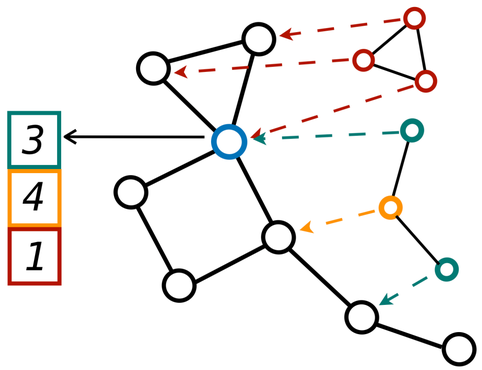
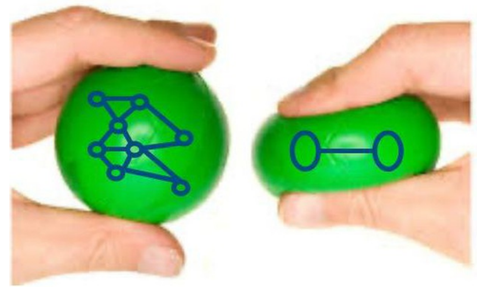
Graph Substructures against Graph Isomorphism:

from graph discrimination to compression

Giorgos Bouritsas

Imperial College
London

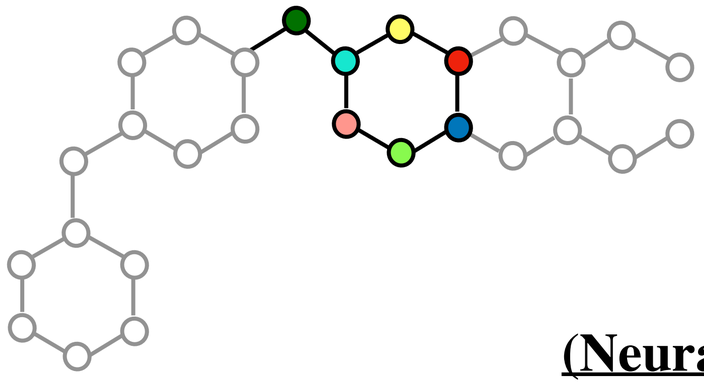




Graph-level learning: general problem setup

Graph $G = (\mathscr{V}, \mathscr{E}, \mathbf{u}_{\mathscr{V}}, \mathbf{u}_{\mathscr{E}})$ sampled from distribution p over a space of graphs $\mathfrak{G}$

$\mathscr{V}, \mathscr{E}, \mathbf{u}_{\mathscr{V}}, \mathbf{u}_{\mathscr{E}}$: vertex & edge set, node & edge features



Hypothesis (e.g. neural net) h

$\longrightarrow$

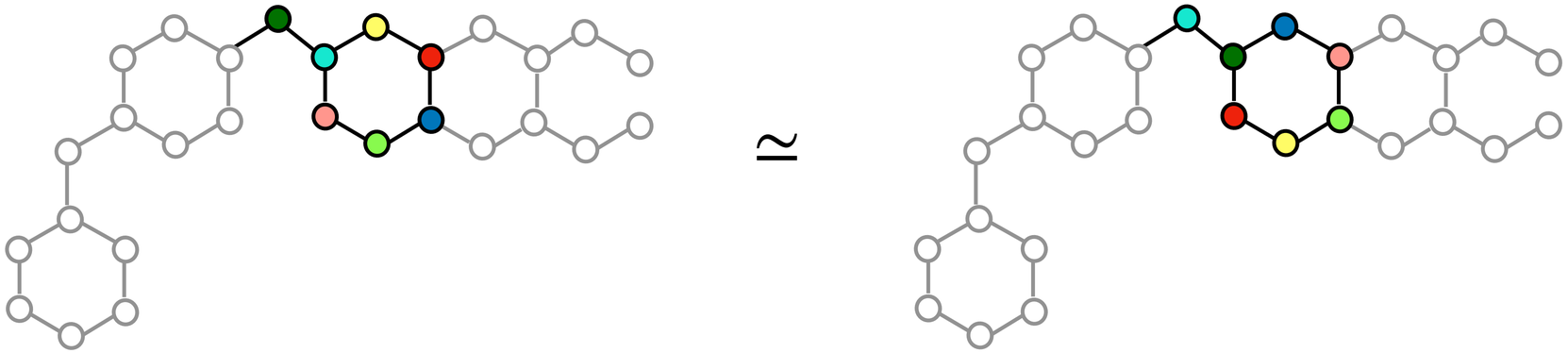
$y \in \mathscr{Y}$ (e.g. regression/classification)

<u>(Neural) graph function approximation:</u>

Find an h that optimises a given objective using observations sampled from p

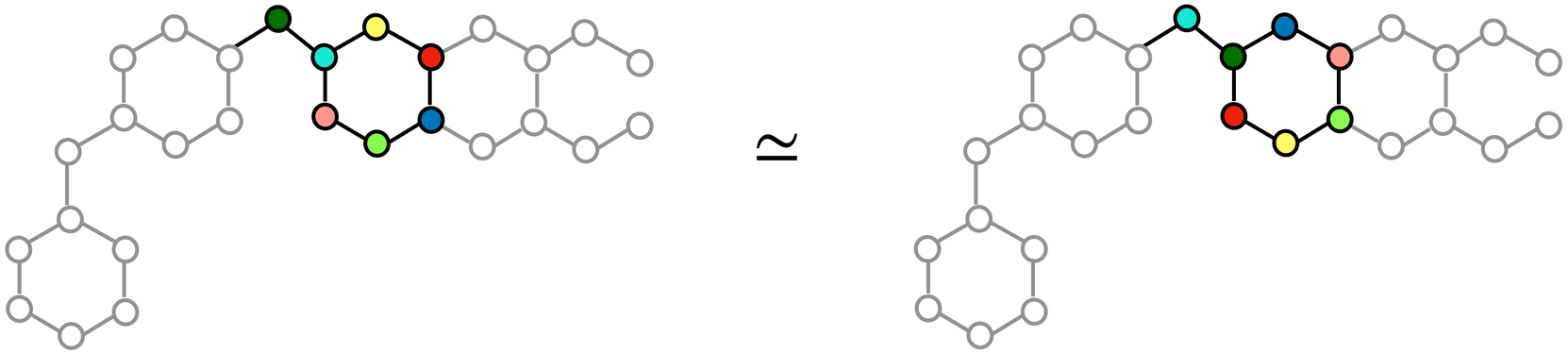
Part I: Graph Discrimination

Reminder: Message-Passing NNs



Condition: $G_1 \simeq G_2 \Rightarrow h(G_1) = h(G_2)$ (GI-invariant function*)

Reminder: Message-Passing NNs



Condition: $G_1 \simeq G_2 \Rightarrow h(G_1) = h(G_2)$ (GI-invariant function*)

Permutation invariant

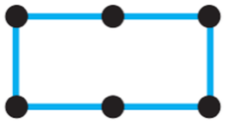
Algorithm 1 MPNN

Input: graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{u}_\mathcal{V}, \mathbf{u}_\mathcal{E})$
Output: graph representation $h(G)$
Initialise node hidden states: $\mathbf{x}_0(i) = \mathbf{u}_\mathcal{V}(i),\ i \in \mathcal{V}$
for layer $t = 1, \ldots, T$ **do**
  for each node $i \in \mathcal{V}$ **do**
    $\mathbf{m}_t(i) = \mathrm{AGGR}_t\Big(\wr\big(\mathbf{x}_{t-1}(i), \mathbf{x}_{t-1}(j), \mathbf{u}_\mathcal{E}(i,j)\big)\int_{j \in \mathcal{N}(i)}\Big)$; ▷ messages
    $\mathbf{x}_t(i) = \mathrm{UP}_t\big(\mathbf{x}_{t-1}(i), \mathbf{m}_t(i)\big)$; ▷ updates
  end for
end for
$h(G) = \mathrm{READ}\Big(\wr\mathbf{x}_T(i)\int_{i \in \mathcal{V}}\Big)$

MPNNs have limited expressivity

Theorem [Xu et al., ICLR 2019, Morris et al., AAAI 2019]:
MPNNs are at most as expressive as the Weisfeiler Leman isomorphism test

Consequence: MPNNs cannot discriminate many graph pairs and compute many graph properties

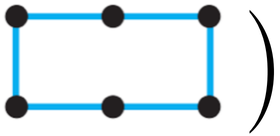
MPNN() = MPNN()

Maron et al., NeurIPS 2019

MPNNs have limited expressivity

Theorem [Xu et al., ICLR 2019, Morris et al., AAAI 2019]:
MPNNs are at most as expressive as the Weisfeiler Leman isomorphism test

