Equivariant Subgraph Aggregation Networks[1]

Beatrice Bevilacqua⋆ Fabrizio Frasca⋆ Derek Lim⋆

Joint work with
Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan,
Michael M. Bronstein, Haggai Maron

December 14, 2021

[1] Project sparked from the 2021 LOGML Summer School

Outline

Outline

Main idea

We define a new family of **provably powerful** **graph neural networks** that treat a **graph as a set of subgraphs**.

Background: Graph Neural Networks (GNNs)

A **GNN** is a parameterized function on graphs.

Definition (Message Passing Neural Network)

A message passing neural network (MPNN) updates node representations $h_v^{(k)} \in \mathbb{R}^d$ by

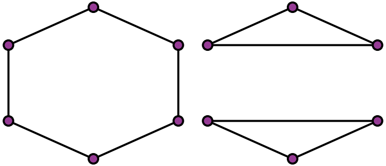
$$\begin{aligned} \mathrm{msg}_v^{(k)} &= \textsc{Aggregate}(\{h_u^{(k)} : u \text{ neighbor of } v\}), \\ h_v^{(k+1)} &= \textsc{Combine}\left(h_v^{(k)}, \mathrm{msg}_v^{(k)}\right). \end{aligned}$$

Note: In their original formulation, they are permutation equivariant!

MPNNs have limited expressive power

No matter how the AGGREGATE and COMBINE functions are parameterized, MPNNs are bounded in their expressive power (they are not universal!).

$\Longrightarrow$ They cannot distinguish between non-isomorphic graphs beyond 1-WL [Xu et al., 2019]

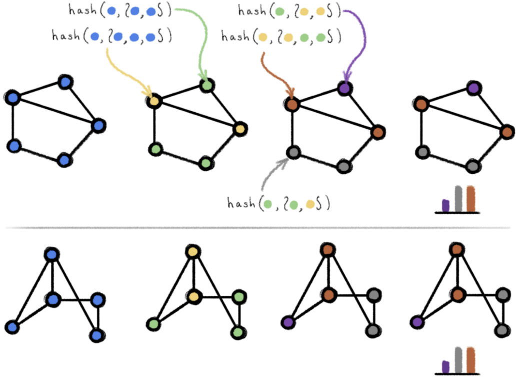


The 1-WL isomorphism test

A polynomially fast heuristic to disambiguate non-isomorphic graphs:

- Iterate node color refinements

$$c_v \leftarrow \mathrm{HASH}(c_v, \{\!\!\{c_w\}\!\!\}_{w \in \mathcal{N}(v)})$$



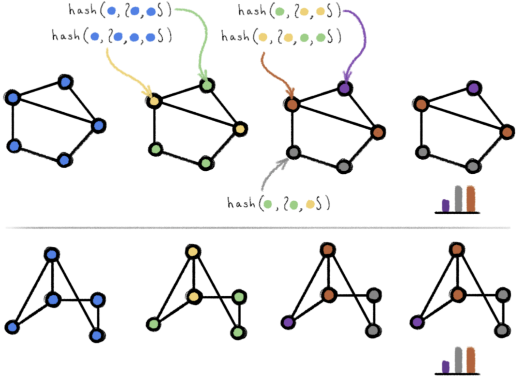
The 1-WL isomorphism test

A polynomially fast heuristic to disambiguate non-isomorphic graphs:

- Iterate node color refinements

$$c_v \leftarrow \mathrm{HASH}(c_v, \{\!\!\{ c_w \}\!\!\}_{w \in \mathcal{N}(v)})$$

- Represent graphs as color histograms



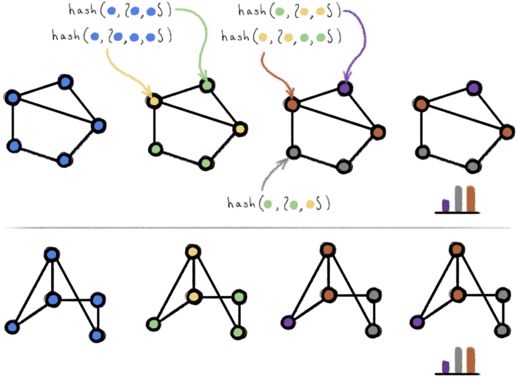
The 1-WL isomorphism test

A polynomially fast heuristic to disambiguate non-isomorphic graphs:

- Iterate node color refinements

$$c_v \leftarrow \mathrm{HASH}(c_v, \{\!\{c_w\}\!\}_{w \in \mathcal{N}(v)})$$

- Represent graphs as color histograms
- *distinct* histograms: *non-isomorphic* graphs



