Abhiroop Sanyal
a5sanyal@uwaterloo.ca

David R. Cheriton School of Computer Science
University of Waterloo

Building powerful and equivariant graph neural networks with structural message-passing

Work By: Clement Vignac, Andreas Loukas, Pascal Frossard

NeurIPS, Vancouver, 2020

Outline

- Message Passing Neural Networks: Definition
- MPNNs: Advantages and Limitations
- Structural Message Passing: Architecture
- SMP: Equivariance
- SMP: Expressive power
- Implementation: Default and Fast-SMP
- Comparison: Time and Space Complexity, Synthetic and Real-World Date Sets

Message Passing Neural Networks: Definition

Notation

Consider a graph $G = (V, E)$ $(|V| = n,\ |E| = m)$ with node features $\mathbf{X} = (\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n})^T \in \mathbb{R}^{n \times c_X}$ and edge features $y_{ij} \in \mathbb{R}^{c_Y}$ for each (v_i, v_j) $\in E$. The neighbourhood of a vertex v_i is denoted as N_i.

Message Passing Neural Networks: Definition

Mesage Passing Neural Networks(MPNNs)

Each message-passing layer in an MPNN consists of the following applications:

1. Message and Aggregation step: $m_i^{(l)} = \text{AGG}\left(\{M^{(l)}\left(x_i^{(l)}, x_j^{(l)}, y_{ij}\right)\}_{j \in N_i}\right)$ and AGG is a symmetric function. In case of (Maron et al.) AGG is simply the sum of its inputs.

Message Passing Neural Networks: Definition

Mesage Passing Neural Networks(MPNNs)

Each message-passing layer in an MPNN consists of the following applications:

1. Message and Aggregation step: $m_i^{(l)} = \mathsf{AGG}\left(\{M^{(l)}\left(x_i^{(l)}, x_j^{(l)}, y_{ij}\right)\}_{j \in N_i}\right)$ and AGG is a symmetric function called the *aggregation* function. In case of (Maron et al.) AGG is simply the sum of its inputs.
2. Update Step: $x_i^{(l+1)} = \mathsf{UP}^{(l)}\left(m_i^{(l)}, x_i^{(l)}\right)$.

MPNNs: Advantages and Limitations

Following are some advantages of MPNNs:

- Permutation Equivariance (Corollary 1, Maron et al.):

Theorem (Permutation Equivariance)

Invariant and Equivariant GNNs can represent any message-passing network to arbitrary precision on compact sets.

MPNNs: Advantages and Limitations

Following are some advantages of MPNNs:

- Permutation Equivariance (Corollary 1, Maron et al.).
- Efficiently exploits sparsity of graphs.

MPNNs: Advantages and Limitations

Following are some advantages of MPNNs:

- Permutation Equivariance (Corollary 1, Maron et al.)
- Efficiently exploits sparsity of graphs
- Inductive bias: Tendency to learn relationships between nearby nodes making it effective for problems such as: (n-body problems, rigid-body collision etc.)

MPNNs: Advantages and Limitations

- Expressivity: (Xu et al.) [Unweighted Graphs]: MPNNs $\leq$ 1-WL ($=$ 2-WL)

MPNNs: Advantages and Limitations

- Expressivity: (Xu et al.) [Unweighted Graphs]: MPNNs $\leq$ 1-WL (= 2-WL). Cannot distinguish the following graphs:

MPNNs: Advantages and Limitations

- Expressivity (attributed/weighted case): (Chen et al.) proved the following:

Theorem (2-WL equivalence for MPNN)

Two attributed graphs that are indistinguishable by 2-WL cannot be distinguished by any MPNN.

MPNNs: Advantages and Limitations

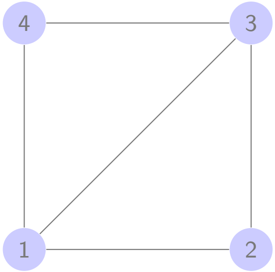
Induced subgraph count:

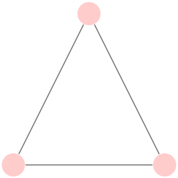
MPNNs: Advantages and Limitations

Induced subgraph count:

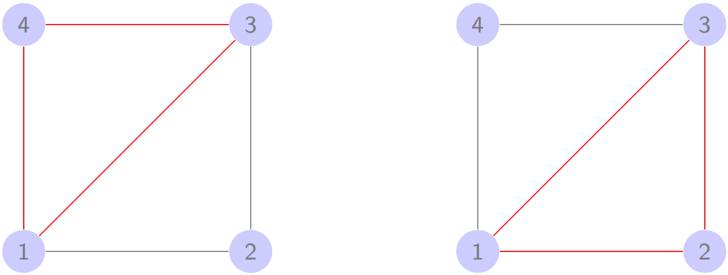
- G (n vertices), $G^{(p)}$ ($n^{(p)}$ vertices) $[n^{(p)} \leq n]$
- Question: How many induced subgraphs of G are isomorphic to $G^{(p)}$?