Consequence: MPNNs cannot discriminate many graph pairs and compute many graph properties

MPNN() = MPNN()

Maron et al., NeurIPS 2019

Towards improving GNN expressivity:

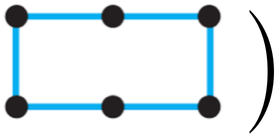
Expressivity vs Computational complexity vs GI-invariance trade-off:

Graph Isomorphism is (?) computationally expensive

MPNNs have limited expressivity

Theorem [Xu et al., ICLR 2019, Morris et al., AAAI 2019]:
MPNNs are at most as expressive as the Weisfeiler Leman isomorphism test

Consequence: MPNNs cannot discriminate many graph pairs and compute many graph properties

MPNN() = MPNN()

Maron et al., NeurIPS 2019

Towards improving GNN expressivity:

Expressivity vs Computational complexity vs GI-invariance trade-off:

Graph Isomorphism is (?) computationally expensive

- Attempt 1: Sacrifice computational complexity (e.g. k-IGNs)
- Attempt 2: Sacrifice GI-invariance (e.g. unique vertex ids)

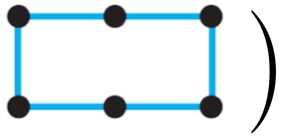
Maron et al.,
ICLR, ICML, NeurIPS'19

Loukas, ICLR'20

MPNNs have limited expressivity

Theorem [Xu et al., ICLR 2019, Morris et al., AAAI 2019]:
MPNNs are at most as expressive as the Weisfeiler Leman isomorphism test

Consequence: MPNNs cannot discriminate many graph pairs and compute many graph properties

MPNN() = MPNN()

Maron et al., NeurIPS 2019

Towards improving GNN expressivity:

Our choice:

Expressivity vs Computational complexity vs GI-invariance trade-off:

Graph Isomorphism is (?) computationally expensive

- Attempt 1: **Sacrifice computational complexity** (e.g. k-IGNs)
- Attempt 2: Sacrifice GI-invariance (e.g. unique vertex ids)

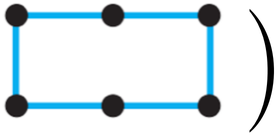
Maron et al.,
ICLR, ICML, NeurIPS'19

Loukas, ICLR'20

MPNNs have limited expressivity

Theorem [Xu et al., ICLR 2019, Morris et al., AAAI 2019]:
MPNNs are at most as expressive as the Weisfeiler Leman isomorphism test

Consequence: MPNNs cannot discriminate many graph pairs and compute many graph properties

MPNN() = MPNN()

Maron et al., NeurIPS 2019

Towards improving GNN expressivity:

Expressivity vs Computational complexity vs GI-invariance trade-off:

Graph Isomorphism is (?) computationally expensive

Our choice:

- Attempt 1: **Sacrifice computational complexity** (e.g. k-IGNs)

Maron et al., ICLR, ICML, NeurIPS'19

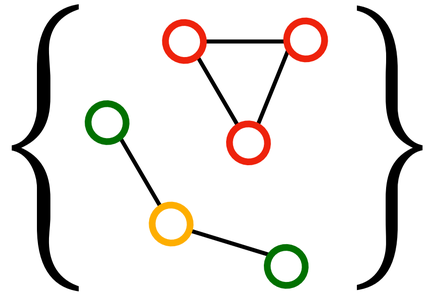
- Attempt 2: Sacrifice GI-invariance (e.g. unique vertex ids)

Loukas, ICLR'20

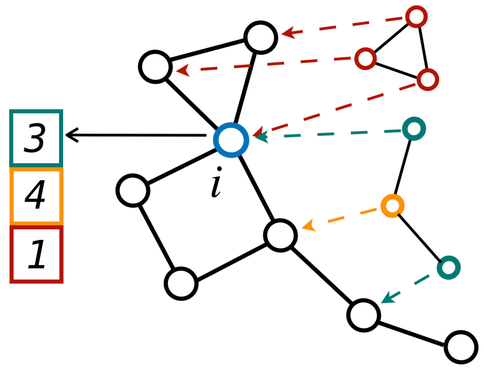
With ideas from

Graph Substructure Networks (GSN)

Dictionary $\mathscr{D} = \{a_1, a_2, \ldots, a_{|\mathscr{D}|}\}$

$\mathscr{D} =$ 

Substructure counting: vcount$(G; \mathscr{D})$

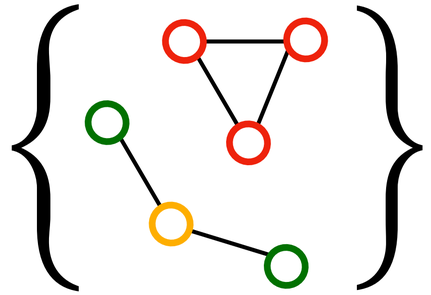
$\mathbf{w}_{\mathscr{V}}(i) =$ 

Algorithm 1 GSN-v

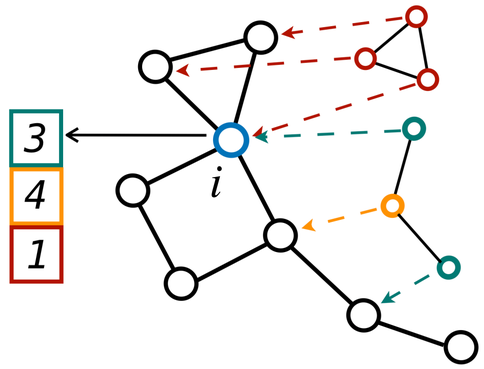
Input: graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{u}_{\mathcal{V}}, \mathbf{u}_{\mathcal{E}})$, dictionary $\mathcal{D}$
Output: graph representation $h(G)$
Precompute substructure counts $\mathbf{w}_{\mathcal{V}} = \texttt{vcount}(G; \mathcal{D})$
Initialise node hidden states: $\mathbf{x}_0(i) = \mathbf{u}_{\mathcal{V}}(i),\ i \in \mathcal{V}$
for layer $t = 1, \ldots, T$ **do**
  for each node $i \in \mathcal{V}$ **do**
    $\mathbf{m}_t(i) = \mathrm{AGGR}_t\Big(\wr\big(\mathbf{x}_{t-1}(i), \mathbf{x}_{t-1}(j), \mathbf{w}_{\mathcal{V}}(i), \mathbf{w}_{\mathcal{V}}(j), \mathbf{u}_{\mathcal{E}}(i,j)\big)\int_{j \in \mathcal{N}(i)}\Big)$;
    $\mathbf{x}_t(i) = \mathrm{UP}_t\big(\mathbf{x}_{t-1}(i), \mathbf{m}_t(i), \mathbf{w}_{\mathcal{V}}(i)\big)$;
  end for
end for
$h(G) = \mathrm{READ}\Big(\wr\mathbf{x}_T(i)\int_{i \in \mathcal{V}}\Big)$

Graph Substructure Networks (GSN)

Dictionary $\mathscr{D} = \{a_1, a_2, \ldots, a_{|\mathscr{D}|}\}$

$\mathscr{D} =$ 

Substructure counting: vcount$(G; \mathscr{D})$

$\mathbf{w}_{\mathscr{V}}(i) =$ 

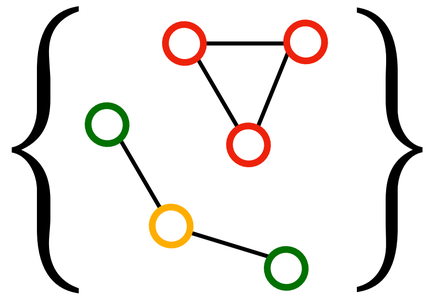
Increase in comp. complexity →

Algorithm 1 GSN-v

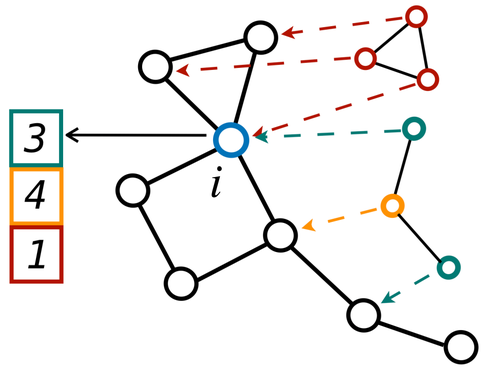
Input: graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{u}_{\mathcal{V}}, \mathbf{u}_{\mathcal{E}})$, dictionary $\mathcal{D}$
Output: graph representation $h(G)$
Precompute substructure counts $\mathbf{w}_{\mathcal{V}} = \texttt{vcount}(G; \mathcal{D})$
Initialise node hidden states: $\mathbf{x}_0(i) = \mathbf{u}_{\mathcal{V}}(i),\ i \in \mathcal{V}$
for layer $t = 1, \ldots, T$ **do**
  for each node $i \in \mathcal{V}$ **do**
    $\mathbf{m}_t(i) = \mathrm{AGGR}_t\Big(\wr\big(\mathbf{x}_{t-1}(i), \mathbf{x}_{t-1}(j), \mathbf{w}_{\mathcal{V}}(i), \mathbf{w}_{\mathcal{V}}(j), \mathbf{u}_{\mathcal{E}}(i,j)\big)\int_{j \in \mathcal{N}(i)}\Big)$;
    $\mathbf{x}_t(i) = \mathrm{UP}_t\big(\mathbf{x}_{t-1}(i), \mathbf{m}_t(i), \mathbf{w}_{\mathcal{V}}(i)\big)$;
  end for
end for
$h(G) = \mathrm{READ}\Big(\wr\mathbf{x}_T(i)\int_{i \in \mathcal{V}}\Big)$