The 1-WL isomorphism test

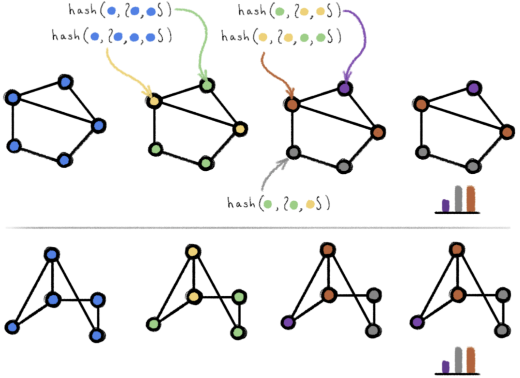
A polynomially fast heuristic to disambiguate non-isomorphic graphs:

- Iterate node color refinements

$$c_v \leftarrow \mathrm{HASH}(c_v, \{\!\{c_w\}\!\}_{w \in \mathcal{N}(v)})$$

- Represent graphs as color histograms
- *distinct* histograms: *non-isomorphic* graphs

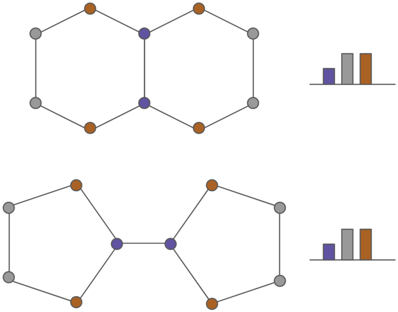
Problem: only a *sufficient* condition.



Beyond 1-WL

Why is expressive power important?

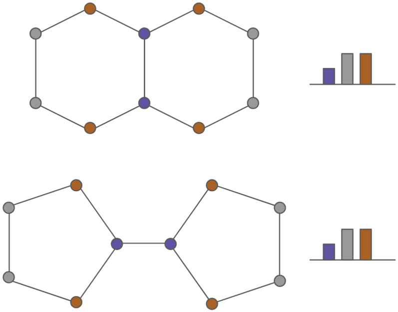
1. 1-WL cannot distinguish some molecular graphs (e.g. *decalin* vs. *bicyclopentyl*)



Beyond 1-WL

Why is expressive power important?

1. 1-WL cannot distinguish some molecular graphs (e.g. *decalin* vs. *bicyclopentyl*)
2. 1-WL cannot count non-trivial patterns (e.g. *triangles* and *cycles*) [Chen et al., 2020]

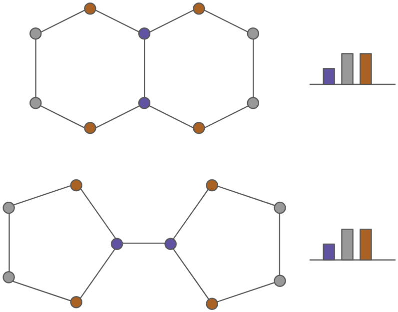


Beyond 1-WL

Why is expressive power important?

1. 1-WL cannot distinguish some molecular graphs (e.g. *decalin* vs. *bicyclopentyl*)
2. 1-WL cannot count non-trivial patterns (e.g. *triangles* and *cycles*) [Chen et al., 2020]

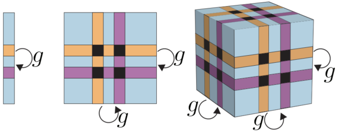
These limitations are directly inherited by MPNNs.



GNNs beyond 1-WL

Currently a prolific research area, with diverse emerging approaches:

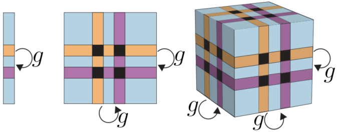
- k-WL inspired architectures: k-GNNs
  [Morris et al., 2019, Maron et al., 2019]
  - non-local
  - computationally intractable for k $\geq$ 3



GNNs beyond 1-WL

Currently a prolific research area, with diverse emerging approaches:

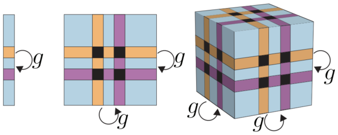
- k-WL inspired architectures: k-GNNs [Morris et al., 2019, Maron et al., 2019]
    - non-local
    - computationally intractable for k $\geq$ 3
- Random node features / identifiers [Abboud et al., 2020, Sato et al., 2021]
    - no permutation equivariance
    - difficulties in generalisation

