Subgraphormer: Subgraph GNNs meet Graph Transformers

Guy Bar-Shalom[1], Beatrice Bevilacqua[2], Haggai Maron[1,3]

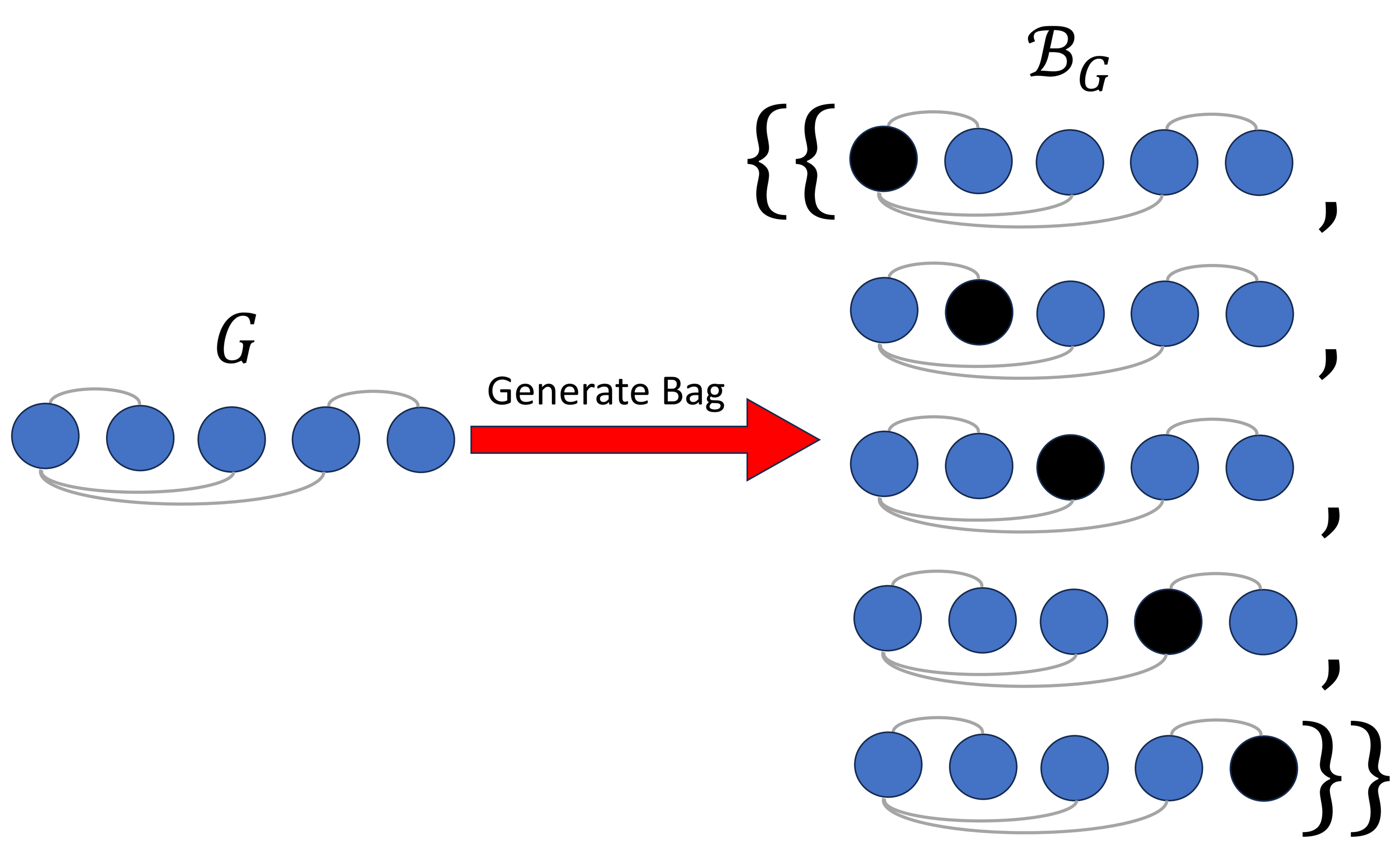
Technion[1], Purdue University[2], NVIDIA Research[3]