Graph Substructure Networks (GSN)

Dictionary $\mathscr{D} = \{a_1, a_2, \ldots, a_{|\mathscr{D}|}\}$

$\mathscr{D} =$ 

Substructure counting: $\text{vcount}(G; \mathscr{D})$

$\mathbf{w}_{\mathscr{V}}(i) =$ 

Increase in comp. complexity

Algorithm 1 GSN-v

Input: graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{u}_{\mathcal{V}}, \mathbf{u}_{\mathcal{E}})$, dictionary $\mathcal{D}$
Output: graph representation $h(G)$
Precompute substructure counts $\mathbf{w}_{\mathcal{V}} = \texttt{vcount}(G; \mathcal{D})$
Initialise node hidden states: $\mathbf{x}_0(i) = \mathbf{u}_{\mathcal{V}}(i),\ i \in \mathcal{V}$
for layer $t = 1, \ldots, T$ **do**
  for each node $i \in \mathcal{V}$ **do**
    $\mathbf{m}_t(i) = \text{AGGR}_t\Big(\wr\big(\mathbf{x}_{t-1}(i), \mathbf{x}_{t-1}(j), \mathbf{w}_{\mathcal{V}}(i), \mathbf{w}_{\mathcal{V}}(j), \mathbf{u}_{\mathcal{E}}(i,j)\big)\int_{j \in \mathcal{N}(i)}\Big)$;
    $\mathbf{x}_t(i) = \text{UP}_t\big(\mathbf{x}_{t-1}(i), \mathbf{m}_t(i), \mathbf{w}_{\mathcal{V}}(i)\big)$;
  end for
end for
$h(G) = \text{READ}\Big(\wr\mathbf{x}_T(i)\int_{i \in \mathcal{V}}\Big)$

Akin to vertex ids

Graph Substructure Networks (GSN)

☑ **GI-invariance:** Subgraph counting is GI-invariant.

☑ **Improved expressivity:**

Theorem 1 (informal):
- *GSN is more expressive than MPNNs for the vast majority of dictionaries*.*
- *There exist dictionaries for which GSN can discriminate graphs that higher-order GNNs cannot**.*

Theorem 2 (informal): *GSN is universal for graphs of size* $\leq n$ *if* $\mathscr{D}$ *contains all graphs of size* $\leq n-1$

🤔 **Computational complexity:** Controllable trade-off between computational complexity and expressivity (*preprocessing*). Worst case $O(n^k)$.

*See also Arvind et al., FCT'19, Chen et al., NeurIPS'20
**See also Barcelo et al., NeurIPS'21

Graph Substructure Networks (GSN)

☑ **GI-invariance:** Subgraph counting is GI-invariant.

☑ **Improved expressivity:**

Theorem 1 (informal):
- *GSN is more expressive than MPNNs for the vast majority of dictionaries*.*
- *There exist dictionaries for which GSN can discriminate graphs that higher-order GNNs cannot**.*

Theorem 2 (informal): *GSN is universal for graphs of size* $\leq n$ *if* $\mathscr{D}$ *contains all graphs of size* $\leq n-1$

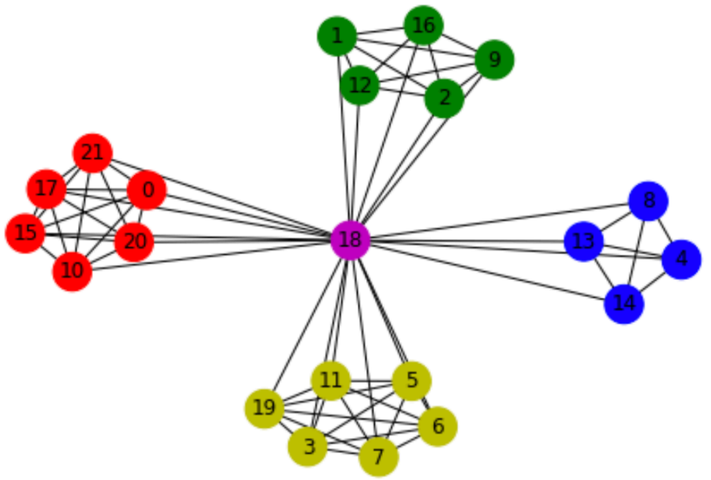
🤔 **Computational complexity:** Controllable trade-off between computational complexity and expressivity (*preprocessing*). Worst case $O(n^k)$.

🤔 **Generalisation:** ???

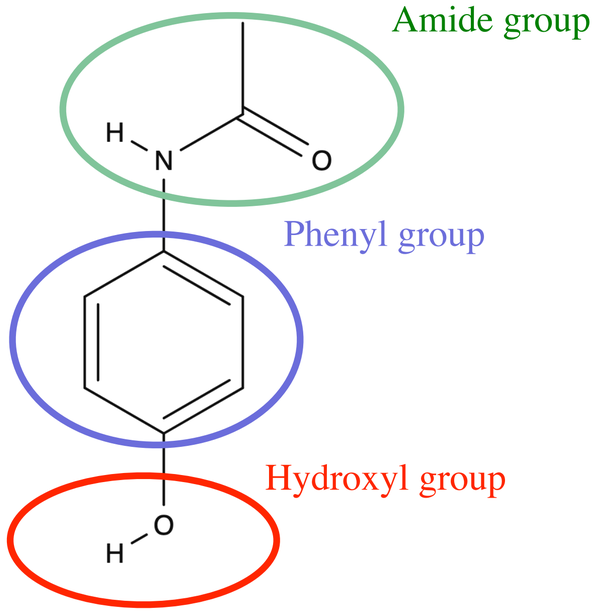
*See also Arvind et al., FCT'19, Chen et al., NeurIPS'20
**See also Barcelo et al., NeurIPS'21

The importance of substructures in downstream tasks

Social Networks**:**
Cliques indicate communities



Molecules**:**
functional groups characterise chemical properties



Graph Substructure Networks (GSN)

☑ **GI-invariance:** Subgraph counting is GI-invariant.

☑ **Improved expressivity:**

Theorem 1 (informal):
- *GSN is more expressive than MPNNs for the vast majority of dictionaries*.*
- *There exist dictionaries for which GSN can discriminate graphs that higher-order GNNs cannot**.*

Theorem 2 (informal): *GSN is universal for graphs of size* $\leq n$ *if* $\mathscr{D}$ *contains all graphs of size* $\leq n - 1$

🤔 **Computational complexity:** Controllable trade-off between computational complexity and expressivity (*preprocessing*). Worst case $O(n^k)$.