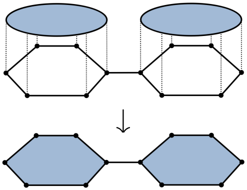


GNNs beyond 1-WL

Currently a prolific research area, with diverse emerging approaches:

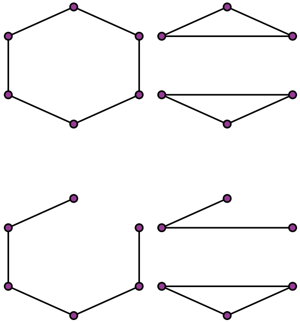
- k-WL inspired architectures: k-GNNs [Morris et al., 2019, Maron et al., 2019]
  - non-local
  - computationally intractable for k $\geq$ 3
- Random node features / identifiers [Abboud et al., 2020, Sato et al., 2021]
  - no permutation equivariance
  - difficulties in generalisation
- Substructure–aware message passing [Bouritsas et al., 2020, Bodnar et al., 2021]
  - some form of domain-knowledge is required





Outline

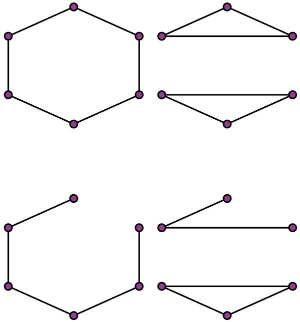
Graphs as sets of subgraphs



Desiderata:

- Provable expressive power (> 1-WL)
- (Almost) no engineering required
- Computationally tractable
- Equivariance to permutations

Graphs as sets of subgraphs



Desiderata:

- Provable expressive power ($> 1\text{-WL}$)
- (Almost) no engineering required
- Computationally tractable
- Equivariance to permutations

Intuition: Even if two graphs are indistinguishable to MPNNs, they may contain subgraphs which are, instead, (easily) separated.

Equivariant Subgraph Aggregation Networks (ESAN)

Recipe?

1. Map a graph into a set of subgraphs (bag) via a *selection policy*: $G \mapsto \{G_1, \ldots, G_m\}$
2. Process the bag in a principled way: respecting the *inherent symmetries* of such object

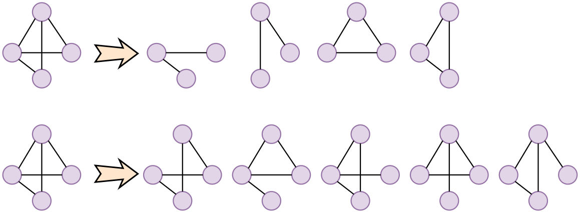
Equivariant Subgraph Aggregation Networks (ESAN)

Recipe?

1. Map a graph into a set of subgraphs (bag) via a *selection policy*: $G \mapsto \{G_1, \ldots, G_m\}$
2. Process the bag in a principled way: respecting the *inherent symmetries* of such object

Policies:

- Node Deletion (ND)
- Edge Deletion (ED)
- Ego-Nets (EGO)
- ...



Background: Equivariance

Let $f : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ and H be a symmetry group acting on $\mathbb{R}^{d_1}$ and $\mathbb{R}^{d_2}$.

Definition

f is **invariant** if $f(h \cdot x) = f(x)$ for all $h \in H$.
f is **equivariant** if $f(h \cdot x) = h \cdot f(x)$ for all $h \in H$.

Background: Equivariance

Let $f : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ and H be a symmetry group acting on $\mathbb{R}^{d_1}$ and $\mathbb{R}^{d_2}$.

Definition

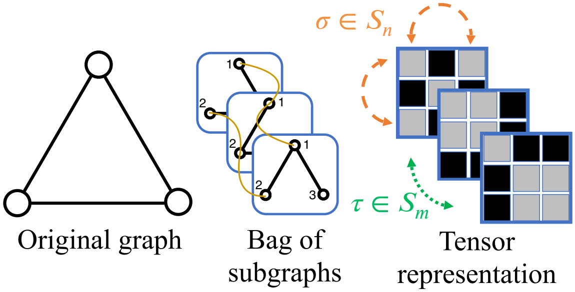
f is **invariant** if $f(h \cdot x) = f(x)$ for all $h \in H$.
f is **equivariant** if $f(h \cdot x) = h \cdot f(x)$ for all $h \in H$.