Introduction

This paper merges two enhanced Graph Neural Network (GNN) architectures: Subgraph GNNs and Graph Transformers. Subgraph GNNs apply GNNs to bags of subgraphs generated from the original graph, which is provably more powerful than traditional message-passing, while Graph Transformers leverage attention mechanisms, on new objects, which are derived from the original graph and preserves important quantities. We propose a novel architecture, called `Subgraphormer` that combines these two approaches, offering improved performance for graph data – with promising results on the ZINC12k dataset.

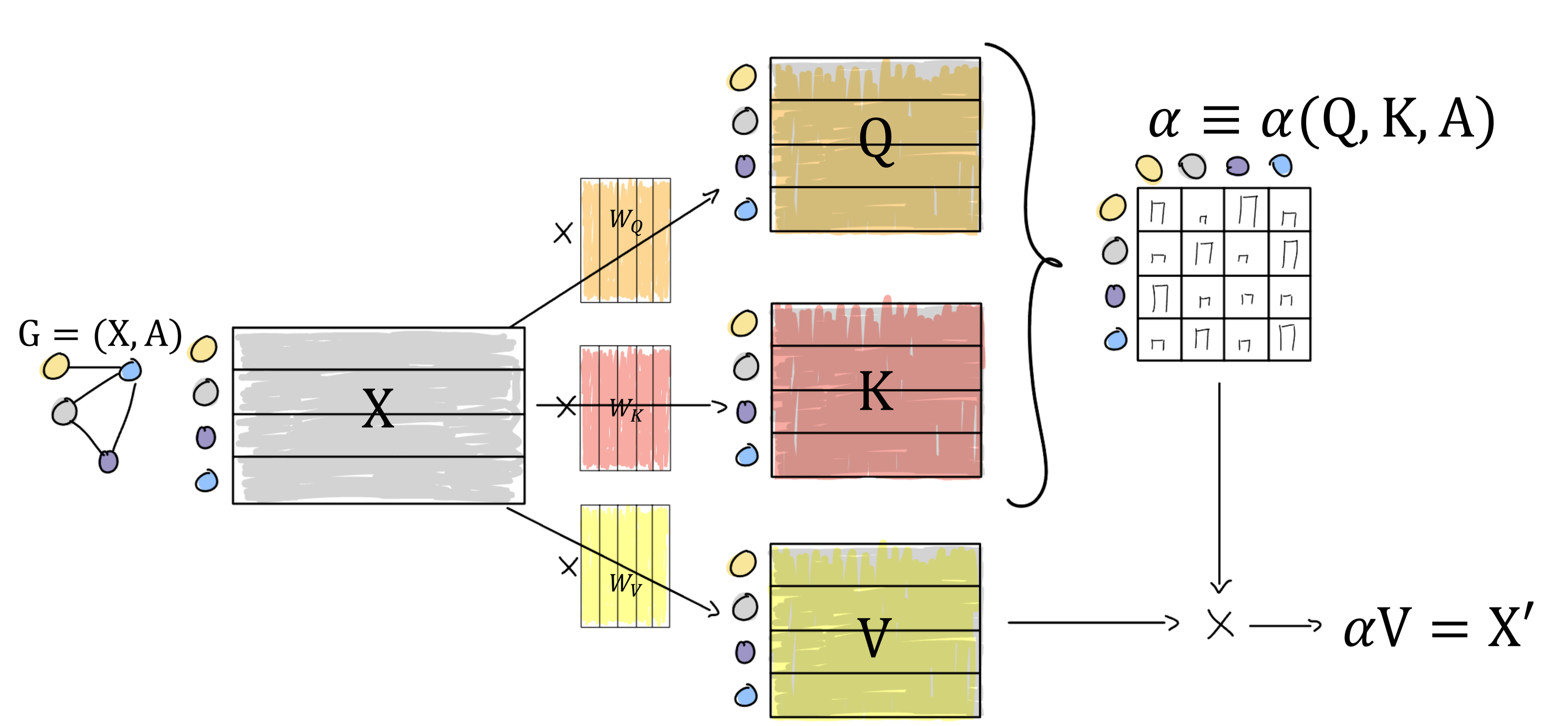
Notation and preliminaries

Subgraph GNNs. Subgraph GNNs represent a graph, G as a multiset or a bag of subgraphs, denoted as $\mathcal{B}_G$. The bag of subgraphs is generated through the application of a predefined selection policy to the original graph. The following is a specific example wherein the predefined policy involves marking the root node of each subgraph.



We denote be x_v^s the feature of node v, in subgraph, s.

Graph Transformers. Graph Transformers are designed to leverage the significant success of the Transformer model, which was originally developed for natural language processing tasks. The core concept is to implement attention-based operations among nodes, enhancing their capability in graph-based applications.



Contributions

1. A graph Transformer model, which builds on insights from Subgraph GNNs, dubbed `Subgraphormer`.
2. A positional and structural encoding scheme tailored to subgraphs, enabling each node to integrate information from multiple subgraphs.

The `Subgraphormer` Model

Overview.

