Simplifying and Empowering Transformers for Large-Graph Representations

Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie,
Haitian Jiang, Yatao Bian, Junchi Yan

Codes: https://github.com/qitianwu/SGFormer
Pitfalls of Graph Neural Networks

- The designs of mainstream GNNs:
  - Locally aggregate neighbored nodes' features in each layer
  - Use neighbored nodes' embs for informative representation

- Common scenarios GNNs show deficient capability:

  - Hard to capture long-range dependence [Dai et al., 2018]
  - Distant signals are overly squashed [Alon et al., 2021]
  - Dissimilar linked nodes propagate wrong signals [Zhu et al., 2020]
  - Fail to distinguish two similar inputs [Xu et al., 2019]

- Long-range reasoning
- Over-squashing
- Heterophily
- Expressivity
GNNs require observed graphs as input:

- **Solution**: Pre-define a graph by some rules (e.g., k nearest neighbors)
- **Limitation**: the pre-defined graph is independent of downstream tasks
Message Passing Beyond Input Graphs

Graph Neural Networks
- local message passing
  - defined over fixed input topology

Transformers
- all-pair message passing
  - on layer-specific latent graphs

Mathematical representation:
- \( x \times \text{adjacency matrix} \times \text{node embs} = \text{next-layer node embs} \)
- \( x \times \text{attention matrix} \times \text{node embs} = \text{next-layer node embs} \)

Q1: computational bottleneck \( O(N^2) \)
Q2: how to incorporate graph inductive bias
Each node is an instance with a label
Train/test on a dataset of nodes in a graph
The graph size can be arbitrarily large

Notations for each node
- $x_u$ node (input) feature
- $y_u$ node ground-truth label
- $\hat{y}_u$ node predicted label
- $z_u^{(l)}$ node embedding at the $l$-th layer

Notations for the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
- $N = |\mathcal{V}|$ node number
- $E = |\mathcal{E}|$ edge number
- $X = [x_u]_{u=1}^N$ node feature matrix
- $Y = [y_u]_{u=1}^N$ label vector/matrix
- $A = [a_{uv}]_{u,v \in \mathcal{V}}$ adjacency matrix
- $Z^{(l)} = [z_u^{(l)}]_{u=1}^N$ node embedding matrix
NodeFormer: All-Pair Attention with $O(N)$

Kernelized softmax message passing

$$z_u^{(l+1)} = \sum_{v=1}^{N} \frac{\exp(q_u^T k_v)}{\sum_{w=1}^{N} \exp(q_u^T k_w)} \cdot v_v$$

where $q_u = W_Q z_u^{(l)}$, $k_u = W_K z_u^{(l)}$, $v_u = W_V z_u^{(l)}$

$$z_u^{(l+1)} = \sum_{v=1}^{N} \frac{\kappa(q_u, k_v)}{\sum_{w=1}^{N} \kappa(q_u, k_w)} \cdot v_v$$

where $\kappa(\cdot, \cdot): \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is a positive-definite kernel

$\kappa(a, b) = \langle \Phi(a), \Phi(b) \rangle_{\mathcal{V}} \approx \phi(a)^\top \phi(b)$

$\phi(\cdot): \mathbb{R}^d \rightarrow \mathbb{R}^m$ is a random feature map

Two summation are shared by all nodes (independent of $u$)

Only compute once

Computation complexity $O(N) + N \cdot O(1) = O(N)$

Qitian Wu et al., NodeFormer: A Scalable Graph Structure Learning Transformer for Node Classification, NeurIPS 2022
Qitian Wu et al., DIFFormer: Scalable (Graph) Transformers Induced by Energy Constrained Diffusion, ICLR 2023

Qitian Wu et al.

SGFormer: Simplifying Graph Transformers

\[
\hat{S}^{(k)}_{ij} = \frac{f\left(\|z_i^{(k)} - z_j^{(k)}\|_2^2\right)}{\sum_{l=1}^{N} f\left(\|z_i^{(k)} - z_l^{(k)}\|_2^2\right)}, \quad 1 \leq i, j \leq N
\]

\[
z_i^{(k+1)} = \left(1 - \tau \sum_{j=1}^{N} \hat{S}^{(k)}_{ij}\right) z_i^{(k)} + \tau \sum_{j=1}^{N} \hat{S}^{(k)}_{ij} z_j^{(k)}, \quad 1 \leq i \leq N
\]