Example (Symmetries in Graphs)

Let G a graph with adjacency matrix $A \in \mathbb{R}^{n \times n}$ and node features $X \in \mathbb{R}^{n \times d}$.
S_n (permutations on n elements) acts on G *by renaming node i to $\sigma(i)$*:

$$(\sigma \cdot A)_{ij} = A_{\sigma^{-1}(i),\sigma^{-1}(j)} \qquad (\sigma \cdot X)_{ij} = X_{\sigma^{-1}(i),j} \tag{1}$$

Required equivariance



We want equivariance to $S_m \times S_n$.

$\tau \in S_m$ permutes graphs in the set.

$\sigma \in S_n$ permutes nodes in each subgraph.

Main idea: node alignment. $\sigma \in S_n$ same across all subgraphs. Node i in each subgraph corresponds to node i in original graph.

Our models

Definition (DS-GNN)

DS-GNN is Siamese:

$$\mathrm{DeepSets}\left(\{\mathrm{MPNN}(A_1, X_1), \dots, \mathrm{MPNN}(A_m, X_m)\}\right) \tag{2}$$

Definition (DSS-GNN)

DSS-GNN defines layers L where:

$$L(\mathbf{A}, \mathbf{X})_i = \mathrm{MPNN}_1(A_i, X_i) + \mathrm{MPNN}_2\left(\sum_{j=1}^{m} A_j, \sum_{j=1}^{m} X_j\right) \tag{3}$$

Main idea: DSS preserves node alignment ($S_m \times S_n$), DS-GNN does not ($S_m \wr S_n$).

Outline

"DSS-WL"

- Get a bag of subgraphs

$$\mathcal{G} \leftarrow \pi(G)$$

"DSS-WL"

- Get a bag of subgraphs

$$\mathcal{G} \leftarrow \pi(G)$$

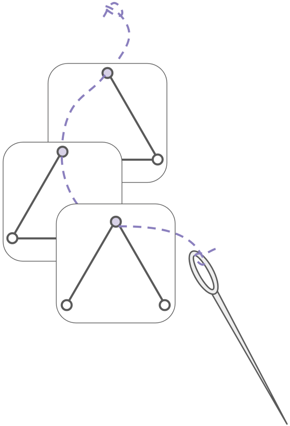
- Augmented node color refinements

$$c_v^S \leftarrow \mathrm{HASH}(c_v^S, N_v^S, C_v, M_v)$$

$$N_v^S = \{\!\{c_w^S\}\!\}_{w \in \mathcal{N}^S(v)} \qquad \triangleleft \text{ adjacent colors on } S$$

$$C_v = \{\!\{c_v^R\}\!\}_{R \in \mathcal{G}} \qquad \triangleleft \textit{ needle}\text{-color}$$

$$M_v = \{\!\{C_w\}\!\}_{w \in \mathcal{N}(v)} \qquad \triangleleft \text{ adjacent } \textit{needle}\text{-colors on } G$$



"DSS-WL"

- Get a bag of subgraphs

$$\mathcal{G} \leftarrow \pi(G)$$

- Augmented node color refinements

$$c_v^S \leftarrow \mathrm{HASH}(c_v^S, N_v^S, C_v, M_v)$$

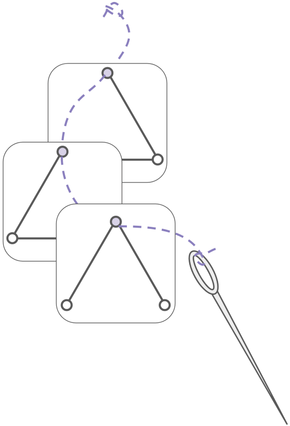
$$N_v^S = \{\!\{c_w^S\}\!\}_{w \in \mathcal{N}^S(v)} \quad \triangleleft \text{ adjacent colors on } S$$

$$C_v = \{\!\{c_v^R\}\!\}_{R \in \mathcal{G}} \quad \triangleleft \textit{ needle}\text{-color}$$

$$M_v = \{\!\{C_w\}\!\}_{w \in \mathcal{N}(v)} \quad \triangleleft \text{ adjacent } \textit{needle}\text{-colors on } G$$

- Subgraph color: HASH the multiset of node colors

$$c_S \leftarrow \mathrm{HASH}(\{\!\{c_v^S\}\!\}_{v \in S})$$



"DSS-WL"