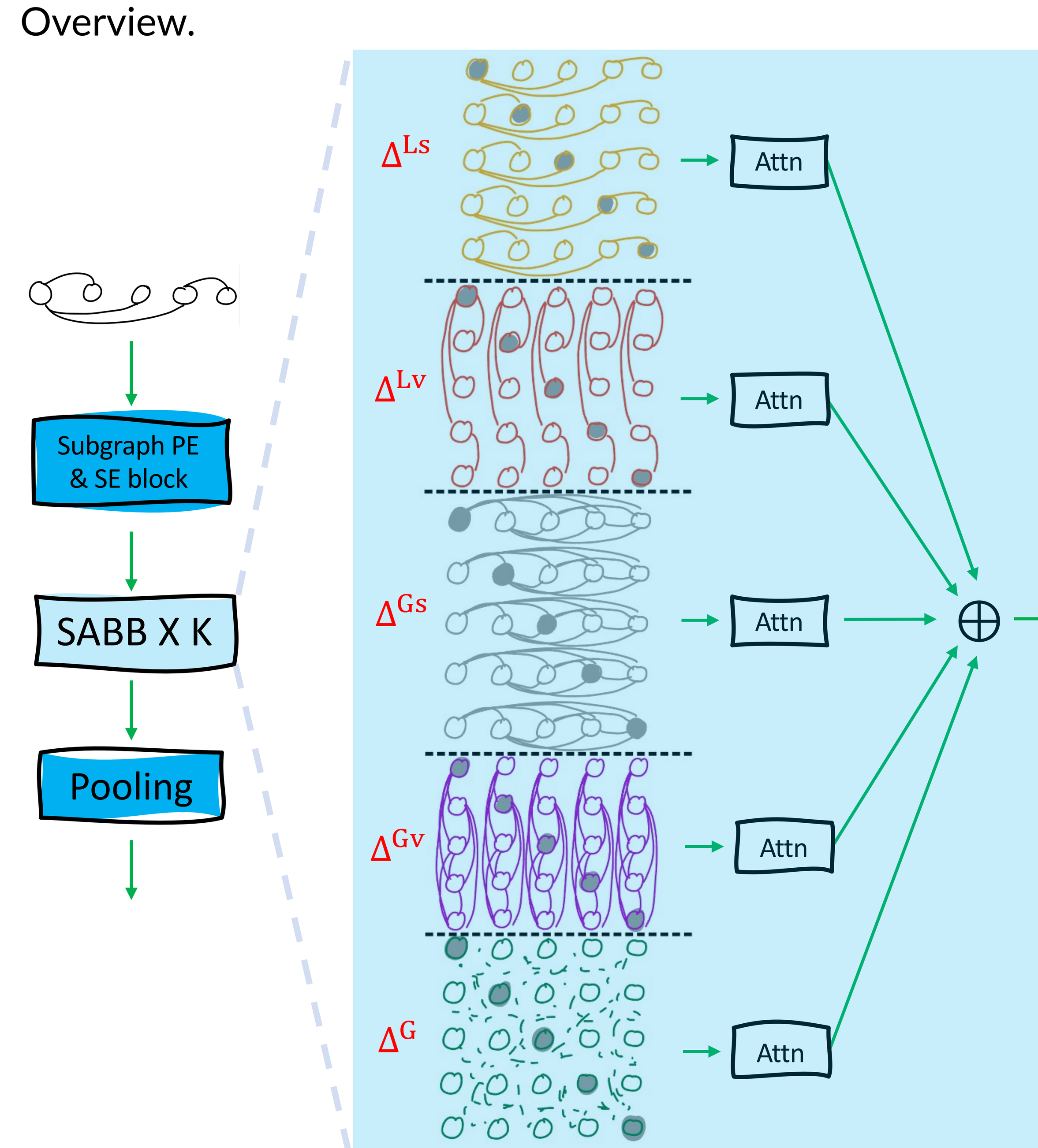


Figure 1. Our architecture, `Subgraphormer`, is composed of a structural and positional encoding layer, a stacking of Subgraph Attention-Based Blocks (SABBs), and a pooling layer.

Subgraph PE & SE block.

1. *Node-Marking.* We add a special mark to each node as follows,
$$x_v^{s;\mathrm{NM}} \leftarrow x_v^s + z_{\mathrm{dist}(s,v)},$$
where $z \in \mathbf{R}^d$ is a learnable embedding indexed by the value of the shortest path from s to v in the original Graph.
2. *Positional Encoding.* Based on the original graph's Laplacian eigendecomposition: $L = D - A = U^T \Lambda U$, define $\mathsf{p}_i \triangleq [U_{i1}, \ldots, U_{ik}]$, we have,
$$x_v^{s;\mathrm{PE}} \leftarrow \mathbf{W}_1^{\mathrm{PE}} \mathrm{LeakyReLU}\big(\mathbf{W}_2^{\mathrm{PE}}[\mathsf{p}_s \oplus \mathsf{p}_v]\big).$$
3. *Structural Encoding.* Based on the original graph's Random Walk operator, $\mathsf{RW} \triangleq D^{-1}A$, define, $\mathsf{r}_i \triangleq [\mathsf{RW}_{ii}, \mathsf{RW}_{ii}^2, \ldots, \mathsf{RW}_{ii}^k]$, we have,
$$x_v^{s;\mathrm{SE}} \leftarrow \mathbf{W}_1^{\mathrm{SE}} \mathrm{LeakyReLU}\left(\mathbf{W}_2^{\mathrm{SE}}[\mathsf{r}_s \oplus \mathsf{r}_v]\right).$$

The three vectors $x_v^{s;\mathrm{NM}}, x_v^{s;\mathrm{PE}}, x_v^{s;\mathrm{SE}}$ are then concatenated and passed through an `MLP` with one hidden layer, along with a residual connection with $x_v^{s;\mathrm{NM}}$.