Global attention inspired by diffusivity function

\[
\omega_{ij}^{(k)} = f\left(\|\tilde{z}_i^{(k)} - \tilde{z}_j^{(k)}\|_2^2\right) = 1 + \left(\frac{z_i^{(k)}}{\|z_i^{(k)}\|_2}\right)^\top \left(\frac{z_j^{(k)}}{\|z_j^{(k)}\|_2}\right)
\]

\[
\sum_{j=1}^{N} S^{(k)}_{ij} z_j^{(k)} = \sum_{j=1}^{N} \frac{1 + (\tilde{z}_i^{(k)})^\top z_j^{(k)}}{\sum_{l=1}^{N} (1 + (\tilde{z}_i^{(k)})^\top \tilde{z}_l^{(k)})} z_j^{(k)}
\]

\[
= \sum_{j=1}^{N} z_j^{(k)} + \frac{(\sum_{j=1}^{N} \tilde{z}_j^{(k)} (z_j^{(k)})^\top) \cdot \tilde{z}_i^{(k)}}{N + (\tilde{z}_i^{(k)})^\top \sum_{l=1}^{N} \tilde{z}_l^{(k)}}
\]

\[O(N)\]
Do We Really Need Deep Attention Layers?

Prior Art

Our Model

Qitian Wu et al., Simplifying and Empowering Transformers on Large-Graph Representations, NeurIPS 2023
Consider the $k$-th attention layer in Transformers:

$$z_u^{(k)} = (1 - \tau)z_u^{(k-1)} + \tau \sum_{v=1}^N c_{uv}^{(k)} z_v^{(k-1)}$$

residual link \hspace{1cm} attention weight

**Theorem 1 (Transformers as Graph Signal Denoising)**

For any given attention matrix $C^{(k)} = [c_{uv}^{(k)}]_{N \times N}$, the $k$-th attention layer is equivalent to a gradient descent operation with step size $\tau/2\lambda$ for an optimization problem with the cost function

$$\min_z \sum_u \| z_u - z_u^{(k-1)} \|^2_2 + \lambda \sum_{u,v} c_{uv}^{(k)} \| z_u - z_v \|^2_2$$

a generalization of Dirichlet energy
How Powerful Are One-Layer Attentions?

For any K-layer attention, there exists a one-layer model that induces the same denoising effect.

**Theorem 2 (Equivalence between Multi-Layer Attentions and One-Layer Attention)**

For any K-layer attention, there exists a one-layer model that induces the same denoising effect.
**Observation:** one-layer all-pair attention is expressive enough for propagating global information among arbitrary node pairs

**SGFormer:** one-layer single-head global attention + auxiliary GNN

- Simple attention with linear complexity:
  \[ Z^{(0)} = f_I(X) \]
  \[
  Q = f_Q(Z^{(0)}), \quad \hat{Q} = \frac{Q}{\|Q\|_F}, \quad K = f_K(Z^{(0)}), \quad \hat{K} = \frac{K}{\|K\|_F}, \quad V = f_V(Z^{(0)})
  \]
  \[
  D = \text{diag} \left( 1 + \frac{1}{N} \hat{Q} (\hat{K}^\top 1) \right), \quad Z = \beta D^{-1} \left[ V + \frac{1}{N} \hat{Q} (\hat{K}^\top V) \right] + (1 - \beta) Z^{(0)}
  \]

- Add an auxiliary GNN at the output layer:
  \[ Z_O = (1 - \alpha) Z + \alpha \text{GN}(Z^{(0)}, A), \quad \hat{Y} = f_O(Z_O) \]

Qitian Wu et al., Simplifying and Empowering Transformers on Large-Graph Representations, NeurIPS 2023
SGFormer: Scaling to Large Graphs

Challenge of training on large graphs:

The graph data cannot be loaded as a whole into a single GPU for training

Mini-batch sampling strategies:

1) using local graph adjacency

2) neighbor sampling

Qitian Wu et al., Simplifying and Empowering Transformers on Large-Graph Representations, NeurIPS 2023
## Comparison of Existing Graph Transformers