MPNNs: Advantages and Limitations





MPNNs: Advantages and Limitations



Induced-Subgraph-Count $= 2$

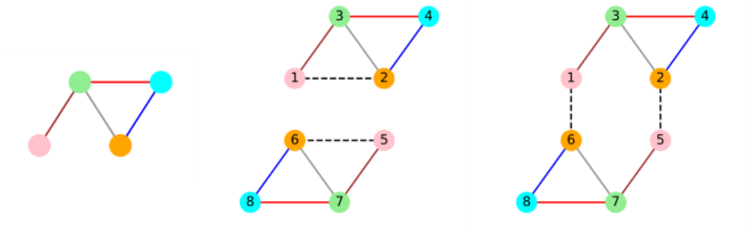
MPNNs: Advantages and Limitations

Induced subgraph count: (Chen at al.) showed:

Theorem (Distinguishing power of 2-WL)

2-WL cannot induced-subgraph-count any connected pattern with ≥ 3 nodes.

MPNNs: Advantages and LImitations



$G^{[\mathrm{P}]}$ $\quad G^{[1]}$ $\quad G^{[2]}$

MPNNs: Advantages and Limitations

Several other limitations to expressive power of MPNNs have been subsequently proved: in molecular design (Elton et al. 2019) and combinatorial optimization (Sato et al. 2019).

MPNNs: Advantages and Limitations

Intuition:

- Only obtains local view of graph from message passing

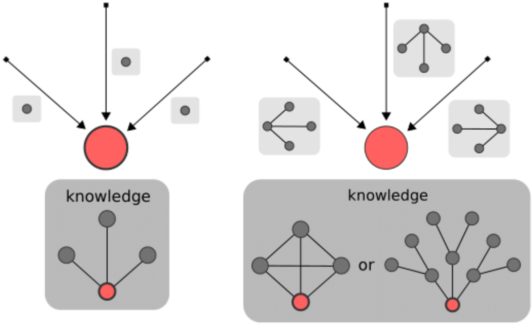
MPNNs: Advantages and Limitations

Intuition:

- Only obtains local view of graph from message passing
- Cannot tell how many unique nodes messages are coming from.

Intuition:

- Only obtains local view of graph from message passing
- Cannot tell how many unique nodes messages are coming from.

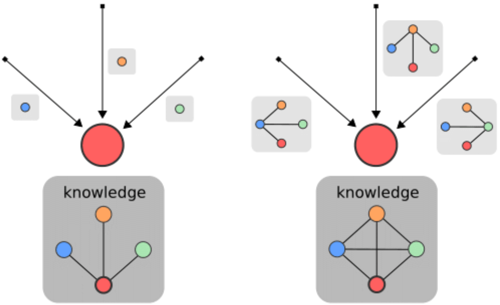


MPNNs: Advantages and Limitations

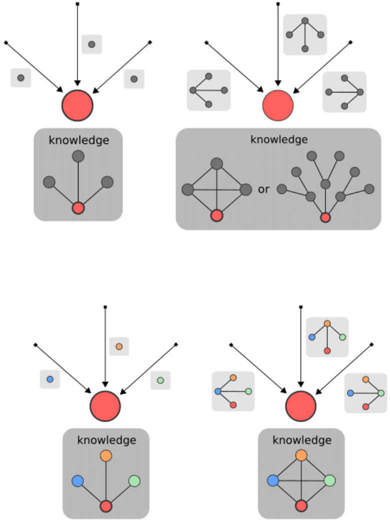
Several works have tried to overcome this problem by providing nodes with node identifiers.

MPNNs: Advantages and Limitations

Several works have tried to overcome this problem by providing nodes with node identifiers.