Subgraph Attention-Based Block. We utilize the GATV2 [2] type layer to calculate the attention matrix α. Our approach particularly focuses on biasing this attention towards the structures of each of the subgraphs, as illustrated in Figure 1. Specifically, the related adjacencies are defined as follows:

$$\text{Local Subg.-to-Subg. Attention: } \Delta^{\mathrm{Ls}}\Big((s,v),(s',v'),G\Big) = \begin{cases} \delta_{ss'} & \text{if } v \text{ and } v' \text{ are neighbors in } G \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Local Node-to-Node Attention: } \Delta^{\mathrm{Lv}}\Big((s,v),(s',v'),G\Big) = \begin{cases} \delta_{vv'} & \text{if } s \text{ and } s' \text{ are neighbors in } G \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Global Same Subgraph Attention: } \Delta^{\mathrm{Gs}}\Big((s,v),(s',v'),G\Big) = \delta_{ss'}$$

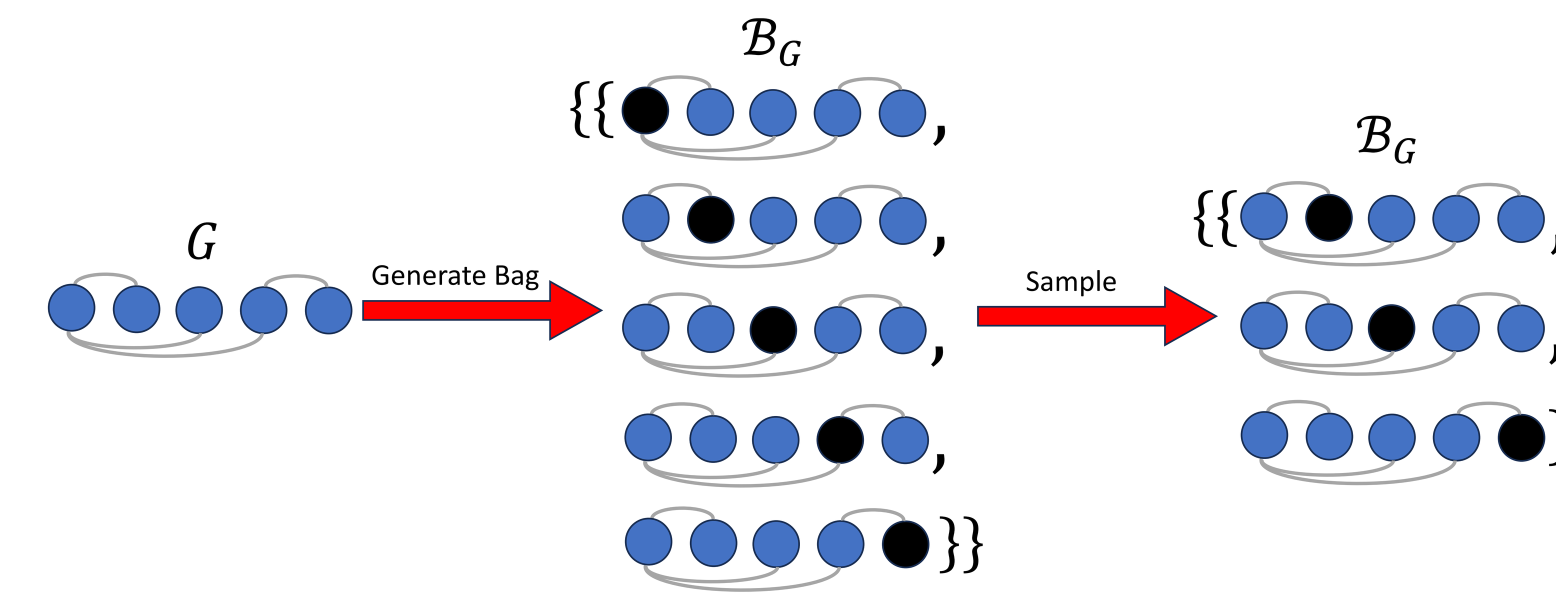
$$\text{Global Same Node Attention: } \Delta^{\mathrm{Gv}}\Big((s,v),(s',v'),G\Big) = \delta_{vv'}$$

$$\text{Global Attention: } \Delta^{\mathrm{G}}\Big((s,v),(s',v'),G\Big) = 1$$

Pooling. The final pooling layer, ρ is implemented as follows, $\rho(B_G) = \frac{1}{S}\sum_{s=1}^{S} \mathtt{MLP}\Big(\sum_{v=1}^{N} x_v^s\Big)$.