- Get a bag of subgraphs

$$\mathcal{G} \leftarrow \pi(G)$$

- Augmented node color refinements

$$c_v^S \leftarrow \mathrm{HASH}(c_v^S, N_v^S, C_v, M_v)$$

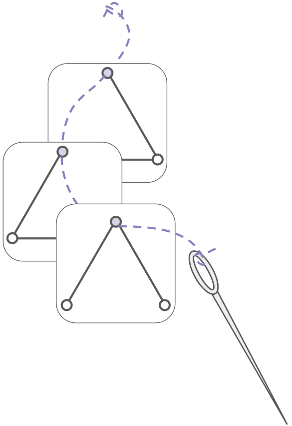
$$N_v^S = \{\!\!\{c_w^S\}\!\!\}_{w \in \mathcal{N}^S(v)} \qquad \triangleleft \text{ adjacent colors on } S$$

$$C_v = \{\!\!\{c_v^R\}\!\!\}_{R \in \mathcal{G}} \qquad \triangleleft \textit{ needle}\text{-color}$$

$$M_v = \{\!\!\{C_w\}\!\!\}_{w \in \mathcal{N}(v)} \qquad \triangleleft \text{ adjacent } \textit{needle}\text{-colors on } G$$

- Subgraph color: HASH the multiset of node colors

$$c_S \leftarrow \mathrm{HASH}(\{\!\!\{c_v^S\}\!\!\}_{v \in S})$$



Eventually, represent the original graph as the *histogram of subgraph colors*.

WL-variants and expressive power

Variant	Policy	Hash inps.	Neural
DSS-WL	(any)	c_v^S, N_v^S, C_v, M_v	DSS-GNN
DS-WL	(any)	c_v^S, N_v^S	DS-GNN
1-WL	$G \mapsto \{G\}$	c_v^S, N_v^S	GNN

- DS-WL: disable cross-bag info sharing
  - independent 1-WL on each subgraph
- 1-WL: employ trivial policy $G \mapsto \{G\}$

[2] on families of bounded-sized graphs; DSS: edge set-preserving policies

WL-variants and expressive power

Variant	Policy	Hash inps.	Neural
DSS-WL	(any)	c_v^S, N_v^S, C_v, M_v	DSS-GNN
DS-WL	(any)	c_v^S, N_v^S	DS-GNN
1-WL	$G \mapsto \{G\}$	c_v^S, N_v^S	GNN

- DS-WL: disable cross-bag info sharing
  - independent 1-WL on each subgraph
- 1-WL: employ trivial policy $G \mapsto \{G\}$

Theorem: There exist policies such that DS(S)-WL > 1-WL.

Theorem: DS(S)-GNN can implement DS(S)-WL[2]. Also: DS-GNN(MPNN) $\leq$ DS-WL.

Corollary: There exist policies such that DS(S)-GNN > MPNN.

[2] on families of bounded-sized graphs; DSS: edge set-preserving policies

Expressiveness experimental validation

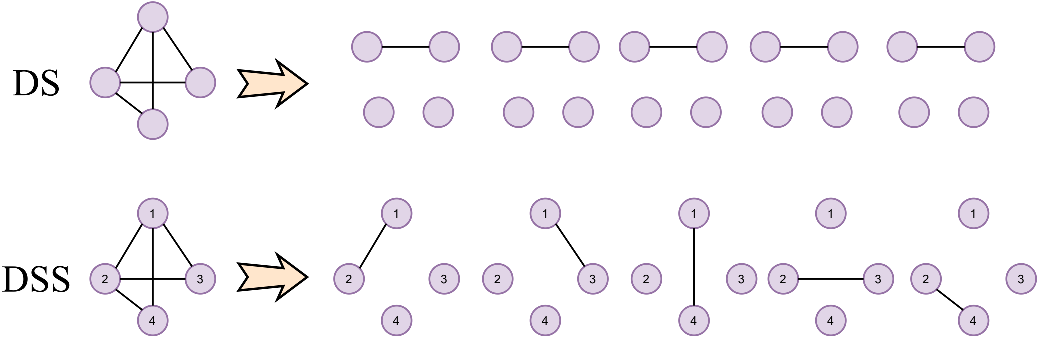
Experiments: DSS-GNN and DS-GNN are perfect on RNI and CSL datasets (which require $>$ 1-WL power).

	EXP	CEXP
GIN [Xu et al., 2019]	51.1±2.1	70.2±4.1
DS-GNN (GIN) (ED/ND/EGO/EGO+)	100±0.0	100±0.0
DSS-GNN (GIN) (ED/ND/EGO/EGO+)	100±0.0	100±0.0
GraphConv [Morris et al., 2019]	50.3±2.6	72.9±3.6
DS-GNN (GraphConv) (ED/ND/EGO/EGO+)	100±0.0	100±0.0
DSS-GNN (GraphConv) (ED/ND/EGO/EGO+)	100±0.0	100±0.0

Outline

DSS-GNN > DS-GNN

Proposition: DSS-GNN > DS-GNN for certain policies.