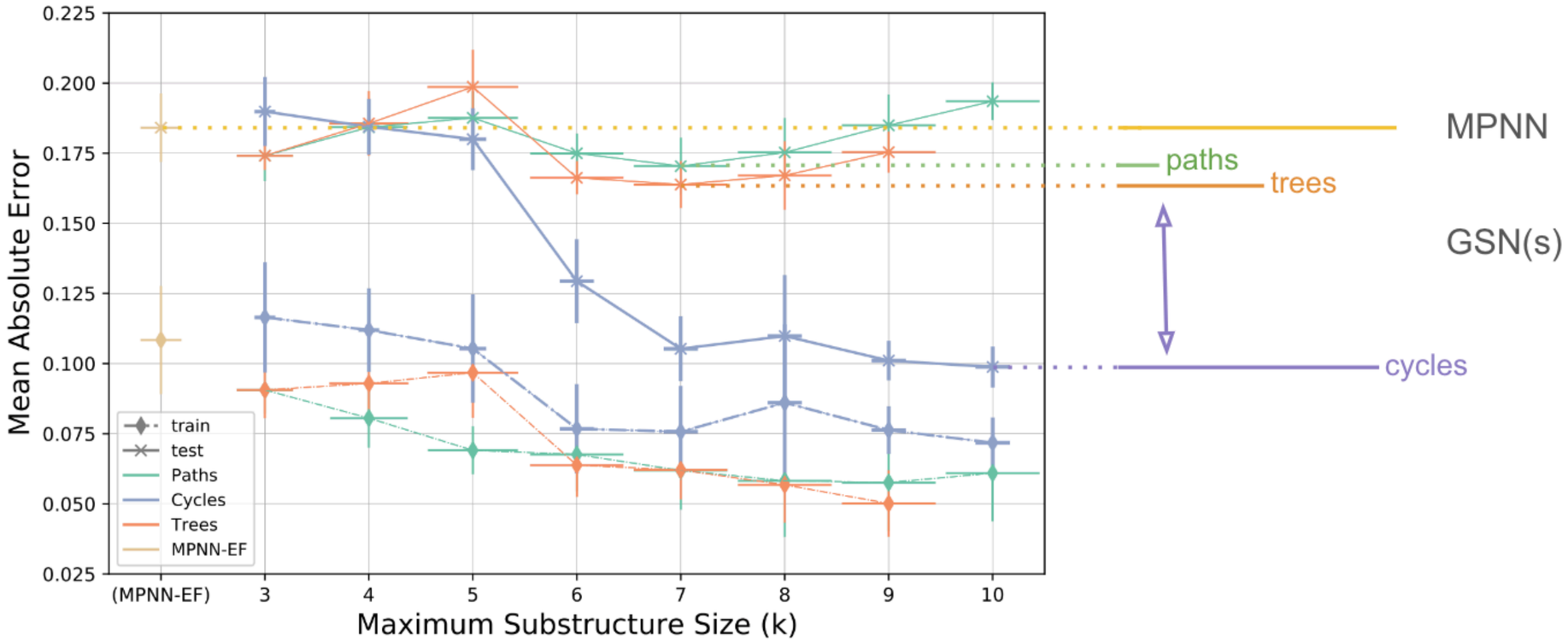
🤔 **Generalisation:** Dictionary selection based on domain knowledge improves generalisation

*See also Arvind et al., FCT'19, Chen et al., NeurIPS'20
**See also Barcelo et al., NeurIPS'21

Generalisation

Dataset	MUTAG	PTC	Proteins	NCI1	Collab	IMDB-B	IMDB-M
RWK* [Gärtner et al., 2003]	79.2±2.1	55.9±0.3	59.6±0.1	>3 days	N/A	N/A	N/A
GK* (k=3) [Shervashidze et al., 2009]	81.4±1.7	55.7±0.5	71.4±0.31	62.5±0.3	N/A	N/A	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	N/A	N/A
WL kernel* [Shervashidze et al., 2011]	90.4±5.7	59.9±4.3	75.0±3.1	**86.0±1.8**	78.9±1.9	73.8±3.9	50.9±3.8
GNTK* [Du et al., 2019a]	90.0±8.5	**67.9±6.9**	75.6±4.2	**84.2±1.5**	**83.6±1.0**	**76.9±3.6**	**52.8±4.6**
DCNN [Atwood and Towsley, 2016]	N/A	N/A	61.3±1.6	56.6±1.0	52.1±0.7	49.1±1.4	33.5±1.4
DGCNN [Zhang et al., 2018]	85.8±1.8	58.6±2.5	75.5±0.9	74.4±0.5	73.8±0.5	70.0±0.9	47.8±0.9
IGN [Maron et al., 2019b]	83.9±13.0	58.5±6.9	**76.6±5.5**	74.3±2.7	78.3±2.5	72.0±5.5	48.7±3.4
GIN [Xu et al., 2019]	89.4±5.6	64.6±7.0	76.2±2.8	82.7±1.7	80.2±1.9	75.1±5.1	52.3±2.8
PPGNs [Maron et al., 2019a]	**90.6±8.7**	66.2±6.6	**77.2±4.7**	83.2±1.1	81.4±1.4	73.0±5.8	50.5±3.6
Natural GN [de Haan et al., 2020]	89.4±1.60	66.8±1.79	71.7±1.04	82.7±1.35	N/A	74.8±2.01	51.3±1.50
WEGL [Kolouri et al., 2021]	N/A	**67.5±7.7**	76.5±4.2	N/A	80.6±2.0	75.4±5.0	52.3±2.9
GIN+GraphNorm [Cai et al., 2021]	**91.6 ± 6.5**	64.9 ± 7.5	**77.4 ± 4.9**	82.7 ± 1.7	80.2 ± 1.0	76.0 ± 3.7	N/A
GSN-e	**90.6±7.5** 6 (cycles)	**68.2±7.2** 6 (cycles)	**76.6±5.0** 4 (cliques)	**83.5± 2.3** 15 (cycles)	**85.5±1.2** 3 (triangles)	**77.8±3.3** 5 (cliques)	**54.3±3.3** 5 (cliques)
GSN-v	**92.2±7.5** 12 (cycles)	67.4±5.7 10 (cycles)	74.59±5.0 4 (cliques)	**83.5±2.0** 3 (triangles)	**82.7±1.5** 3 (triangles)	**76.8±2.0** 4 (cliques)	**52.6±3.6** 3 (triangles)

Generalisation

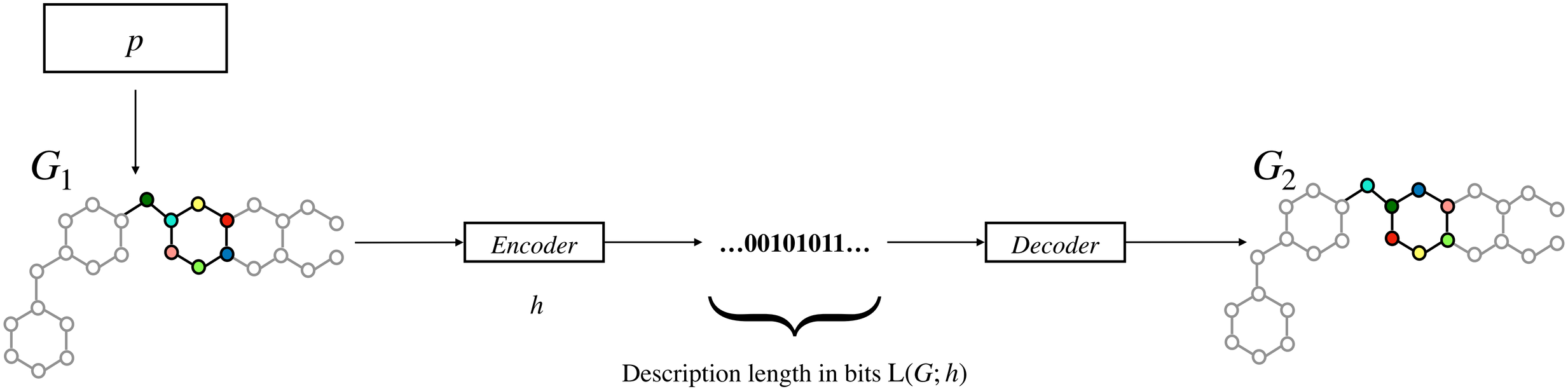


- Vertex identification $\Rightarrow$ Improved expressivity
- Vertex identification $\not\Rightarrow$ Improved generalisation

Slide credit: F. Frasca
(LOG tutorial on GNN expressivity)

Part II: Graph Compression

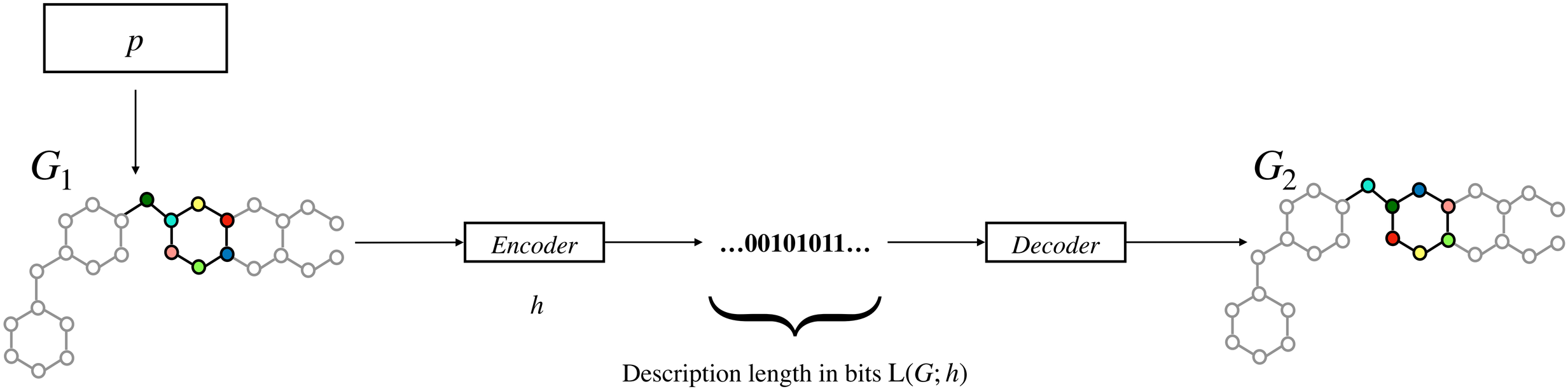
Lossless compression: unlabelled graphs



Condition: $h(G_1) = h(G_2) \Rightarrow G_1 \simeq G_2$ (GI-injective function*)