MPNNs: Advantages and Limitations



MPNNs: Advantages and Limitations

In particular, (Loukas, 2019) show that MPNNs with node identifiers, unbounded width, depth $\geq diam(\mathrm{G})$ and sufficiently powerful message and update functions are Turing-Universal.

MPNNs: Advantages and Limitations

Issues: randomly selected node identifiers leads to poor generalization and loss of permutation equivariance.

MPNNs: Advantages and Limitations

Issues: randomly selected node identifiers leads to poor generalization and loss of permutation equivariance.

More clever identifiers: Deferred to experiments!

Structural Mesage Passing

The authors propose Structural Message Passing as an improvement to maintain expressive power and permutation equivariance at the same time.

Structural Mesage Passing

The authors propose Structural Message Passing as an improvement to maintain expressive power and permutation equivariance at the same time.

Key Difference: Store matrix $U_i \in \mathbb{R}^{n \times c}$ [called "local context"] instead of vector x_i

Structural Mesage Passing

Key Difference: Store matrix $U_i \in \mathbb{R}^{n \times c}$ instead of vector x_i. U_{ij} is the c-dimensional representation v_i has of v_j.

Structural Mesage Passing

Key Difference: Store matrix $U_i \in \mathbb{R}^{n \times c}$ instead of vector x_i. U_{ij} is the c-dimensional representation v_i has of v_j.

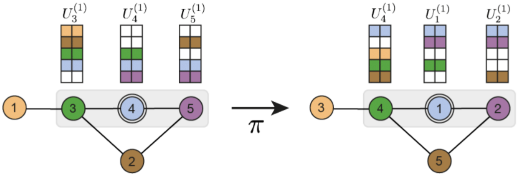


Figure 1: In the SMP model, each local context $\boldsymbol{U}_i^{(l)}$ is an $n \times c_l$ matrix, with each row storing the c_l-dimensional representation of a node (denoted by color). The figure shows the local context in the output of the first layer and blank rows correspond to nodes that have not been encountered yet. Upon node reordering, the lines of the local context are permuted but their content remains unchanged.

Structural Message Passing: Architecture

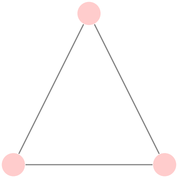
- Initialization: Let $U_i^{(l)}$ denote the local context of v_i at layer l. Initially, $U_i^{(0)}$ is defined as as:

$$U_i^{(0)}[i,:] = [1, \mathbf{x}_i]$$

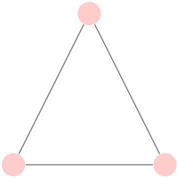
where $\mathbf{x}_i$ is the feature vector associated with node i.

Structural Message Passing: Architecture

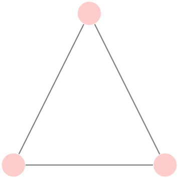
Assumption: $x_i := \deg(v_i) \in \mathbb{R}$.



Assumption: $x_i := \deg(v_i) \in \mathbb{R}$.



$$U_1^{(0)} = \begin{bmatrix} 1 & 2 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$



$$U_1^{(0)} = \begin{bmatrix} 1 & 2 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$U_2^{(0)} = \begin{bmatrix} 0 & 0 \\ 1 & 2 \\ 0 & 0 \end{bmatrix}$$

$$U_3^{(0)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 2 \end{bmatrix}$$

Structural Message Passing: Architecture

- Initialization:

$$U_i^{(0)}[i, :] = [1, \mathbf{x}_i]$$

- Layers:

  1. Message and Aggregation step:
     $\hat{U_i}^{(l)} = \mathsf{AGG}\left(\{M^{(l)}\left(U_i^{(l)}, U_j^{(l)}, y_{ij}\right)\}_{j \in N_i}\right)$
  2. Update Step: $U_i^{(l+1)} = \mathsf{UP}^{(l)}\left(U_i^{(l)}, \hat{U_i}^{(l)}\right)$.

Structural Message Passing: Architecture

The l-th SMP layer can be expresed in tensor form as follows:

$$\mathbf{U}^{l+1} = [U_1^{(l+1)}, ..., U_n^{(l+1)}] := f^{(l)} \left(\mathbf{U}^{(l)}, \mathbf{Y}, A \right)$$

Here $\mathbf{Y} \in \mathbb{R}^{n \times n \times c_Y}$ contains edge features.