Subgraph policy: map G to set of single edges.

DSS-GNN can reconstruct graph $A = \sum_j A_j$ because of node alignment.
DS-GNN only sees number of edges (no alignment).

DSS-GNN > DS-GNN

Experiments: DSS-GNN tends to outperform DS-GNN.

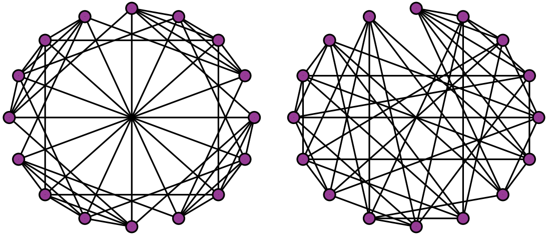
DSS-GNN is better than base encoder **91%** of the time.
DS-GNN is better than base encoder 75% of the time.

Subgraph policy matters

Proposition: On the family of *strongly regular graphs*:

$$\text{Edge-deletion} > \text{Node-deletion} = \text{Depth-}n \text{ EGO}(+) = \text{3-WL}.$$

Strongly regular graphs: highly-symmetric graphs that are hard cases for graph isomorphism.



Subgraph policy choice in the experiments

No strong correlation between the performance of a policy and the application domain.
The EGO(+) policies were found generally more consistent in their results across datasets.

Subgraph policy choice in the experiments

No strong correlation between the performance of a policy and the application domain.
The EGO(+) policies were found generally more consistent in their results across datasets.

	ZINC ↓
DS-GNN (GIN) (ED)	0.172±0.008
DS-GNN (GIN) (ND)	0.171±0.010
DS-GNN (GIN) (EGO)	0.126±0.006
DS-GNN (GIN) (EGO+)	0.116±0.009
DSS-GNN (GIN) (ED)	0.172±0.005
DSS-GNN (GIN) (ND)	0.166±0.004
DSS-GNN (GIN) (EGO)	0.107±0.005
DSS-GNN (GIN) (EGO+)	0.102±0.003

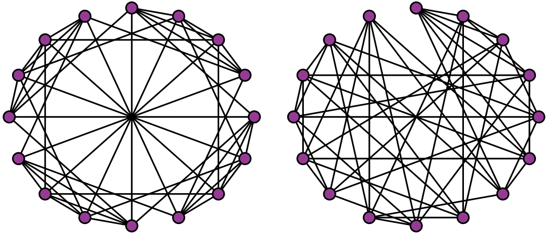
Subgraph policy choice in the experiments

No strong correlation between the performance of a policy and the application domain.
The EGO(+) policies were found generally more consistent in their results across datasets.

	ZINC ↓
DS-GNN (GIN) (ED)	0.172±0.008
DS-GNN (GIN) (ND)	0.171±0.010
DS-GNN (GIN) (EGO)	0.126±0.006
DS-GNN (GIN) (EGO+)	0.116±0.009
DSS-GNN (GIN) (ED)	0.172±0.005
DSS-GNN (GIN) (ND)	0.166±0.004
DSS-GNN (GIN) (EGO)	0.107±0.005
DSS-GNN (GIN) (EGO+)	0.102±0.003

Base encoder matters

Proposition: Using a 3-WL base encoder is stronger than using a 1-WL (MPNN) base encoder.



Outline

ESAN outperforms the base encoder

	OGBG-MOLHIV	OGBG-MOLTOX21
BASE	75.58±1.40	74.91±0.51
ESAN	78.00±1.42	77.95±0.40

ESAN[3] generally improves the base encoder accuracy (e.g. 2.4% improvement on OGBG-MOLHIV and 3% improvement on OGBG-MOLTOX21)

[3] In this section, we report the performance of our best ESAN model

ESAN is competitive with SoTA

	MUTAG	PTC	PROTEINS	NCI1	NCI109	IMDB-B
GIN [Xu et al., 2019]	89.4±5.6	64.6±7.0	76.2±2.8	82.7±1.7	82.2±1.6	75.1±5.1
GraphConv [Morris et al., 2019]	90.5±4.6	64.9±10.4	73.9±6.1	82.4±2.7	81.7±1.0	76.1±3.9
PPGNs [Maron et al., 2019]	90.6±8.7	66.2±6.6	77.2±4.7	83.2±1.1	82.2±1.4	73.0±5.8
GSN [Bouritsas et al., 2020]	92.2±7.5	68.2±7.2	76.6±5.0	83.5±2.0	N/A	**77.8**±3.3
CIN [Bodnar et al., 2021]	**92.7**±6.1	68.2±5.6	77.0±4.3	83.6±1.4	**84.0**±1.6	75.6±3.7
ESAN	92.0±5.0	**69.2**±6.5	**77.3**±3.8	**83.8**±2.4	83.1±0.8	77.1±3.0

