures, GSN [Giorgos Bouritsas and Bronstein, 2022] and CWN [Cristian Bodnar and Bronstein, 2021a]. GSN gives the same complexity as of standard GNNs for small number of subgraphs. ESAN can match complexity of GSN, only in the case when stochastic sampling used. However, the prepossessing phase for GSN is expensive. For CWN, time complexity is given by a general lifting procedure that generate two dimensional cellular complex. n_p denotes the number of cells at dimension p and B_p is the maximum boundary size. The number of nodes is given by $n_0 = n$ and number of edges by $n_1 = \mathcal{O}(n\Delta_{max})$. The complexity of CWN is given by, $\mathcal{O}(n\Delta_{max} + n_2)$. In a general graph distribution, the number of 2-cells may grow exponentially, however n_2 is contained in case of molecules.

0.3.2 Space Complexity

Time complexity of ESAN is given by $\mathcal{O}(|S|(n + n\Delta_{max}))$, which includes computing n node features, by keeping track of subgraph connectivity. Therefore, the complexity is $\mathcal{O}\left(n^2\Delta_{max} + (n\Delta_{max})^2\right)$ for edge-deleted policy and $\mathcal{O}(n^2 + n^2\Delta_{max})$ for node-deleted and

ego-net subgraph selection policies. However for DS-GNN architechture the space complexity improves to $\mathcal{O}(n + n\Delta_{max})$. Subgraph selection policies like low depth ego-net policies has lesser number of edges, which means smaller subgraph sizes thus they have increased efficiency.

0.3.3 Preprocessing

Applying subgraph selection policy to generate subgraphs from the Original Graph G, is a one time work and done before the training. Node Deleted and Edge deleted subgraph selection policies takes $\mathcal{O}(nm)$ and $\mathcal{O}(m^2)$ respectively, here n is the number of nodes and m is number of edges. Moreover, for sparse graphs, m can have a upper bound of $\mathcal{O}(n\Delta_{max})$, where Δ_{max} is usually small for sparse graphs. Ego-Net policy takes $\mathcal{O}(n(n+m))$, which includes $\mathcal{O}(n+m)$ for breadth first search, to compute the k-neighbourhood of a node. Furthermore, the runtime complexity might get significantly reduced for sparse graphs.

0.4 Theoretical Analysis

In this section we examine the expressive power of our architecture. We introduce a WL analogues for ESAN and analyse how different design choices affects the expressiveness of ESAN.

0.4.1 WL Analogue for ESAN

The paper introduced two refinement procedures, **DSS-WL** and its variant **DS-WL** that encodes bags of sub-graphs, inspired by the WL isomorphism test [Weisfeiler and Leman, 1968]. The idea is that a pair of graphs are more effectively separated by encoding subset of their contained subgraphs rather than encoding the nodes of the graphs as done in WL isomorphism test. Below is a description of the procedure.

0.4.1.1 Initiation

The DSS-WL takes in as input, the pair of graphs G^1, G^2 and a subgraph selection policy, π. First, the algorithm applies the subgraph selection policy and generates subgraphs from the original graphs. Then each node in each subgraph is assigned a color based on the initial coloring (if provided) or randomly.

0.4.1.2 Refinement

The color of the node, v in a subgraph S is refined according to the formula:

$$c_{v,S}^{t+1} \leftarrow \text{HASH}\Big(c_{v,S}^{t}, N_{v,S}^{t}, C_{v}^{t}, M_{v}^{t}\Big)$$

Here, $N_{v,S}^{t}$ is multiset of colors in v's neighbourhood in a subgraph S, C_{v}^{t} is multiset of v's colors across all subgraphs and M_{v}^{t} is a multiset of aggregated colors of v's neighbourhood across all subgraphs according to the original graph connectivity.

0.4.1.3 Termination

The subgraph S in each step of the algorithm is associated with a color c_s, which is a multiset of color associated with the nodes in the subgraph. So, each graph is represented by multisets of subgraphs colors. As soon as the multisets start to diverge the algorithm terminates and the graphs are concluded to be non-isomorphic. However, if the refinement step assigns same representation two both graphs, the test is inconclusive.
DS-WL is a variant of DSS-WL where the refinement rule does not consider input C_v^t and M_v^t. That is the inputs representing information sharing across subgraphs is disabled, which results in a vanilla WL that runs on each subgraph independently.

0.4.2 WL Analogue and Expressive Power

The paper presents couple of results (for proof refer to the main paper) regarding the expressive power of the DSS-WL.