Objective: $\min_h \mathbb{E}_{G\sim p}[\mathrm{L}(G; h)]$

Equivalent problem

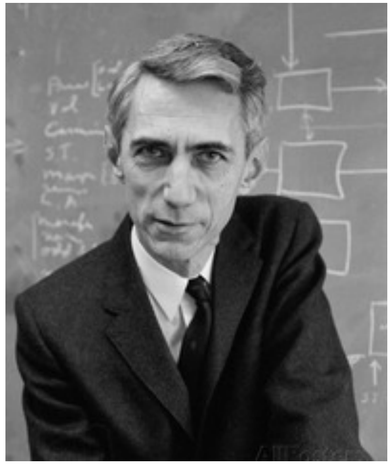


Optimise for: *(1) a* GI-injective *graph encoder* h_{enc} *and*

(2) a probability distribution q

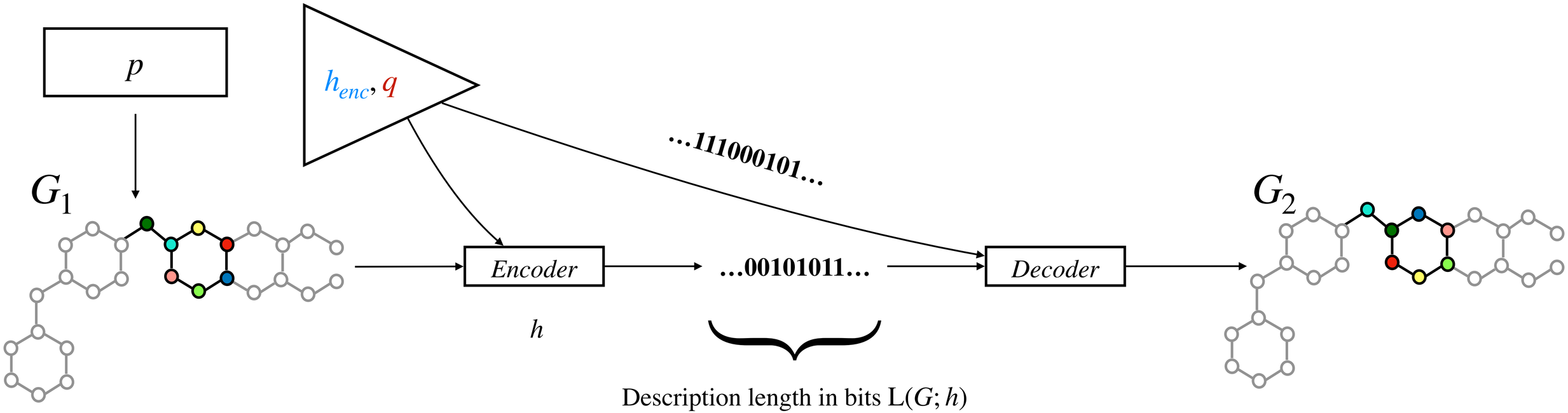
Objective: $\min_{h_{enc},\, q} \mathbb{E}_{G\sim p}[-\log q(h_{enc}(G))]$

Trust me here: *Every distribution* q *can be converted to an invertible code*



Shannon'1948

Equivalent problem (parametric models)

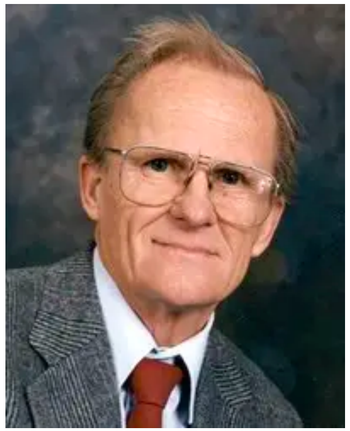


Encoder & decoder need to know the parameters of the model

Minimum Description Length

Objective: $\min_{h_{enc}, q} \mathbb{E}_{G\sim p}[-\log q(h_{enc}(G))] + \frac{1}{N}\mathrm{L}\left(h_{enc}, q\right)$

Data compression vs Model compression trade-off



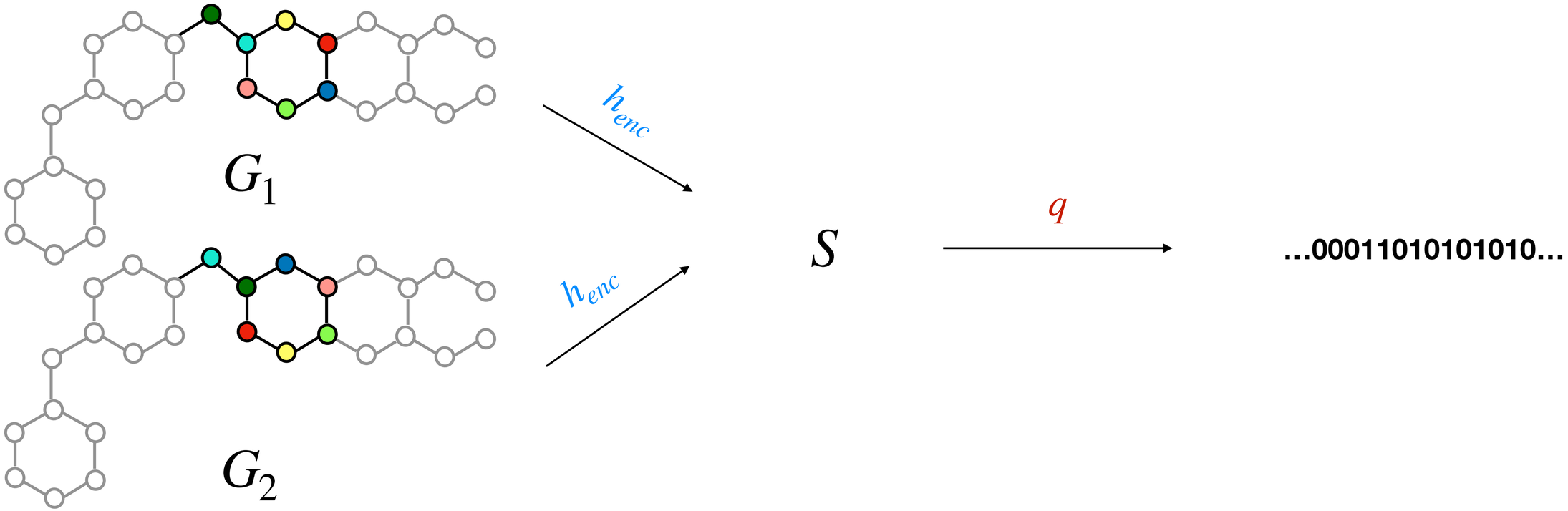
Risannen'1978

Challenge: Graph isomorphism (the usual suspect)

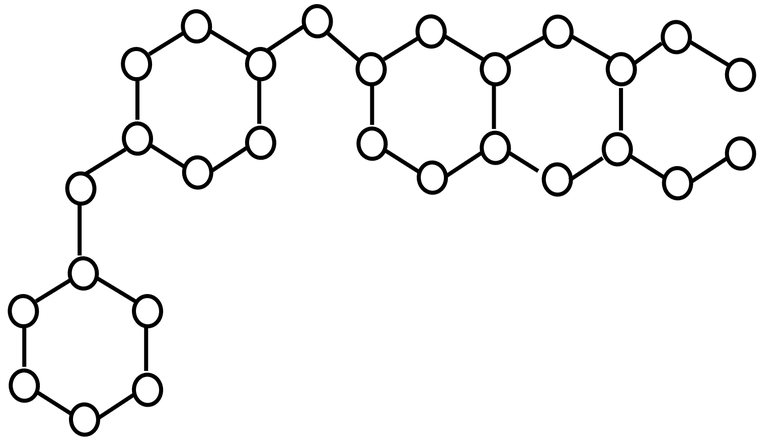
Theorem 3 (informal). *Optimally compressing unlabelled graphs requires* ***solving graph isomorphism****:*

$$\min_{h_{enc},\, q} \mathbb{E}_{G\sim p}[-\log q\big(h_{enc}(G)\big)] \Rightarrow \max_{h_{enc}} \mathbb{E}_{G\sim p}\Big[\Big|\big\{G' \in \mathfrak{G} \mid h_{enc}(G') = h_{enc}(G)\big\}\Big|\Big]$$