ESAN achieves excellent results with respect to SoTA methods

ESAN on ZINC

Best performing model amongst all provably expressive, **domain agnostic** GNNs

	ZINC $\downarrow$
PNA [Corso et al., 2020]	0.188±0.004
DGN [Beaini et al., 2021]	0.168±0.003
SMP [Vignac et al., 2020]	0.138±?
GIN [Xu et al., 2019]	0.252±0.017
HIMP [Fey et al., 2020]	0.151±0.006
GSN [Bouritsas et al., 2020]	0.108±0.018
CIN-SMALL [Bodnar et al., 2021]	0.094±0.004
ESAN	0.102±0.003

ESAN on ZINC

Best performing model amongst all provably expressive, **domain agnostic** GNNs

Competitive with (provably powerful) GNNs employing **domain specific** structural information, often outperforming them.

	ZINC ↓
PNA [Corso et al., 2020]	0.188±0.004
DGN [Beaini et al., 2021]	0.168±0.003
SMP [Vignac et al., 2020]	0.138±?
GIN [Xu et al., 2019]	0.252±0.017
HIMP [Fey et al., 2020]	0.151±0.006
GSN [Bouritsas et al., 2020]	0.108±0.018
CIN-SMALL [Bodnar et al., 2021]	0.094±0.004
ESAN	0.102±0.003

Outline

Complexity

Table: Complexity of graph networks that are more expressive than 1-WL. $\Delta_{\max}$ denotes the maximum degree over all nodes.

	PPGN	3-IGN	3-GNN	Ours
Time	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^4)$	$\mathcal{O}\left(\lvert S\rvert n\Delta_{\max}\right)$
Space	$\mathcal{O}(n^2)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$	$\mathcal{O}\left(\lvert S\rvert(n+n\Delta_{\max})\right)$

Table: Bag sizes and bag construction complexity.

	$\lvert S\rvert$	Time complexity
ND	$\Theta(n)$	$\mathcal{O}(n^2\Delta_{\max})$
ED	$\mathcal{O}(n\Delta_{\max})$	$\mathcal{O}((n\Delta_{\max})^2)$
EGO(+)	$\Theta(n)$	$\mathcal{O}(n(n+n\Delta_{\max}))$

Sampling approach

To reduce computational cost, sample a **subset of the subgraphs** for each graph.
At inference: majority voting on the outputs of k independent samplings ($k = 5$).

Sampling approach

To reduce computational cost, sample a **subset of the subgraphs** for each graph.
At inference: majority voting on the outputs of k independent samplings ($k = 5$).

Benefits:

1. Enables running on larger graphs
2. Allows increasing the batch size resulting in faster runtime

Sampling experiments

Sampling does not significantly reduce accuracy, and in some cases even **increases accuracy**

		OGBG-MOLHIV	OGBG-MOLTOX21	EXP	CEXP
GIN		75.58±1.40	74.91±0.51	51.2±2.1	70.2±4.1
DS-GNN (GIN) (ED)	**100%**	76.43±2.12	75.12±0.50	100±0.0	100±0.0
	50%	76.29±1.33	74.59±0.71	100±0.0	100±0.0
	20%	76.57±1.48	75.67±0.89	100±0.0	99.9±0.2
	5%	77.82±1.00	76.39±1.11	99.7±0.4	99.9±0.2
DS-GNN (GIN) (ND)	**100%**	76.19±0.96	75.34±1.21	100±0.0	100±0.0
	50%	77.23±1.32	74.82±1.05	100±0.0	99.9±0.2
	20%	77.65±0.84	75.66±0.46	100±0.0	99.9±0.2
	5%	78.26±1.02	76.51±1.04	97.2±1.1	99.8±0.8

Runtime

In practice, the additional computational complexity is empirically tractable.

Table: Timing comparison (in seconds) per epoch on a RTX 2080 GPU.