<table>
<thead>
<tr>
<th></th>
<th>pos emb</th>
<th>multi-head</th>
<th>pre-processing</th>
<th>all-pair expressivity</th>
<th>complexity</th>
<th>largest demo of #nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GraphTransformer [Dwivedi et al. 2020]</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>yes</td>
<td>$O(N^2)$</td>
<td>0.2K</td>
</tr>
<tr>
<td>Graphormer [Ying et al. 2021]</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>yes</td>
<td>$O(N^2)$</td>
<td>0.3K</td>
</tr>
<tr>
<td>SAT [Chen et al. 2022]</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>yes</td>
<td>$O(N^2)$</td>
<td>0.2K</td>
</tr>
<tr>
<td>EGT [Hussain et al. 2022]</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>yes</td>
<td>$O(N^2)$</td>
<td>0.5K</td>
</tr>
<tr>
<td>GraphGPS [Rampáse et al. 2022]</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>yes</td>
<td>$O(N^2)$</td>
<td>1.0K</td>
</tr>
<tr>
<td>NodeFormer [Wu et al. 2022]</td>
<td>R</td>
<td>R</td>
<td>-</td>
<td>yes</td>
<td>$O(N + E)$</td>
<td>2M</td>
</tr>
<tr>
<td>SGFormer</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>yes</td>
<td>$O(N + E)$</td>
<td>0.1B</td>
</tr>
</tbody>
</table>
# Experiment on Medium-Sized Graphs