Structural Message Passing: Architecture

- Initialization
- Layers (L many)
- Pooling: (for clasification purpose) For node classification an equivariant network for sets is applied simultaneously to each node

$$f_{eq}\left(\mathbf{U}^{(0)}, \mathbf{Y}, A\right) = \sigma \circ f^{(L)} \cdots \circ f^{(1)}\left(\mathbf{U}^{(0)}, \mathbf{Y}, A\right)$$

Structural Message Passing: Architecture

- Initialization
- Layers (L many)
- Pooling: (for clasification purpose)

$$f_{eq}\left(\mathbf{U}^{(0)}, \mathbf{Y}, A\right) = \sigma \circ f^{(L)} \cdots \circ f^{(1)}\left(\mathbf{U}^{(0)}, \mathbf{Y}, A\right)$$

For permutation invariance once uses a symmetric pooling function (sum or average followed by softmax):

$$f_{inv} = \textit{pool} \circ f_{eq}\left(\mathbf{U}^{(0)}, \mathbf{Y}, A\right)$$

SMP: Equivariance

Let $\pi \in S_n$ (the group of permutations on n elements). The action of π on a tensor $\mathbf{U} \in \mathbb{R}^{n \times n \times c}$ is given by:

$$\pi \cdot \mathbf{U}[i, j, k] := \mathbf{U}\left[\pi^{-1}(i), \pi^{-1}(j), k\right]$$

Definition (Permutation Equivariance)

An SMP-layer $f^{(r)}$ is permutation equivariant if the following holds for all $\pi \in S_n$:

$$\pi \cdot f^{(r)}(\mathbf{U}, \mathbf{Y}, A) = f^{(r)}(\pi \cdot \mathbf{U}, \pi \cdot \mathbf{Y}, \pi \cdot A)$$

SMP: Equivariance

Theorem (Permutation Equivariance of SMP)

Suppose for all permutation π in S_n, we have the following:
$UP\left(\pi \cdot \mathbf{U}, \pi \cdot \tilde{\mathbf{U}}\right) = \pi \cdot UP\left(\mathbf{U}, \tilde{\mathbf{U}}\right)$,
$AGG\left(\{\pi \cdot \mathbf{U}_j\}_{v_j \in N_i}\right) = \pi \cdot AGG\left(\{\mathbf{U}_j\}_{v_j \in N_i}\right)$ *and*
$M\left(\pi \cdot \mathbf{U}, \pi \cdot \tilde{\mathbf{U}}, \mathbf{y}\right) = \pi \cdot M\left(\mathbf{U}, \tilde{\mathbf{U}}, \mathbf{y}\right)$. *Then SMP is permutation equivariant.*

SMP: Expressive Power

Theorem (Expressive power of MPNNs)

Consider the family $\mathcal{F}$ of graphs with n nodes. Then $\exists$ a permutation-equivariant SMP network f of "large enough" depth and width such that for any two graphs $G_1 = (A_1, \mathbf{X}_1, \mathbf{Y}_1)$ and $G_2 = (A_2, \mathbf{X}_2, \mathbf{Y}_2)$, the following holds for every pair of vertices $v_i \in V_1$ and $v_j \in V_2$:

- *If G_1 and G_2 are not isomorphic, then $\forall\ \pi \in S_n$, we have:*

$$\pi \cdot f(A_1, \mathbf{Y}_1, \mathbf{X}_1)[i, :, :] \neq f(A_2, \mathbf{Y}_2, \mathbf{X}_2)[j, :, :]$$

- *If G_1 and G_2 are isomorphic, then $\exists\ \pi \in S_n$, not depending on i and j such that:*

$$\pi \cdot f(A_1, \mathbf{Y}_1, \mathbf{X}_1)[i, :, :] = f(A_2, \mathbf{Y}_2, \mathbf{X}_2)[j, :, :]$$

SMP: Expressive Power

Theorem (Expressive power of MPNNs)

Consider the family $\mathcal{F}$ of graphs with n nodes. Then $\exists$ a permutation-equivariant SMP network f of "large enough" depth and width such that for any two graphs $G_1 = (A_1, \mathbf{X}_1, \mathbf{Y}_1)$ and $G_2 = (A_2, \mathbf{X}_2, \mathbf{Y}_2)$, the following holds for every pair of vertices $v_i \in V_1$ and $v_j \in V_2$:

- *If G_1 and G_2 are not isomorphic, then $\forall\ \pi \in S_n$, we have:*

$$\pi \cdot f(A_1, \mathbf{Y}_1, \mathbf{X}_1)[i, :, :] \neq f(A_2, \mathbf{Y}_2, \mathbf{X}_2)[j, :, :]$$

- *If G_1 and G_2 are isomorphic, then $\exists\ \pi \in S_n$, not depending on i and j such that:*

$$\pi \cdot f(A_1, \mathbf{Y}_1, \mathbf{X}_1)[i, :, :] = f(A_2, \mathbf{Y}_2, \mathbf{X}_2)[j, :, :]$$

The second condition is a consequence of permutation equivariance.