Stochastic sampling

To improve our model's scalability, we implement stochastic sampling by randomly selecting subgraphs (specifically we use $\{0.05\%, 0.2\%, 0.5\%\}$) from the bag $\mathcal{B}_G$. Our attention mechanism is adapted to disregard unsampled subgraphs, nullifying edges from (or to) nodes in unselected subgraphs.



Experiments

In Table 1 we benchmark `Subgraphormer` against Transformer-based approaches, and Subgraph-based approaches, as well as other baselines. In Table 2 we demonstrate the performance of our stochastic sampling approach.

(1) Table 1 clearly demonstrates that all variants of `Subgraphormer` improve over all baselines.

(2) Table 2 shows that `Subgraphormer` + SE and `Subgraphormer` + SE + PE variants consistently outperform ESAN [1] across all sampling percentages.

Model	ZINC (Test MAE ↓)
GSN	0.101 ± 0.010
CIN (small)	0.094 ± 0.004
GIN	0.252 ± 0.017
SAN	0.139 ± 0.006
URPE	0.086 ± 0.007
GPS	0.070 ± 0.004
Graphormer	0.122 ± 0.006
Graphormer-GD	0.081 ± 0.009
K-Subgraph SAT	0.094 ± 0.008
NGNN	0.111 ± 0.003
SUN	0.083 ± 0.003
ESAN	0.102 ± 0.003
OSAN	0.154 ± 0.008
GNN-AK	0.105 ± 0.010
GNN-AK+	0.091 ± 0.002
GNN-SSWL	0.082 ± 0.003
GNN-SSWL+	0.070 ± 0.005
`Subgraphormer`	0.064 ± 0.001
`Subgraphormer` + SE	0.066 ± 0.003
`Subgraphormer` + PE	$\mathbf{0.062 \pm 0.002}$
`Subgraphormer` + SE + PE	0.067 ± 0.002

Table 1

Model		ZINC (Test MAE ↓)
ESAN	(100%)	0.102 ± 0.003
ESAN	(50%)	0.155 ± 0.007
ESAN	(20%)	0.166 ± 0.005
ESAN	(5%)	0.179 ± 0.001
`Subgraphormer`	(100%)	0.064 ± 0.001
`Subgraphormer`	(50%)	$\mathbf{0.079 \pm 0.050}$
`Subgraphormer`	(20%)	0.129 ± 0.010
`Subgraphormer`	(5%)	0.217 ± 0.008
`Subgraphormer` + SE	(100%)	0.065 ± 0.002
`Subgraphormer` + SE	(50%)	0.081 ± 0.005
`Subgraphormer` + SE	(20%)	0.121 ± 0.014
`Subgraphormer` + SE	(5%)	$\mathbf{0.143 \pm 0.001}$
`Subgraphormer` + PE	(100%)	$\mathbf{0.062 \pm 0.002}$
`Subgraphormer` + PE	(50%)	0.082 ± 0.005
`Subgraphormer` + PE	(20%)	0.130 ± 0.003
`Subgraphormer` + PE	(5%)	0.227 ± 0.012
`Subgraphormer` + SE + PE	(100%)	0.067 ± 0.002
`Subgraphormer` + SE + PE	(50%)	0.081 ± 0.006
`Subgraphormer` + SE + PE	(20%)	$\mathbf{0.114 \pm 0.005}$
`Subgraphormer` + SE + PE	(5%)	0.164 ± 0.005

Table 2

Link to our paper and References

References.

[1] Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael M Bronstein, and Haggai Maron. Equivariant subgraph aggregation networks. *arXiv preprint arXiv:2110.02910*, 2021.

[2] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

Figure 2. Paper.

Email: *guy.b@campus.technion.ac.il*