## Results on medium-sized node classification graphs

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Cora</th>
<th>CiteSeer</th>
<th>PubMed</th>
<th>Actor</th>
<th>Squirrel</th>
<th>Chameleon</th>
<th>Deezer</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>2,708</td>
<td>3,327</td>
<td>19,717</td>
<td>7,600</td>
<td>2223</td>
<td>890</td>
<td>28,281</td>
</tr>
<tr>
<td># edges</td>
<td>5,278</td>
<td>4,552</td>
<td>44,324</td>
<td>29,926</td>
<td>46,998</td>
<td>8,854</td>
<td>92,752</td>
</tr>
<tr>
<td>GCN</td>
<td>81.6 ± 0.4</td>
<td>71.6 ± 0.4</td>
<td>78.8 ± 0.6</td>
<td>30.1 ± 0.2</td>
<td>38.6 ± 1.8</td>
<td>41.3 ± 3.0</td>
<td>62.7 ± 0.7</td>
</tr>
<tr>
<td>GAT</td>
<td>83.0 ± 0.7</td>
<td>72.1 ± 1.1</td>
<td>79.0 ± 0.4</td>
<td>29.8 ± 0.6</td>
<td>35.6 ± 2.1</td>
<td>39.2 ± 3.1</td>
<td>61.7 ± 0.8</td>
</tr>
<tr>
<td>SGC</td>
<td>80.1 ± 0.2</td>
<td>71.9 ± 0.1</td>
<td>78.7 ± 0.1</td>
<td>27.0 ± 0.9</td>
<td>39.3 ± 2.3</td>
<td>39.0 ± 3.3</td>
<td>62.3 ± 0.4</td>
</tr>
<tr>
<td>JKNet</td>
<td>81.8 ± 0.5</td>
<td>70.7 ± 0.7</td>
<td>78.8 ± 0.7</td>
<td>30.8 ± 0.7</td>
<td>39.4 ± 1.6</td>
<td>39.4 ± 3.8</td>
<td>61.5 ± 0.4</td>
</tr>
<tr>
<td>APPNP</td>
<td><strong>83.3 ± 0.5</strong></td>
<td>71.8 ± 0.5</td>
<td><strong>80.1 ± 0.2</strong></td>
<td>31.3 ± 1.5</td>
<td>35.3 ± 1.9</td>
<td>38.4 ± 3.5</td>
<td><strong>66.1 ± 0.6</strong></td>
</tr>
<tr>
<td>H₂GCN</td>
<td>82.5 ± 0.8</td>
<td>71.4 ± 0.7</td>
<td>79.4 ± 0.4</td>
<td>34.4 ± 1.7</td>
<td>35.1 ± 1.2</td>
<td>38.1 ± 4.0</td>
<td>66.2 ± 0.8</td>
</tr>
<tr>
<td>SIGN</td>
<td>82.1 ± 0.3</td>
<td>72.4 ± 0.8</td>
<td>79.5 ± 0.5</td>
<td>36.5 ± 1.0</td>
<td>40.7 ± 2.5</td>
<td>41.7 ± 2.2</td>
<td>66.3 ± 0.3</td>
</tr>
<tr>
<td>CPGNN</td>
<td>80.8 ± 0.4</td>
<td>71.6 ± 0.4</td>
<td>78.5 ± 0.7</td>
<td>34.5 ± 0.7</td>
<td>38.9 ± 1.2</td>
<td>40.8 ± 2.0</td>
<td>65.8 ± 0.3</td>
</tr>
<tr>
<td>GloGNN</td>
<td>81.9 ± 0.4</td>
<td>72.1 ± 0.6</td>
<td>78.9 ± 0.4</td>
<td>36.4 ± 1.6</td>
<td>35.7 ± 1.3</td>
<td>40.2 ± 3.9</td>
<td>65.8 ± 0.8</td>
</tr>
<tr>
<td>Graphormer&lt;sub&gt;SMALL&lt;/sub&gt;</td>
<td>OOM</td>
<td>OOM</td>
<td>OOM</td>
<td>OOM</td>
<td>OOM</td>
<td>OOM</td>
<td>OOM</td>
</tr>
<tr>
<td>Graphormer&lt;sub&gt;SMALLER&lt;/sub&gt;</td>
<td>75.8 ± 1.1</td>
<td>65.6 ± 0.6</td>
<td>OOM</td>
<td>OOM</td>
<td>40.9 ± 2.5</td>
<td>41.9 ± 2.8</td>
<td>OOM</td>
</tr>
<tr>
<td>Graphormer&lt;sub&gt;ULTRASMAL&lt;/sub&gt;</td>
<td>74.2 ± 0.9</td>
<td>63.6 ± 1.0</td>
<td>OOM</td>
<td>33.9 ± 1.4</td>
<td>39.9 ± 2.4</td>
<td>41.3 ± 2.8</td>
<td>OOM</td>
</tr>
<tr>
<td>GraphTrans&lt;sub&gt;SMALL&lt;/sub&gt;</td>
<td>80.7 ± 0.9</td>
<td>69.5 ± 0.7</td>
<td>OOM</td>
<td>32.6 ± 0.7</td>
<td><strong>41.0 ± 2.8</strong></td>
<td>42.8 ± 3.3</td>
<td>OOM</td>
</tr>
<tr>
<td>GraphTrans&lt;sub&gt;ULTRASMAL&lt;/sub&gt;</td>
<td>81.7 ± 0.6</td>
<td>70.2 ± 0.8</td>
<td>77.4 ± 0.5</td>
<td>32.1 ± 0.8</td>
<td>40.6 ± 2.4</td>
<td>42.2 ± 2.9</td>
<td>OOM</td>
</tr>
<tr>
<td>NodeFormer</td>
<td>82.2 ± 0.9</td>
<td><strong>72.5 ± 1.1</strong></td>
<td>79.9 ± 1.0</td>
<td><strong>36.9 ± 1.0</strong></td>
<td>38.5 ± 1.5</td>
<td>34.7 ± 4.1</td>
<td>66.4 ± 0.7</td>
</tr>
<tr>
<td>SGFormer</td>
<td><strong>84.5 ± 0.8</strong></td>
<td>72.6 ± 0.2</td>
<td><strong>80.3 ± 0.6</strong></td>
<td><strong>37.9 ± 1.1</strong></td>
<td><strong>41.8 ± 2.2</strong></td>
<td><strong>44.9 ± 3.9</strong></td>
<td><strong>67.1 ± 1.1</strong></td>
</tr>
</tbody>
</table>
Experiment on Large-Sized Graphs

