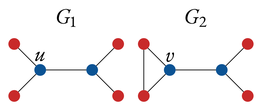
Structure-Aware Transformer for Graph Representation Learning

Dexiong Chen[†,1,2], Leslie O'Bray[†,1,2] and Karsten Borgwardt[1,2]

[†]: These authors contributed equally. [1]: Department of Biosystems Science and Engineering, ETH Zürich, Switzerland. [2]: SIB Swiss Institute of Bioinformatics, Switzerland.

Motivation and contribution

- GNNs struggle from problems of under-reaching, over-smoothing, and over-squashing.
- Transformers can address those issues, but have had limited performance to date on graphs because the existing self-attention calculation uses *attributed similarity* between nodes, as opposed to *structural similarity*.
- Information about the graph is typically only incorporated via a positional encoding.
- Nodes can have the same position while being structurally different (see nodes u and v).



Structure-Aware Transformer (SAT)

- We extend self-™attention to account for local structures by extracting a *subgraph representation* rooted at each node.
- We propose several methods for automatically generating *subgraph representations*.
- The resulting framework can leverage *any GNN* to extract subgraph representations and empirically outperforms the base GNN ⇒ SAT is an effortless enhancer of any GNN.
- The representations from the structure-aware attention are *at least as expressive as* the representations from the structure extractor.
- We provide a bound on the distance between the structure-aware representations.

A structure-aware self attention

The original self-attention calculation is:

$$\mathrm{Attn}(\mathbf{X}) := \mathrm{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{out}}})\mathbf{V} \in^{n\times d_{out}},$$

where $\mathbf{Q} = \mathbf{X}\mathbf{W}_\mathbf{Q}$, $\mathbf{K} = \mathbf{X}\mathbf{W}_\mathbf{K}$ and $\mathbf{V} = \mathbf{X}\mathbf{W}_\mathbf{V}$.

We can rewrite this as a kernel smoother:

$$\mathrm{Attn}(x_v) = \sum_{u\in V} \frac{\kappa_{\exp}(x_v, x_u)}{\sum_{w\in V}\kappa_{\exp}(x_v, x_w)} f(x_u),\ \forall v \in V,$$

where

$$\kappa_{\exp}(x, x') := \exp\left(\langle \mathbf{W}_\mathbf{Q}x, \mathbf{W}_\mathbf{K}x'\rangle/\sqrt{d_{out}}\right) f(x) = \mathbf{W}_\mathbf{V}x \in \mathbb{R}^{d_{out}}$$

We define our **structure-aware attention** as:

$$\text{SA-attn}(v) := \sum_{u\in V} \frac{\kappa_{\text{graph}}(S_G(v), S_G(u))}{\sum_{w\in V}\kappa_{\text{graph}}(S_G(v), S_G(w))} f(x_u),$$

where

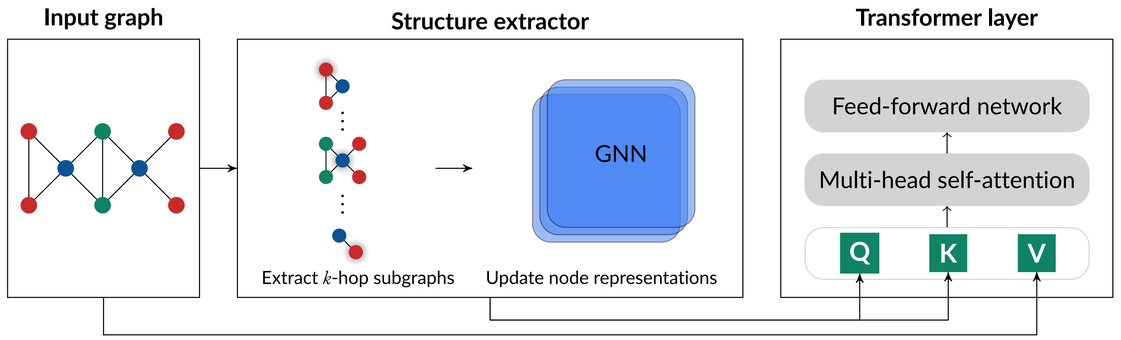
$$\kappa_{\text{graph}}(S_G(v), S_G(u)) = \kappa_{\exp}(\varphi(v, G), \varphi(u, G)),$$

and $\varphi(v, G)$ is a *structure extractor* that extracts vector representations of some subgraph centered at u with node features $\mathbf{X}$, e.g. k-subtree SAT and k-subgraph SAT:

$$k\text{-subtree SAT}: \varphi(u, G) = \mathrm{GNN}^{(k)}_G(u)$$

$$k\text{-subgraph SAT}: \varphi(u, G) = \sum_{v\in\mathcal{N}^k_u(u)} \mathrm{GNN}^{(k)}_G(v)$$