SMP: Expressive Power

If $\mathcal{F}$ has graphs on diameter at most Δ and width at most D, then $\exists\ f$ of depth at most $\Delta + 1$ and width at most $2D + c_X + nc_Y$.

SMP $\geq$ MPNN

Theorem (SMP $\geq$ MPNN)

Suppose an MPNN f maps $x_i^{(0)}$ to $x_i^{(L)}$. Then, $\exists$ perm-equiv. SMP g with L layers such that:

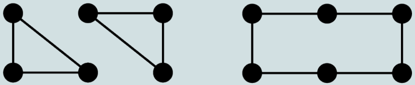
1. $\forall\ l \leq L$: $\mathbf{U}^{(l)}[i, i, :] = x_i^{(l)}$
2. $\forall\ l \leq L$, $\forall\ j \neq i$: $\mathbf{U}^{(l)}[i, j, :] = 0$

Theorem (SMP $>$ MPNN)

$\exists$ an SMP network f that gives different outputs for the following graphs while any MPNN will view these graphs as isomorphic.

SMP $> GNN$

Theorem (SMP > MPNN)



Proof.

MPNNs cannot distinguish these graphs because 1-WL cannot (Xu et al.). $\square$

Theorem (SMP > MPNN)

Proof.

Consider an SMP of 3 layers with $U_i^{(0)} = 1_i$ and update:

$$U_i^{(l+1)} = \sum_{v_j \in N_i} U_j^{(l)}$$

$\square$

SMP $\geq$ MPNN

Theorem (SMP > MPNN)

Proof.

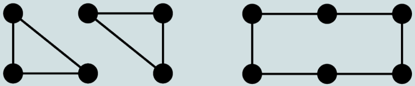
Consider an SMP of 3 layers with $U_i^{(0)} = 1_i$ and update:

$$U_i^{(l+1)} = \sum_{v_j \in N_i} U_j^{(l)}$$

Note that $\mathbf{U}^{(3)} = A^3$ where A is the adjacency matrix of the input [proof on board]. $\square$

SMP $\geq$ MPNN

Theorem (SMP > MPNN)



Proof.

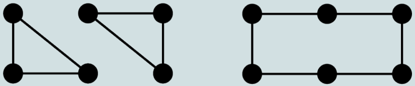
Consider an SMP of 3 layers with $U_i^{(0)} = 1_i$ and update:

$$U_i^{(l+1)} = \sum_{v_j \in N_i} U_j^{(l)}$$

Note that $\mathbf{U}^{(3)} = A^3$ where A is the adjacency matrix of the input.

Use $tr(\cdot)$ as $pool$ (perm-invariant). $\square$

Theorem (SMP > MPNN)



Proof.

Consider an SMP of 3 layers with $U_i^{(0)} = 1_i$ and update:

$$U_i^{(l+1)} = \sum_{v_j \in N_i} U_j^{(l)}$$

Note that $\mathbf{U}^{(3)} = A^3$ where A is the adjacency matrix of the input .

Use $tr(\cdot)$ as $pool$ (perm-invariant). Then $f(G_1) = 2 \neq 0 = f(G_2)$ □

Similarity to PPGN

Theorem

A fast-SMP with k layers can be approximated by a $2k$-block PPGN.

Implementation of SMPs: Default Architecture

The default version of SMP has an architecture similar to that of standard MPNNs.

- Message Function: $M_{\text{def}}\left(U_i^{(l)}, U_j^{(l)}\right) = MLP\left(U_i^{(l)}, U_j^{(l)}, y_{ij}\right)$

Implementation of SMPs: Default Architecture

- Message Function: $M_{\text{def}}\left(U_i^{(l)}, U_j^{(l)}\right) = MLP\left(U_i^{(l)}, U_j^{(l)}, y_{ij}\right)$
- Aggregation: Sum of all messages normalized by the average degree of the graph.