Results on large node classification graphs

<table>
<thead>
<tr>
<th>Method</th>
<th>ogbn-proteins</th>
<th>Amazon2m</th>
<th>poking</th>
<th>ogbn-arxiv</th>
<th>ogbn-papers100M</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>132,534</td>
<td>2,449,029</td>
<td>1,632,803</td>
<td>169,343</td>
<td>111,059,956</td>
</tr>
<tr>
<td># edges</td>
<td>39,561,252</td>
<td>61,859,140</td>
<td>30,622,564</td>
<td>1,166,243</td>
<td>1,615,685,872</td>
</tr>
<tr>
<td>MLP</td>
<td>72.04 ± 0.48</td>
<td>63.46 ± 0.10</td>
<td>60.15 ± 0.03</td>
<td>55.50 ± 0.23</td>
<td>47.24 ± 0.31</td>
</tr>
<tr>
<td>GCN</td>
<td>72.51 ± 0.35</td>
<td>83.90 ± 0.10</td>
<td>62.31 ± 1.13</td>
<td>71.74 ± 0.29</td>
<td>OOM</td>
</tr>
<tr>
<td>SGC</td>
<td>70.31 ± 0.23</td>
<td>81.21 ± 0.12</td>
<td>52.03 ± 0.84</td>
<td>67.79 ± 0.27</td>
<td>63.29 ± 0.19</td>
</tr>
<tr>
<td>GCN-NSampler</td>
<td>73.51 ± 1.31</td>
<td>83.84 ± 0.42</td>
<td>63.75 ± 0.77</td>
<td>68.50 ± 0.23</td>
<td>62.04 ± 0.27</td>
</tr>
<tr>
<td>GAT-NSampler</td>
<td>74.63 ± 1.24</td>
<td>85.17 ± 0.32</td>
<td>62.32 ± 0.65</td>
<td>67.63 ± 0.23</td>
<td>63.47 ± 0.39</td>
</tr>
<tr>
<td>SIGN</td>
<td>71.24 ± 0.46</td>
<td>80.98 ± 0.31</td>
<td>68.01 ± 0.25</td>
<td>70.28 ± 0.25</td>
<td>65.11 ± 0.14</td>
</tr>
<tr>
<td>NodeFormer</td>
<td>77.45 ± 1.15</td>
<td>87.85 ± 0.24</td>
<td>70.32 ± 0.45</td>
<td>59.90 ± 0.42</td>
<td>-</td>
</tr>
<tr>
<td>SGFormer</td>
<td>79.53 ± 0.38</td>
<td>89.09 ± 0.10</td>
<td>73.76 ± 0.24</td>
<td>72.63 ± 0.13</td>
<td>66.01 ± 0.37</td>
</tr>
</tbody>
</table>

**SGFormer** can be trained in full-graph manner on ogbn-arxiv

Mini-batch training for proteins, Amazon2M, poking with batch size 10K/100K

For Papers100M, using batch size 0.4M only requires 3.5 hours on a 24GB GPU
## Experiment Results

### Comparison of training/inference time per epoch and memory cost

<table>
<thead>
<tr>
<th>Method</th>
<th>Tr (ms)</th>
<th>Inf (ms)</th>
<th>Mem (GB)</th>
<th>Tr (ms)</th>
<th>Inf (ms)</th>
<th>Mem (GB)</th>
<th>Tr (ms)</th>
<th>Inf (ms)</th>
<th>Mem (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphormer</td>
<td>563.5</td>
<td>537.1</td>
<td>5.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GraphTrans</td>
<td>160.4</td>
<td>40.2</td>
<td>3.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>NodeFormer</td>
<td>68.5</td>
<td>30.2</td>
<td>1.2</td>
<td>321.4</td>
<td>135.5</td>
<td>2.9</td>
<td>5369.5</td>
<td>1410.0</td>
<td>4.6</td>
</tr>
<tr>
<td>SGFormer</td>
<td>15.0</td>
<td>3.8</td>
<td>0.9</td>
<td>15.4</td>
<td>4.4</td>
<td>1.0</td>
<td>2481.4</td>
<td>382.5</td>
<td>2.7</td>
</tr>
</tbody>
</table>

### Scalability test of training time/memory costs w.r.t. number of nodes

- **Training time**
- **GPU memory cost**
Experiment Results

**Obs 1:** one-layer attention of SGFormer is highly competitive and efficient as well

**Obs 2:** one-layer attention of other (all-pair) models can also yield promising acc
Conclusions

Graph Transformers have become a popular research topic in GNN community.

Some open problems:
1) poor scalability (quadratic complexity)
2) lack of principled guidance for attention designs
3) inefficiency, complicated model

   - all-pair message passing with linear complexity
   - scale to 2M nodes
   - handle no-graph tasks
   Codes: https://github.com/qitianwu/NodeFormer

[2] DIFFormer: Scalable (Graph) Transformers Induced by Energy Constrained Diffusion. in ICLR 2023
   - principled global attention designs
   - superiority for low labeled rates
   Codes: https://github.com/qitianwu/DIFFormer

[3] Simplifying and Empowering Transformers for Large-Graph Representations. in NeurIPS 2023
   - simple attention (one-layer single-head)
   - 30x inference speed-up
   - scale to 0.1B nodes
   Codes: https://github.com/qitianwu/SGFormer

Email: echo740@sjtu.edu.cn  WeChat: myronwqt228