An instance of a Structure-Aware Transformer: k-subgraph SAT

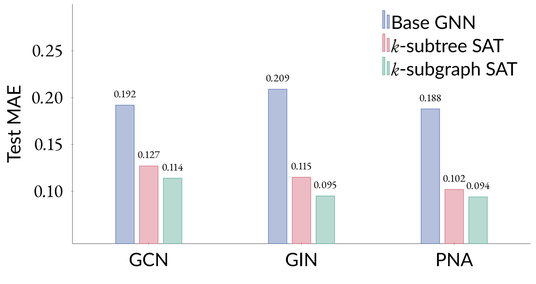


Comparison to state-of-the-art GNNs and graph Transformers

	ZINC ↓	CLUSTER ↑	PATTERN ↑
# graphs	12,000	12,000	14,000
Avg. # nodes	23.2	117.2	118.9
Avg. # edges	49.8	4,303.9	6,098.9
Metric	MAE	Accuracy	Accuracy
GIN	0.387±0.015	64.716±1.553	85.590±0.011
GAT	0.384±0.007	70.587±0.447	78.271±0.186
PNA	0.188±0.004	67.077±0.977	86.567±0.075
Transformer+RWPE	0.310±0.005	29.622±0.176	86.183±0.019
Graph Transformer	0.226±0.014	73.169±0.622	84.808±0.068
SAN	0.139±0.006	76.691±0.650	86.581±0.037
Graphormer	0.122±0.006	–	–
k-subtree SAT	0.102±0.005	77.751±0.121	**86.865±0.043**
k-subgraph SAT	**0.094±0.008**	**77.856±0.104**	86.848±0.037

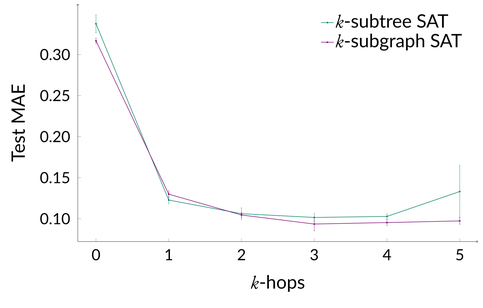
	OGBG-PPA ↑	OGBG-CODE2 ↑
# graphs	158,100	452,741
Avg. # nodes	243.4	125.2
Avg. # edges	2,266.1	124.2
Metric	Accuracy	F1 score
GCN	0.6839±0.0084	0.1507±0.0018
GCN-Virtual Node	0.6857±0.0061	0.1595±0.0018
GIN	0.6892±0.0100	0.1495±0.0023
GIN-Virtual Node	0.7037±0.0107	0.1581±0.0026
DeeperGCN	0.7712±0.0071	–
ExpC	**0.7976±0.0072**	–
Transformer	0.6454±0.0033	0.1670±0.0015
GraphTrans	–	0.1830±0.0024
k-subtree SAT	0.7522±0.0056	**0.1937±0.0028**

SAT enhances sparse GNNs (ZINC ↓)



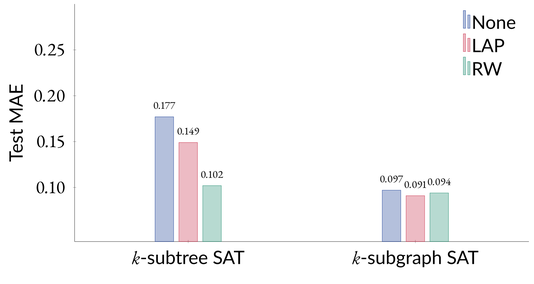
- SAT uses a GNN (i.e. a "base GNN") to create structure-aware node representations in the graph Transformer.
- Empirically, SAT *always* improves upon the performance of the base GNN it uses.

Effect of k in SAT (ZINC ↓)



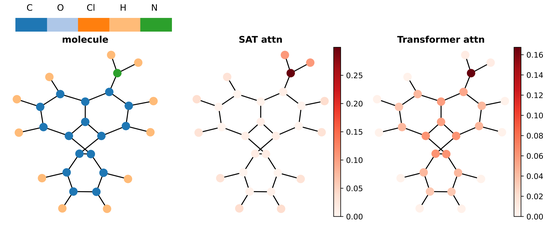
- The effect when considering different values of k (which determines the k-hop subgraph).
- $k = 0$ is equivalent to a vanilla Transformer. $k \geq 1$ incorporates structures into the node embeddings.

Effect of absolute encoding (ZINC ↓)



- Adding an absolute encoding improves performance.
- The performance gain from including an absolute encoding is much smaller compared to the performance gain from using the structure-aware node embeddings.

Interpretability of SAT



- When molecules were screened for mutagenicity, SAT was better able to identify motifs that are known to be mutagenic than a vanilla Transformer with RWPE.
- SAT puts more attention weights on the known mutagenic motifs.

ETH zürich

D BSSE

MLCB

SIB Swiss Institute of Bioinformatics