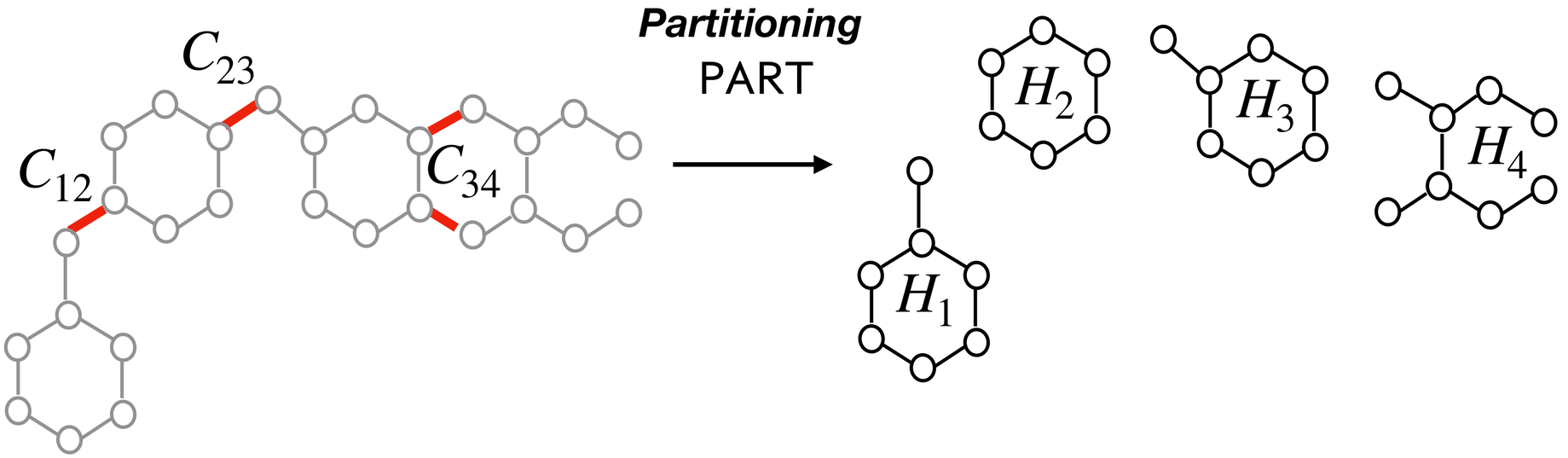
Compression quality vs computational complexity trade-off



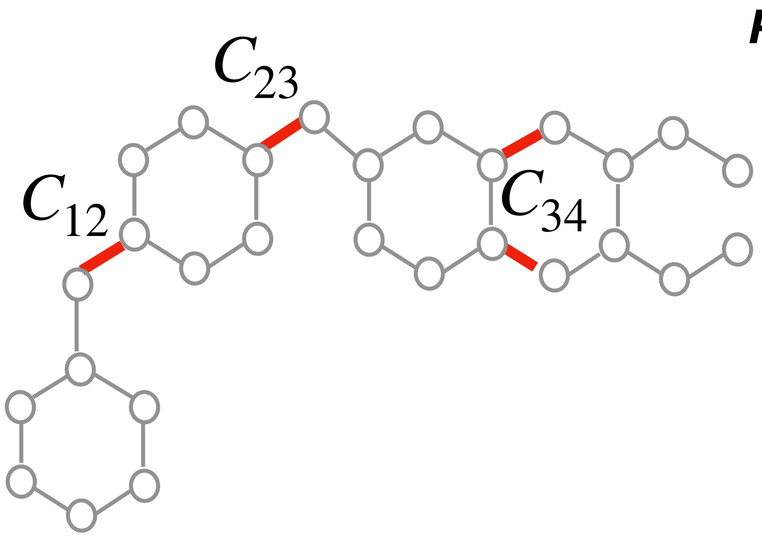
Partition and Code: Dictionary coding + subgraph isomorphism



