Decoupled Greedy Learning of Graph Neural Networks

Yewen Wang*, Jian Tang*, Yizhou Sun*, Wolf Guy*
*: Department of Computer Science, University of California, Los Angeles, †: MILA

Motivation
Improve the efficiency of GNN training:
- Problem 1: Recursive computation
- Problem 2: Update-locking
  - each layer heavily relies on upper layers’ feedback to
  update itself
  - it must wait for the information to propagate through
    the whole network before updating

Main Contributions
- Introduce a decoupled greedy learning algorithm for GNNs
- achieves update-unlocking
- enables GNN layers to be trained in parallel
- Less time, less per-GPU memory, good for time/hardware-limited applications
- Leverage a lazy-update scheme
- Further improves efficiency
- Our method can be used in more general cases:
  - not limited to the deep GCN model
  - not limited to node classification task
  - can be combined with other scalability-enhancing GNNs and can be applied to other graph-related tasks

Conventional GNN and Layer-wise GNN

![Conventional GNN and Layer-wise GNN Diagram](Image)

Fig1. High level framework of conventional GNN (upper) and layer-wise GNN. The aggregation step (A) corresponds to $D^{-1/2}AD^{-1/2}H^{(l-1)}$operation and the transformation step corresponds to $f(s(W^{(l)}O))$operation

Decoupled Greedy Learning GNN

- Method:
  - Decouple the GNN into different layers, append one auxiliary greedy objective (node classification) after each layer, and enable parallelization.
  - Leverage the Lazy Update scheme to improve efficiency.
  - Analogy: block coordinate descent method.
- Main Algorithms

Algorithm 1: Decoupled Greedy Learning (DGL) of GNNs

1. Initialize: $H^{(0)} = X$.
2. for $t = 1$ to $T$ do
3. for $l = 1$ to $L$ do
4. $H^{(l)} = f_s(LU(H^{(l-1)}, W^{(l)}))$ // Get node embeddings and store them as $H^{(l)}$.
5. $(W^{(l)}, b^{(l)})$ // Update with $f_{\text{node}}(W^{(l)}, b^{(l)})$.
6. end for
7. end for

Algorithm 2: Decoupled Greedy Learning (DGL) of GNNs with Lazy Update Scheme

1. Initialize: $H^{(0)} = X$.
2. for $t = 1$ to $T$ do
3. for $l = 1$ to $L$ do
4. $H^{(l)} = LU(H^{(l-1)}, W^{(l)})$ // Get node embeddings.
5. $(W^{(l)}, b^{(l)})$ // Update with $f_{\text{node}}(W^{(l)}, b^{(l)})$.
6. end for
7. end for

Complexity Comparison

<table>
<thead>
<tr>
<th>Methods</th>
<th>Memory (per GPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-Batch GCN</td>
<td>$O(LNK + LK^2)$</td>
<td>$O(LNK + LK^2)$</td>
</tr>
<tr>
<td>GraphSage</td>
<td>$O(MR + LK)$</td>
<td>$O(MR + LK)$</td>
</tr>
<tr>
<td>VR-GCN</td>
<td>$O(MR + LK)$</td>
<td>$O(MR + LK)$</td>
</tr>
<tr>
<td>FanGCN</td>
<td>$O(LK^{layer} + LK)$</td>
<td>$O(LK^{layer} + LK)$</td>
</tr>
<tr>
<td>LADIES</td>
<td>$O(LK^{layer} + LK)$</td>
<td>$O(LK^{layer} + LK)$</td>
</tr>
<tr>
<td>ClassGCN</td>
<td>$O(MR + LK)$</td>
<td>$O(MR + LK)$</td>
</tr>
<tr>
<td>LU-DGL (ours)</td>
<td>$O(NK + 2K)$</td>
<td>$O(NK + 2K)$</td>
</tr>
<tr>
<td>LU-DGL (ours)</td>
<td>$O(NK + 2K)$</td>
<td>$O(NK + 2K)$</td>
</tr>
</tbody>
</table>

Tab.1 Summary of Complexity. $D$ is the avg degree, $b$ is the batch size, $N_{\text{node}}$ and $S_{\text{layer}}$ are the num of sampled neighbors the sampling-based baseline, $K$ is the dim of embedding vectors, $L$ is the num of layers, $N$ is the num of nodes in the graph, $A$ is the adj matrix, $T$ is the num of iterations, $T_{\text{wait}}$ is the waiting time for LU-DGL-GCN.

Results

- Compare GCN, LGCN, LU-DGL-GCN: Our method is very efficient , it can save time and per-GPU memory without too much compromising on performance.
- Compare GN, LGN, LU-DGL-GCN: Our method is not limited to GCN but can be combined with other GNN models.
- Compare LADIES, LADIES, LU-DGL-LADIES: The proposed method can be combined with other scalability-enhancing methods for GNNs.

Acknowledgement
This work was partially funded by IVADO Professor startup & operational funds, IVADO Fundamental Research Project grant PRF-2019-3583197277 [G.W.], NSF III-1705169, NSF CAREER Award 1741634, NSF #1937599, DARPA HR00112090027, Okawa Foundation Grant, and Amazon Research Award [Y.S.]