L²-GCN: Layer-Wise and Learned Efficient Training of Graph Convolutional Networks

Yuning You*, Tianlong Chen*, Zhangyang Wang, Yang Shen

Texas A&M University

* Equal Contribution

This work was presented at CVPR 2019
Motivation

- The forward propagation of GCN layer:
  - FA: aggregation of the neighborhood information;
  - FT: non-linear transformation.

- Concatenation of FA & FT $\rightarrow$ inefficient GCN training for large graphs.

- Decoupling FA & FT in GCN training can greatly reduce computational burden.
L-GCN: Layer-wise GCN

- Propose layer-wise training to decouple FA & FT.

- For each GCN layer, FA is performed once then fed for FT.

- Optimization is for each layer individually.
Theoretical Justification of L-GCN

• We provide further analysis following the graph isomorphism framework\cite{1}:
  
  – The power of aggregation-based GNN := the ability it maps different graphs (rooted subtrees of vertices) into different embeddings;
  
  – GNN is at most as powerful as the WL test.

• We prove that if GCN is as powerful as the WL test through conventional training, there exists the same powerful model through layer-wise training (see Theorem 5).


Theoretical Justification of L-GCN

- Insight in Theorem 5: for the powerful enough GCN through conventional training, we might obtain the same powerful model through layer-wise training.

- Furthermore, we prove that if GCN is not as powerful as the WL test through conventional training, through layer-wise training its power is non-decreasing with layer number increasing (see Theorem 6).

- Insight in Theorem 6: for the not powerful enough GCN through conventional training, through layer-wise training we might obtain a more powerful model if we make it deeper.
L²-GCN: Layer-wise and Learned GCN

- Lastly, to avoid manually adjusting the training epochs for each layer, a learned controller is proposed to automatically deal with this process.
Experiments

- Experiments show that L-GCN is faster than state-of-the-arts by at least an order of magnitude, with a consistent of memory usage not dependent on dataset size, while maintaining comparable prediction performance. With the learned controller, L²-GCN can further cut the training time in half.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Fl (%) Time Memory</td>
<td>Fl (%) Time Memory</td>
<td>Fl (%) Time Memory</td>
<td>Fl (%) Time Memory</td>
<td>Fl (%) Time Memory</td>
</tr>
<tr>
<td>Cora</td>
<td>85.0 18s 655M</td>
<td>85.5 6.02s 659M</td>
<td>85.4 5.47s 253M</td>
<td>84.7 0.45s 619M</td>
<td>84.1 0.38s 619M</td>
</tr>
<tr>
<td>PubMed</td>
<td>86.5 483s 675M</td>
<td>87.4 32s 851M</td>
<td>86.4 118s 375M</td>
<td>86.8 2.93s 619M</td>
<td>85.8 1.50s 631M</td>
</tr>
<tr>
<td>PPI</td>
<td>68.8 402s 849M</td>
<td>- - -</td>
<td>98.6 63s 759M</td>
<td>97.2 49s 629M</td>
<td>96.8 26s 631M</td>
</tr>
<tr>
<td>Reddit</td>
<td>93.4 998s 4343M</td>
<td>92.6 761s 4429M</td>
<td>96.0 201s 1271M</td>
<td>94.2 44s 621M</td>
<td>94.0 34s 635M</td>
</tr>
<tr>
<td>Amazon-670K</td>
<td>83.1 2153s 849M</td>
<td>76.1 548s 1621M</td>
<td>92.7 534s 625M</td>
<td>91.6 54s 601M</td>
<td>91.2 30s 613M</td>
</tr>
<tr>
<td>Amazon-3M</td>
<td>- - -</td>
<td>- - -</td>
<td>- - -</td>
<td>- - -</td>
<td>- - -</td>
</tr>
</tbody>
</table>

TAMU HPRC cluster: Terra (GPU); Software: Anaconda/3-5.0.0.1
Thank you for listening.

Code: https://github.com/Shen-Lab/L2-GCN