Partition and Code: Dictionary coding + subgraph isomorphism

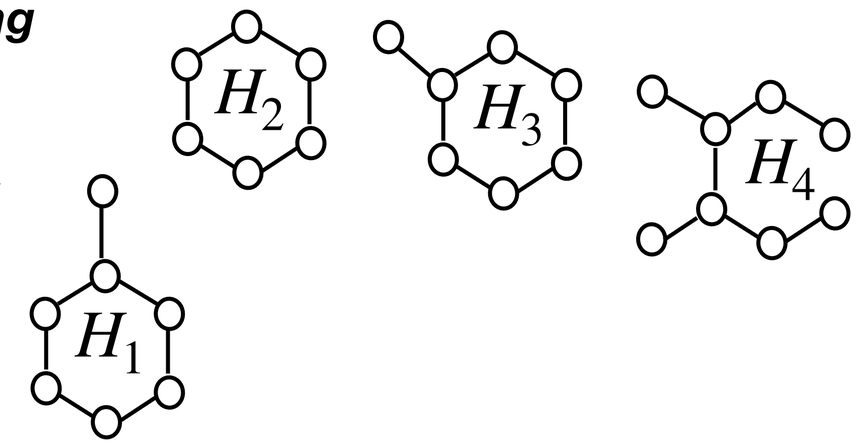


Partition and Code: Dictionary coding + subgraph isomorphism



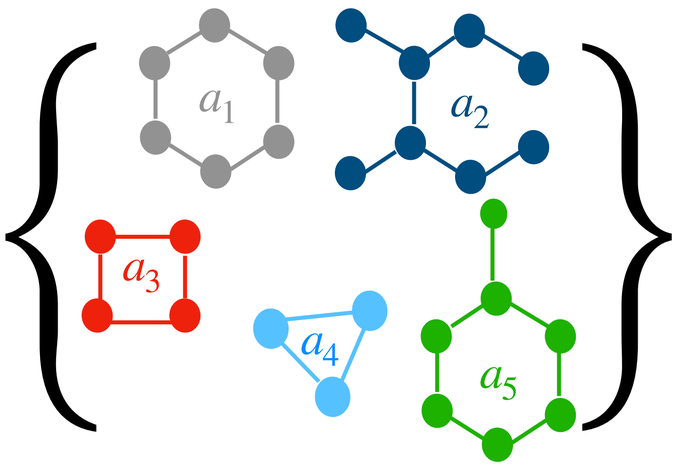
Partitioning

PART

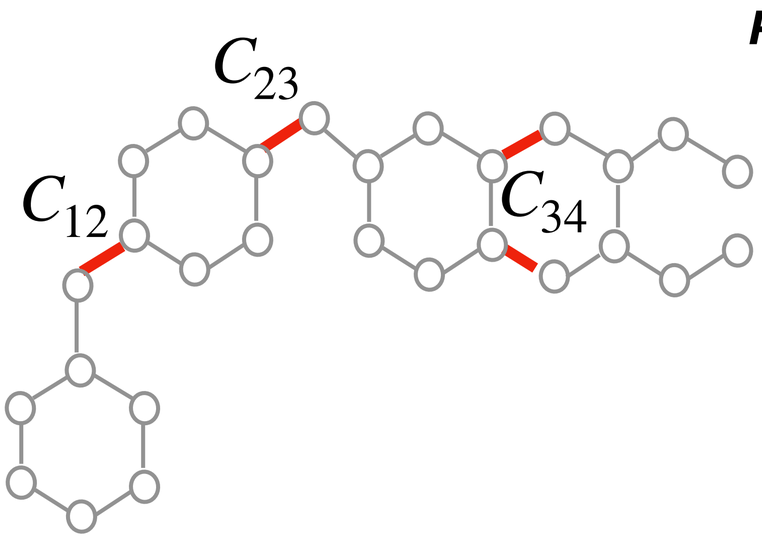


Dictionary

$\mathscr{D} = \{a_1, a_2, \ldots, a_{|\mathscr{D}|}\}$

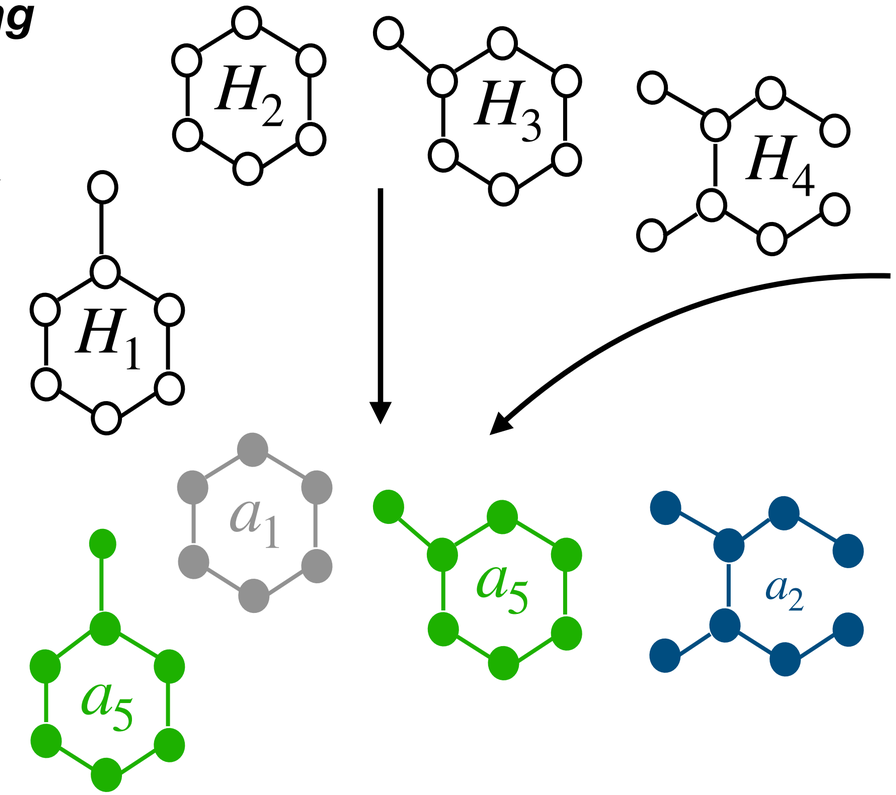


Partition and Code: Dictionary coding + subgraph isomorphism



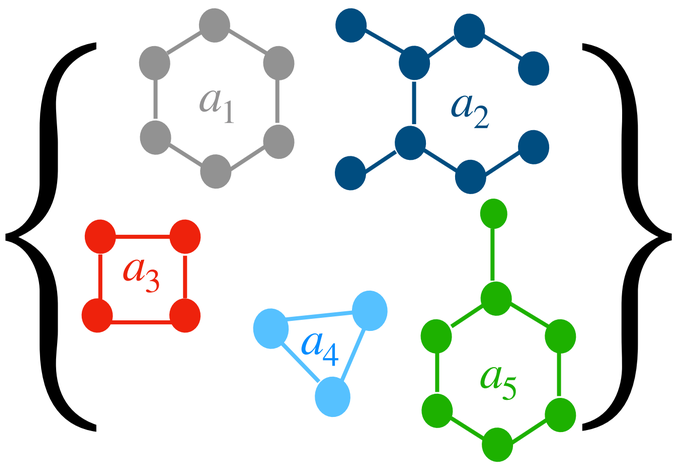
Partitioning

PART

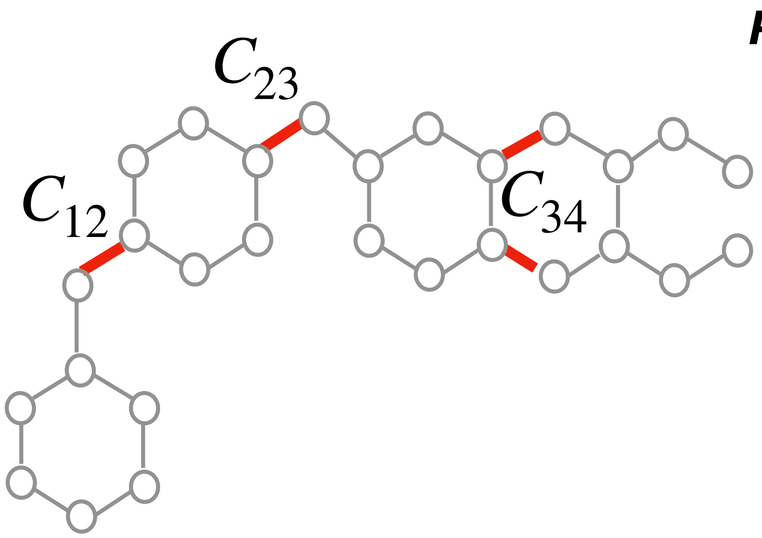


Dictionary

$\mathscr{D} = \{a_1, a_2, \ldots, a_{|\mathscr{D}|}\}$

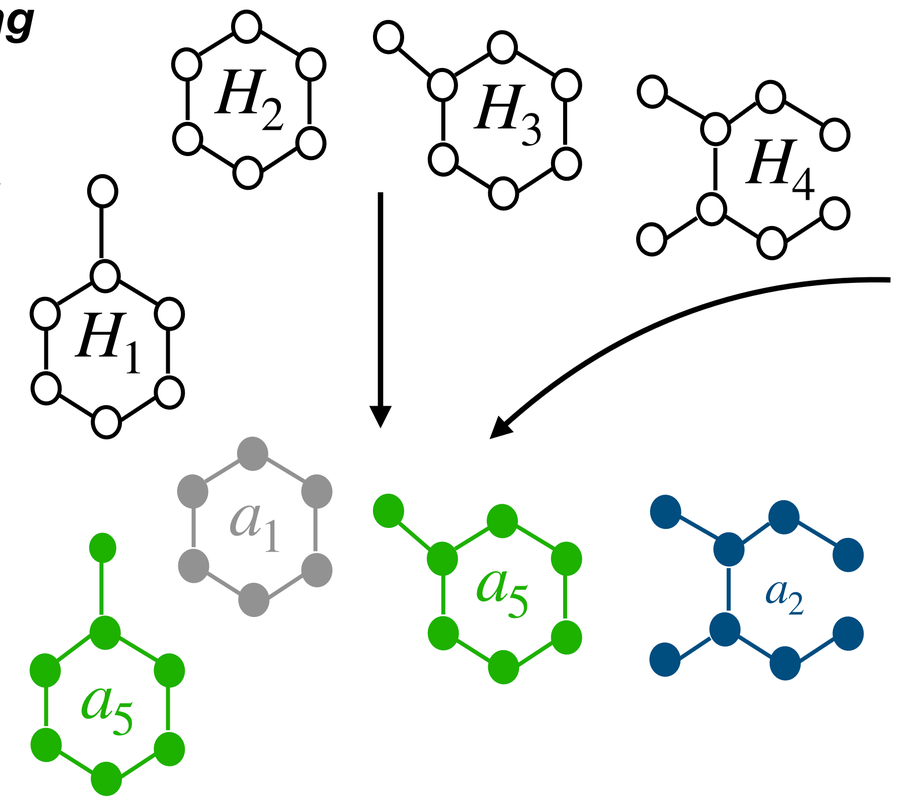


Partition and Code: Dictionary coding + subgraph isomorphism



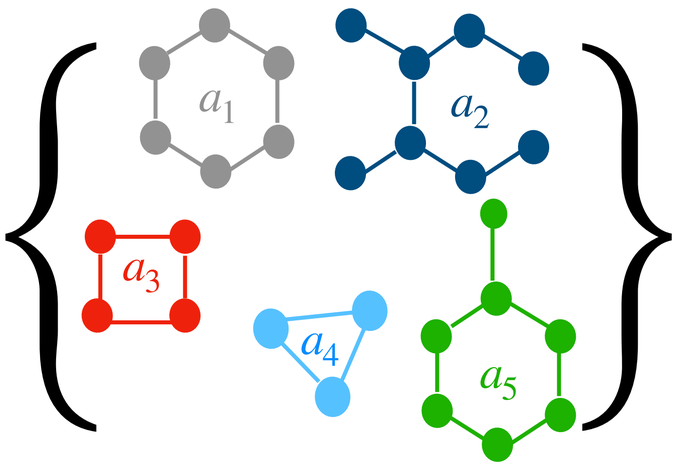
Partitioning

PART



Dictionary

$\mathscr{D} = \{a_1, a_2, \ldots, a_{|\mathscr{D}|}\}$



Graph encoding:
$h_{enc}(G) = (C, \mathscr{H})$

Cuts:
$C = (C_{12}, C_{13}, C_{14}, C_{23}, C_{24}, C_{34})$

Dictionary Subgraph indices:
$\mathscr{H} = \{5,1,5,2\}$

Graph distribution:
$-\log q\left(h_{enc}(G)\right)$

Cut distribution:
$-\log q(C \mid \mathscr{H})$
…00010011…

+

Subgraph distribution:
$-\log q(\mathscr{H})$
…01010001…

Model encoding:
$\mathrm{L}(h_{enc}, q) \approx L(\mathscr{D})$
…11000101…

Trade-offs

☑ **Computational complexity:** We efficiently solve GI for small graphs of size $k = O(1)$.