Implementation of SMPs: Default Architecture

- Message Function: $M_{\text{def}}\left(U_i^{(l)}, U_j^{(l)}\right) = MLP\left(U_i^{(l)}, U_j^{(l)}, y_{ij}\right)$
- Aggregation: Sum of all messages normalized by the average degree (d_{avg}) of the graph.

(Velickovic et al., 2020) : Normalized aggregator avoids exploding norm problem.

Implementation of SMPs: Default Architecture

- Aggregation: $\hat{U_i}^{(l)} = \frac{\sum_{v_j \in N_i} MLP\left(U_i^{(l)}, U_j^{(l)}, y_{ij}\right)}{d_{avg}}$

Implementation of SMPs: Default Architecture

- Aggregation: $\hat{U_i}^{(l)} = \frac{\sum_{v_j \in N_i} MLP\left(U_i^{(l)}, U_j^{(l)}, y_{ij}\right)}{d_{avg}}$
- Update: $U_i^{(l+1)} = MLP\left(U_i^{(l)}, \hat{U_i}^{l}\right)$

Implementation of SMPs: Fast SMP

Assumption: Unweighted Graphs.

Implementation of SMPs: Fast SMP

- Message function: $M_{fast}^{(l)}\left(U_i^{(l)}, U_j^{(l)}\right) = U_j^{(l)} + \left(U_i^{(l)} W_1^{(l)}\right) \odot \left(U_j^{(l)} W_2^{(l)}\right)$

where $W_1^{(l)}$ and $W_2^{(l)}$ are learnable matrices.

Implementation of SMPs: Fast SMP

- Message function: $M_{fast}^{(l)}\left(U_i^{(l)}, U_j^{(l)}\right) = U_j^{(l)} + \left(U_i^{(l)} W_1^{(l)}\right) \odot \left(U_j^{(l)} W_2^{(l)}\right)$

where $W_1^{(l)}$ and $W_2^{(l)}$ are learnable matrices.

- The aggregation function is the same as before.

Implementation of SMPs: Fast SMP

- The aggregation function is the same as before.

Update Step: $U_i^{(l+1)} = U_i^{(l)} + \frac{\sum_{v_j \in N_i} M_{fast}^{l}\left(U_i^{(l)}, U_j^{(l)}\right)}{d_{avg}}$

Why is Fast SMP faster

Expand update equation (using linearity):

$$U_i^{(l+1)} = U_i^{(l)} + \frac{1}{d_{avg}} \times \left[\sum_{v_j \in N_i} \left(U_j^{(l)} + \left(U_i^{(l)} W_1^{(l)} \right) \odot \left(U_j^{(l)} W_2^{(l)} \right) \right) \right]$$

Why is Fast SMP faster

Expand update equation (using linearity):

$$
\begin{aligned}
U_i^{(l+1)} &= U_i^{(l)} + \frac{1}{d_{avg}} \times \left[\sum_{v_j \in N_i} \left(U_j^{(l)} + \left(U_i^{(l)} W_1^{(l)} \right) \odot \left(U_j^{(l)} W_2^{(l)} \right) \right) \right] \\
&= U_i^{(l)} + \frac{1}{d_{avg}} \times \left[\sum_{v_j \in N_i} U_j^{(l)} + \left(U_i^{(l)} W_1^{(l)} \right) \odot \left(\sum_{v_j \in N_i} \left(U_j^{(l)} W_2^{(l)} \right) \right) \right]
\end{aligned}
$$

Why is Fast SMP faster

$$U_i^{(l+1)} = U_i^{(l)} + \frac{1}{d_{avg}} \times \left[\sum_{v_j \in N_i} U_j^{(l)} + \left(U_i^{(l)} W_1^{(l)} \right) \odot \left(\sum_{v_j \in N_i} \left(U_j^{(l)} W_2^{(l)} \right) \right) \right]$$

In the above equation, input to both sums is the local contexts of the vertices v_j.