Theroem 1: DS(S)-WL is strictly more powerful than 1-WL.
The idea behind this is, that there exists subgraph selection policies such that,

1. Any pair of graphs that is distinguishable by WL is also distinguished by DS(S)-WL.
2. There exists pairs of graphs which are distinguishable by DS(S)-WL but cannot be distinguished by WL. Circulant Skip Link graphs (see definition below) is an example of such graph pairs.

Circulant Skip Link graphs: As defined in Ryan Murphy and Ribeiro [2019], Circulant Skip Link, CSL(M, R) represents an undirected 4-regular graph(degree of all nodes is 4) with vertices $\{0, 1, ..., M1\}$ whose edges form a cycle and have skip links. Here, R and M are co-prime such that $R < M - 1$, for cycle $\{j, j + 1\}, j \in \{0, 1, M - 2\}$ and $\{M - 1, 0\}$ and for skip links, we recursively define a sequence $s_1 = 0$, $s_{i+1} = (s_i + R) \mod M$ and $\{s_i, s_{i+1}\}$ for $i \in \mathbb{N}$.

The paper presents the below result, which proves that DS(S)-WL is able to distinguish such pairs of CSL graphs (See Figure 1 for example of ESAN with CSL using Edge Deleted subgraph selection policy).

Lemma 1 CSL$(n, 2)$ *can be distinguished from any CSL*(n, k) *with* $k \in [3, n/2 - 1]$ *by DS-WL and DSS-WL with either the* ND, EGO, or EGO+ policy.

Furthermore, the paper present this result which shows that the expressive power of our variants can be attained by ESAN.

Theorem 2 DSS-GNN at least as powerful as DS(S)-WL; DS-GNN at most as powerful as DS-WL.
Let $\mathcal{F}$ *be any family of bounded-sized graphs endowed with node labels from a finite set. There exist selection policies such that, for any two graphs* G^1, G^2 *in* $\mathcal{F}$*, distinguished by DS(S)-WL, there is a DS(S)-GNN model in the form of Equation (0.1) assigning* G^1, G^2 *distinct representations. Also, DS-GNN with MPNN base graph encoder is at most as powerful as DS-WL.*

The above theorem is valid until each egde that exists in original graph appears at least

once in any of the subgraphs. Following this assumption, DS(S)-GNN has a power at least equal to DS(S)-WL. In conjunction with Theorem 2, we can therefore conclude that there are policies for subgraph selection that make ESAN more powerful than WL.
Furthermore, the paper presents the following corollary based on results of [Keyulu Xu and Jegelka, 2019] and [Christopher Morris and Grohe, 2019] (MPNN being at most as stronger as WL test) and Theorem 1, Theorem 2.

Corollary 1 (DS(S)-GNN is strictly more powerful than MPNNs). *There exists subgraph selection policies such that DSS-GNN and DS-GNN architectures are strictly more powerful than standard MPNN models in distinguishing non-isomorphic graphs.*

0.4.3 Expressiveness of Design Choices

This section analyses various significant design choices for ESAN.

0.4.3.1 DSS vs DS matters

DSS-GNN performs as well or better than DS-GNN, depending on the subgraph selection policy.

0.4.3.2 Base graph encoder

A DSS-GNN with either of depth-l $\widehat{EGO}$ or $\widehat{EGO}+$ as subgraph selection policy and a 3-WL base graph encoder is strictly powerful than 3-WL and a DSS-GNN with 3-WL base graph encoder more powerful than the one with 1-WL base graph encoder. That is, ESAN gains expressiveness by increasing the expressive power of the graph encoder.

0.4.3.3 Subgraph selection policy

The significance of Subgraph selection policy in respect to expressive power can be explained through Strongly Regular (SR) Graphs.SR graphs are defined by 4 parameters (n, k, λ, μ). 1-WL and 3-WL cannot distinguish SR graphs with same parameters. A DS-GNN with ND, EGO+ or ED subgraph policy can easily distinguish SR graphs with different parameters. However, it cannot distinguish SR graphs with same parameters just like 1-WL and 3-WL. Moreover, a DSS-GNN with ED policy can distinguish some SR graphs with same parameters. Thus for SR graphs, DS-GNN with ND, EGO are as strong as 3-WL test and DS-GNN with ED is more powerful than 3-WL.

0.5 Experiments

In this section we test ESAN on several data-sets and compare the results with base graph encoders. We run tests with different combinations of sub-graph selection policies, the two variants DSS-GNN and DS-GNN, full approach or stochastic variant, to compare the expressive power of various combinations.