☑ **Compression quality :** Linear gains against PnC (and other compressors) without GI-testing

$*1 - \delta = \mathbb{E}_{G \sim p}[\frac{\#\text{ dictionary subgraphs}}{\#\text{ total subgraphs}}]$

Trade-offs

☑ **Computational complexity:** We efficiently solve GI for small graphs of size $k = O(1)$.

☑ **Compression quality :** Linear gains against PnC (and other compressors) without GI-testing

Theorem 4 (informal). *The gains of a PnC compressor that matches subgraphs by solving GI vs one that matches subgraphs via equality are:*

$$\mathbb{E}_{G\sim p}[\mathrm{L}_{PnC-S}(G)] \approx \mathbb{E}_{G\sim p}[\mathrm{L}_{PnC-G}(G)] - n(1-\delta)\log k^*$$

The optimal gains of a compressor that solves GI are:

$$\mathbb{E}_{G\sim p}[\mathrm{L}_{PnC-S}(G)] \approx \mathbb{E}_{G\sim p}[\mathrm{L}_{PnC-G}(G)] - \Theta(n\log n)$$

$^*1-\delta = \mathbb{E}_{G\sim p}\left[\frac{\#\text{ dictionary subgraphs}}{\#\text{ total subgraphs}}\right]$

Results (molecules)

Method type	Graph type	Small Molecules								
	Dataset name	MUTAG			PTC			ZINC		
		data	total	params	data	total	params	data	total	params
Null	Uniform (raw adjac.)	-	8.44	-	-	17.43	-	-	10.90	-
	Edge list	-	7.97	-	-	9.38	-	-	8.60	-
	Erdős-Renyi	-	4.78	-	-	5.67	-	-	5.15	-
PnC	PnC + SBM	3.81	4.11	49	4.38	4.79	155	3.34	3.45	594
	PnC + Louvain	2.20	**2.51**	47	2.65	**3.15**	166	1.96	**1.99**	196
	PnC + PropClust	2.42	**3.03**	63	3.38	**4.02**	178	2.20	**2.35**	726
	PnC + Neural Part.	2.17±0.02	**2.45±0.02**	46±1	2.63±0.26	**2.97±0.14**	143±31	2.01±0.02	**2.07±0.03**	384±105

Theorem 5 (informal)
$\Theta(n^2)$ *gains*

Results (molecules)

Method type	Graph type	Small Molecules								
	Dataset name	MUTAG			PTC			ZINC		
		data	total	params	data	total	params	data	total	params
Null	Uniform (raw adjac.)	-	8.44	-	-	17.43	-	-	10.90	-
	Edge list	-	7.97	-	-	9.38	-	-	8.60	-
	Erdős-Renyi	-	4.78	-	-	5.67	-	-	5.15	-
Partitioning (non-parametric)	SBM-Bayes	-	4.62	-	-	5.12	-	-	4.75	-
	Louvain	-	4.80	-	-	5.27	-	-	4.77	-
	PropClust	-	4.92	-	-	5.40	-	-	4.85	-
PnC	PnC + SBM	3.81	4.11	49	4.38	4.79	155	3.34	3.45	594
	PnC + Louvain	2.20	**2.51**	47	2.65	**3.15**	166	1.96	**1.99**	196
	PnC + PropClust	2.42	**3.03**	63	3.38	**4.02**	178	2.20	**2.35**	726
	PnC + Neural Part.	2.17±0.02	**2.45±0.02**	46±1	2.63±0.26	**2.97±0.14**	143±31	2.01±0.02	**2.07±0.03**	384±105

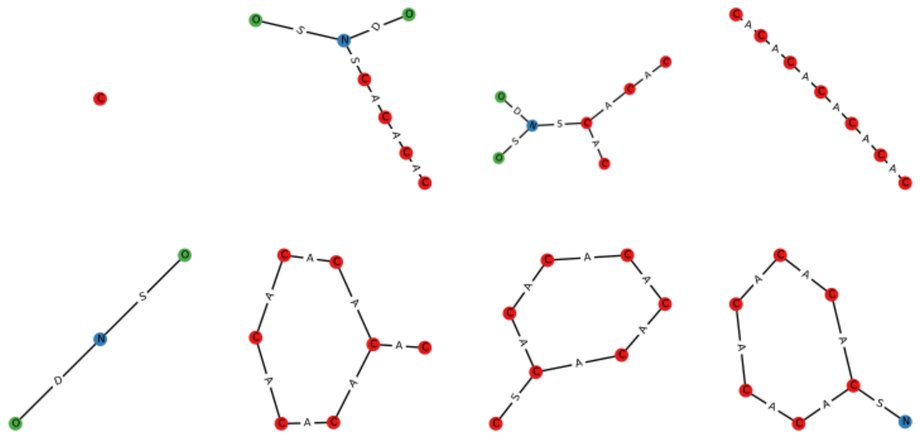
Theorem 5 (informal)
$\Theta(n)$ *gains*

Results (molecules)

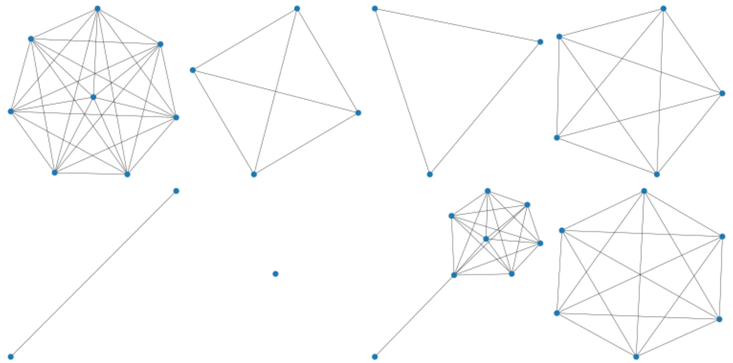
Method type	Graph type	Small Molecules								
	Dataset name	MUTAG			PTC			ZINC		
		data	total	params	data	total	params	data	total	params
Null	Uniform (raw adjac.)	-	8.44	-	-	17.43	-	-	10.90	-
	Edge list	-	7.97	-	-	9.38	-	-	8.60	-
	Erdős-Renyi	-	4.78	-	-	5.67	-	-	5.15	-
Partitioning (non-parametric)	SBM-Bayes	-	4.62	-	-	5.12	-	-	4.75	-
	Louvain	-	4.80	-	-	5.27	-	-	4.77	-
	PropClust	-	4.92	-	-	5.40	-	-	4.85	-
Neural (likelihood)	GraphRNN	1.33	3338.21	388K	1.57	1394.59	389K	1.62	43,16	388K
	GRAN	0.81	12557.75	1460K	2.18	5269.82	1470K	1.30	157.7	1461K
	GraphRNN (pruned)	1.95	12.39	1.08K	2.16	6.71	1.10K	1.79	**2.02**	1.90K
	GRAN (pruned)	2.59	24.56	2.23K	4.31	14.00	2.36K	3.26	3.47	1.69K
PnC	PnC + SBM	3.81	4.11	49	4.38	4.79	155	3.34	3.45	594
	PnC + Louvain	2.20	**2.51**	47	2.65	**3.15**	166	1.96	**1.99**	196
	PnC + PropClust	2.42	**3.03**	63	3.38	**4.02**	178	2.20	**2.35**	726
	PnC + Neural Part.	2.17±0.02	**2.45±0.02**	46±1	2.63±0.26	**2.97±0.14**	143±31	2.01±0.02	**2.07±0.03**	384±105

Deep generative models suffer from overparameterisation.

Learnt dictionaries (explainability)



Molecules



Social Networks

Take home messages

- Graph Isomorphism affects graph ML problems in multiple ways.
- GI creates inevitable trade-offs (expressivity vs comp. complexity or compression quality vs comp. complexity).
- Graph substructures offer a way to strike a balance.
- Additional advantages: generalisation + explainability.

Giorgos Bouritsas

Fabrizio Frasca

Stefanos Zafeiriou

Michael Bronstein

Nikos Karalias

Andreas Loukas

Imperial College London

EPFL