Why is Fast SMP faster

$$U_i^{(l+1)} = U_i^{(l)} + \frac{1}{d_{avg}} \times \left[\sum_{v_j \in N_i} U_j^{(l)} + \left(U_i^{(l)} W_1^{(l)} \right) \odot \left(\sum_{v_j \in N_i} \left(U_j^{(l)} W_2^{(l)} \right) \right) \right]$$

Contrast with update for default SMP:

$$U_i^{(l+1)} = MLP \left(U_i^{(l)}, \frac{1}{d_{avg}} \times \sum_{v_j \in N_i} MLP \left(U_i^{(l)}, U_j^{(l)}, y_{ij} \right) \right)$$

Why is Fast-SMP faster

- Computes one message per node instead of one per edge

Comparison: Time and Space Complexity

Table: Time and space complexity of the forward pass expressed in terms of number of nodes n, number of edges m, number of node colors χ, and width c. For connected graphs, we trivially have $\chi \leq n \leq m + 1 \leq n^2$.

Method	Memory per layer	Time complexity per layer
GIN	$\Theta(n\ c)$	$\Theta(m\ c + n\ c^2)$
MPNN	$\Theta(n\ c)$	$\Theta(m\ c^2)$
Fast SMP (with coloring)	$\Theta(n\ \chi\ c)$	$\Theta(m\ \chi\ c + n\ \chi\ c^2)$
Fast SMP	$\Theta(n^2\ c)$	$\Theta(m\ n\ c + n^2\ c^2)$
SMP	$\Theta(n^2\ c)$	$\Theta(m\ n\ c^2)$
PPGN	$\Theta(n^2\ c)$	$\Theta(n^3\ c + n^2\ c^2)$
Local order-3 WL	$\Theta(n^3\ c)$	$\Theta(n^4\ c + n^3\ c^2)$

Comparison of Performance: Synthetic data sets

Table 2: Experiments on cycle detection, viewed as a graph classification problem.

(a) Test accuracy on the detection of cycles of various length with 10,000 training samples. (Best seen in color.) Only SMP solves the problem in all configurations.

Cycle length	4				6				8			
Graph size	12	20	28	36	20	31	42	56	28	50	66	72
MPNN	98.5	93.2	91.8	86.7	98.7	95.5	92.9	88.0	98.0	96.3	92.5	89.1
GIN	98.3	97.1	95.0	93.0	99.5	97.2	95.1	92.7	98.5	98.8	90.8	92.5
GIN + degree	99.3	98.2	97.3	96.7	99.2	97.1	97.1	94.5	99.3	98.7		95.4
GIN + rand id	99.0	96.2	94.9	88.3	99.0	97.8	95.1	96.1	98.6	98.0	97.2	95.3
RP [18]	100	99.9	99.7	97.7	99.0	97.4	92.1	84.1	99.2	97.1	92.8	80.6
PPGN	100	100	100	99.8	98.3	99.4	93.8	87.1	99.9	98.7	84.4	76.5
Ring-GNN	100	99.9	99.9	99.9	100	100	100	100	99.1	99.8	74.4	71.4
SMP	100	100	100	100	100	100	100	100	100	100	100	99.9

(b) Test accuracy (%) when evaluating the generalization ability of inductive networks. Each network is trained on one graph size ("In-distribution"), validated on a second size, then tested on a third ("Out-of-distribution"). SMP is the only powerful network evaluated that generalizes well. *OOM* = out of memory.

Setting	In-distribution			Out-of-distribution		
Cycle length	4	6	8	4	6	8
Graph size	20	31	50	36	56	72
GIN	93.9	99.7	98.8	81.1	85.8	**88.8**
PPGN	99.9	99.5	98.7	50.0	50.0	50.0
Ring-GNN	100	100	99.9	50.0	50.0	*OOM*
SMP	100	99.8	99.5	**99.8**	**87.8**	79.5

(c) Test accuracy (%) on the detection of 6 cycles for graphs with 56 nodes trained on less data. Thanks to its equivariance properties, SMP requires much less data for training.

Train samples	200	500	1000	5000
GIN + random identifiers	65.8	70.8	80.6	96.4
SMP	**87.7**	**97.4**	**97.6**	**99.5**

Comparison of Performance: Real-World Data Sets

Table 4: Mean absolute error (MAE) on ZINC, trained on a subset of 10k molecules.

Model	No edge features	With edge features
Gated-GCN [53]	0.435	0.282
GIN [53]	0.408	0.252
PNA [50]	0.320	0.188
DGN [54]	**0.219**	0.168
MPNN-JT [53]	–	0.151
MPNN (ablation)	0.272	0.189
SMP (Ours)	**0.219**	**0.138**