- **EXP, CEXP, CSL**: On these datasets, results shows that 1-WL GNN, GIN and Graphconv performs no better than a random guess. However, ESAN gives perfect accuracy with either of 1-WL, GIN or Graphconv base encoders. Moreover, even stochastic variants gives similar accuracy.

- **TUDatasets**: On TUDatasets SoTA method outperforms all other methods. However, reports excellent results right behing SoTA. ESAN outperforms other methods such as ID-GNN and RNI. The results shows that, particularly successful configurations are:
  1. DSS-GNN (EGO+) + GraphConv/GIN
  2. DS-GNN (ND) + GIN

  Moreover, analysis of results also tells stochastic variant is a strong alternative for base encoder based methods.
- **OGB**: On OGB dataset all ESAN variants outperforms their base encoder GIN. Furthermore, DSS-GNN performs better that DS-GNN with GCN base encoder.
- **ZINC12K**: Results show that all variants of ESAN outperforms their base GIN encoder. However, in general ESAN performs competitively with other method irrespective of selected subgraph policies. It outperforms PNA model. However other methods are only outperformed when ESAN is equipped with EGO(+) policy.

0.6 Conclusion

The paper presented a novel framework, ESAN, with increased expressiveness over its former counterpart like MPNN and performs significantly good with several graph classification benchmarks. Though, ESAN is more computationally expensive then MPNN, and the paper also describes a stochastic version to alleviate this problem.

Bibliography

[1] D. L. B. S. C. C. G. B. M. M. B. H. M. Beatrice Bevilacqua, Fabrizio Frasca. Equivariant subgraph aggregation networks, 2021. URL `https://arxiv.org/abs/2110.02910`. Number: arXiv:2110.02910. 1, 2, 4, 5, 6, 8

[2] K. M. Charles R Qi, Hao Su and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *In Proceedings of the IEEE conference on computer vision and pattern recognition*, page 652–660, 2017. 6

[3] M. F. W. L. H. J. E. L. G. R. Christopher Morris, Martin Ritzert and M. Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, page 4602–4609, 2019. 2, 3, 8, 11

[4] P. M. Christopher Morris, Gaurav Rattan. Weisfeiler and leman go sparse: Towards scalable higher-order graph embeddings. In *H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), Advances in Neural Information Processing Systems*, volume 33, page 21824–21840. Curran Associates, Inc, 2020. 3

[5] N. O. Y. G. W. P. L. G. M. Cristian Bodnar, Fabrizio Frasca and M. Bronstein. Weisfeiler and lehman go cellular: Cw networks. 34, 2021a. 3, 8

[6] S. Z. Giorgos Bouritsas, Fabrizio Frasca and M. M. Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting, 2022. URL `https://arxiv.org/abs/2006.09252`. Number: arXiv:2006.09252. 3, 8

[7] G. C. Haggai Maron, Or Litany and E. Fetaya. On learning sets of symmetric elements. In *International Conference on Machine Learning*, page 6734–6744. PLMR, 2020. 5, 6

[8] N. S. Haggai Maron, Heli Ben-Hamu and Y. Lipman. Provably powerful graph networks. In *NeurIPS*, 2019a. 8

[9] N. S. Haggai Maron, Heli Ben-Hamu and Y. Lipman. Invariant and equivariant graph networks. In *International Conference on Learning Representations (ICLR)*, 2019b. 8

[10] P. F. R. O. V. G. E. D. Justin Gilmer, Samuel S. Schoenholz. Neural message passing for quantum chemistry. In *2017 PLMR 34th International Conference on Machine Learning*, pages 1263–1272. PLMR, 2017. 2

[11] J. L. Keyulu Xu, Weihua Hu and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. 3, 11

[12] S. R. B. P. R. R. S. Manzil Zaheer, Satwik Kottur and A. J. Smola. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017. 6

[13] T. C. P. V. Michael M. Bronstein, Joan Bruna. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, 2021. URL `https://arxiv.org/abs/2104.13478`. Number: arXiv:2104.13478. 2, 3, 4

[14] V. R. Ryan Murphy, Balasubramaniam Srinivasan and B. Ribeiro. Relational pooling for graph representations. In *International Conference on Machine Learning*, page 4663–4673. PLMR, 2019. 10

[15] B. Weisfeiler and A. Leman. The reduction of a graph to canonical form and the algebra which appears therein. 2(9):12–16, 1968. 2, 9