	NCI1			ZINC	
	BASELINE	100% SUBGRAPHS	20% SUBGRAPHS	BASELINE	100% SUBGRAPHS
GIN	1.00 ± 0.05	-	-	1.45 ± 0.01	-
DS-GNN (GIN) (ED)	-	2.07 ± 0.01	1.73 ± 0.01	-	3.56 ± 0.03
DS-GNN (GIN) (ND)	-	2.08 ± 0.03	1.71 ± 0.01	-	3.42 ± 0.02
DS-GNN (GIN) (EGO)	-	1.96 ± 0.01	1.72 ± 0.01	-	3.02 ± 0.04
DS-GNN (GIN) (EGO+)	-	2.01 ± 0.02	1.73 ± 0.01	-	3.09 ± 0.04
DSS-GNN (GIN) (ED)	-	3.26 ± 0.07	2.94 ± 0.01	-	4.25 ± 0.01
DSS-GNN (GIN) (ND)	-	3.19 ± 0.07	2.91 ± 0.02	-	4.12 ± 0.03
DSS-GNN (GIN) (EGO)	-	3.14 ± 0.04	2.79 ± 0.02	-	3.63 ± 0.02
DSS-GNN (GIN) (EGO+)	-	3.19 ± 0.03	2.90 ± 0.01	-	3.69 ± 0.05

Outline

Other similar methods

Very recently, other methods of various motivations have been proposed:

- Reconstruction GNN [Cotta et al., 2021] $\approx$ DS-GNN (ND)
- DropGNN [Papp et al., 2021] $\approx$ DS-GNN (ND) with sampling
- GNN-AK [Zhao et al., 2021] $\approx$ DSS-GNN (EGO)

Our ESAN framework **unifies** and **generalizes** these methods with bags of subgraphs and equivariance!!

Conclusions

The ESAN framework:

1. decomposes a graph into a set of subgraphs (bag)
2. processes the bag respecting the emerging symmetry

Experimentally: competitive performance on various graph-wise benchmarks

ESAN naturally enjoys:

- provable expressiveness
- native equivariance
- being domain-agnostic

Future directions:

- other policies?
- structured bags
- stochastic policies & approx. equivariance

References I

Abboud, R., Ceylan, I. I., Grohe, M., and Lukasiewicz, T. (2020).
The surprising power of graph neural networks with random node initialization.
arXiv preprint arXiv:2010.01179.

Beaini, D., Passaro, S., Létourneau, V., Hamilton, W. L., Corso, G., and Liò, P. (2021).
Directional graph networks.
In *International Conference on Machine Learning.*

Bodnar, C., Frasca, F., Otter, N., Wang, Y. G., Liò, P., Montúfar, G., and Bronstein, M. (2021).
Weisfeiler and lehman go cellular: CW networks.
In *Advances in Neural Information Processing Systems*, volume 34.

Bouritsas, G., Frasca, F., Zafeiriou, S., and Bronstein, M. M. (2020).
Improving graph neural network expressivity via subgraph isomorphism counting.
arXiv preprint arXiv:2006.09252.

References II

Chen, Z., Chen, L., Villar, S., and Bruna, J. (2020).
Can graph neural networks count substructures?
Advances in neural information processing systems.

Corso, G., Cavalleri, L., Beaini, D., Liò, P., and Veličković, P. (2020).
Principal neighbourhood aggregation for graph nets.
In *Advances in Neural Information Processing Systems.*

Cotta, L., Morris, C., and Ribeiro, B. (2021).
Reconstruction for powerful graph representations.
In *NeurIPS.*

Fey, M., Yuen, J. G., and Weichert, F. (2020).
Hierarchical inter-message passing for learning on molecular graphs.
In *ICML Graph Representation Learning and Beyond (GRL+) Workhop.*

References III

Maron, H., Ben-Hamu, H., Serviansky, H., and Lipman, Y. (2019).
Provably powerful graph networks.
In *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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609.

Papp, P. A., Martinkus, K., Faber, L., and Wattenhofer, R. (2021).
Dropgnn: Random dropouts increase the expressiveness of graph neural networks.
In *35th Conference on Neural Information Processing Systems (NeurIPS)*.

Sato, R., Yamada, M., and Kashima, H. (2021).
Random features strengthen graph neural networks.
In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 333–341. SIAM.

References IV

Vignac, C., Loukas, A., and Frossard, P. (2020).
Building powerful and equivariant graph neural networks with structural message-passing.
In *NeurIPS*.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019).
How powerful are graph neural networks?
In *International Conference on Learning Representations*.

Zhao, L., Jin, W., Akoglu, L., and Shah, N. (2021).
From stars to subgraphs: Uplifting any gnn with local structure awareness.