LightNE: A Lightweight Graph Processing System for Network Embedding

Jiezhong Qiu, Laxman Dhulipala, Jie Tang, Richard Peng, Chi Wang

https://github.com/xptree/LightNE.
Roadmap

• Introduction to Network Embedding

• LightNE: Co-design of Algorithm and System

• Experiments on graphs with billions of edges.
Real-world Graphs

Knowledge Graph  Internet Graph  Transportation Graph

Question: How to design machine learning models for large-scale real-world graphs?
Background: Network Embedding

• Given a graph $G = (V, E)$, aim to learn a function $f : V \rightarrow \mathbb{R}^d$ to capture neighborhood similarity and community membership.

A toy example from DeepWalk [1]

Background: DeepWalk

- Sampling random walk sequences on the input graph
- Train a skip-graph model (word2vec) on the sampled sequences

Scalability issue:
- Alibaba embeds a 600-billion-node commodity graph by first partitioning it into 50-million-node subgraphs, and then embedding each subgraph separately with 100 GPUs running DeepWalk [1]

Background: Network Embedding as Matrix Factorization

- NetMF[1]: DeepWalk is implicitly and asymptotically factorizing:

\[ M \triangleq \text{trunc}_\log^o \left( \frac{\text{vol}(G)}{b} \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r D^{-1} \right) \]

- NetSMF[2]:
  - Sparisify r-step random walk matrix \((D^{-1} A)^r\) with PathSampling.
  - Need \(O(m\log n)\) samples

---

Algorithm 1: PathSampling.

1. **Procedure** PathSample\((G, u, v, r)\)
2. Let a random edge \((u, v)\) be given.
3. Sample a random number \(s\) uniformly in \([0, r - 1]\).
4. \(u' \leftarrow\) random walk \(u\) for \(s\) steps on graph \(G\)
5. \(v' \leftarrow\) random walk \(v\) for \(r - 1 - s\) steps on graph \(G\).
6. **return** edge \((u', v')\)


Roadmap

• Introduction to Network Embedding

• LightNE: Co-design of Algorithm and System

• Experiments on graphs with billions of edges.
LightNE: Design Goal

- **Scalable**: Embed graphs with 1B edges within 1.5 hours.
- **Lightweight**: Occupy hardware costs below 100 dollars measured by cloud rent to process 1B to 100B edges.
- **Accurate**: Achieve the highest accuracy in downstream tasks under the same time budget and similar resources.
LightNE: Algorithm and System Co-design

- Store input graph in GBBS compressed CSR format
- Highly optimized parallel processing

GBBS Parallel Hashing: selectively sample edges in parallel
  - Sample edge \((u, v)\) with prob. \(P_e = 1/d_u + 1/d_v\)
  - Reduce #samples \(M\) from \(O(m \log n)\) to \(O((m \log n)^{1/2})\)

Parallel Randomized SVD: scale embedding by \(X \leftarrow \sum_{r=0}^{k} c_r (I - D^{-1/2}) X (I - D^{-1/2})^T\)
  - \(c_r\)'s are chosen to be coefficients of Chebyshev polynomials

Accelerated Spectral Propagation


Roadmap

• Introduction to Network Embedding

• LightNE: Co-design of Algorithm and System

• Experiments on graphs with billions of edges.
  • Baselines: GraphVite, Pytorch-Big-Graph, NetSMF, ProNE
Comparison to NetSMF and ProNE

- Open Academic Graph (67,768,244 nodes, 895,368,962 edges)
- LightNE-small (#samples=\(m\)) and LightNE-large (#samples=200\(m\))

<table>
<thead>
<tr>
<th>Metric</th>
<th>Method</th>
<th>Time</th>
<th>0.001%</th>
<th>0.01%</th>
<th>0.1%</th>
<th>1%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NetSMF (M=8Tm)</td>
<td>22.4 h</td>
<td>30.43</td>
<td>31.66</td>
<td>35.77</td>
<td>38.88</td>
</tr>
<tr>
<td></td>
<td>ProNE+</td>
<td>21 min</td>
<td>23.56</td>
<td>29.32</td>
<td>31.17</td>
<td>31.46</td>
</tr>
<tr>
<td></td>
<td>LIGHTNE-Small</td>
<td>20.9 min</td>
<td>23.89</td>
<td>30.23</td>
<td>32.16</td>
<td>32.35</td>
</tr>
<tr>
<td></td>
<td>LIGHTNE-Large</td>
<td>1.53 h</td>
<td>44.50</td>
<td>52.89</td>
<td>54.98</td>
<td>55.23</td>
</tr>
</tbody>
</table>

- LightNE-Large achieves **15x speedup** (1.53h v.s. 22.4h) and **significant performance gain**, comparing to NetSMF.
- Not only does LightNE-Small run **faster** than ProNE+ (20.9 min v.s. 21 min), but also **outperforms ProNE+ significantly**.
- Estimated price of LightNE-Large: \(1.53h \times 13$/h = 20\$\)
Comparison to NetSMF and ProNE

Figure 2: Efficiency-effectiveness trade-off curve of LightNE.
Very Large Graphs

None of the existing network embedding systems can handle such large graphs in a single machine!

<table>
<thead>
<tr>
<th></th>
<th>ClueWeb-Sym</th>
<th>Hyperlink2014-Sym</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>978,408,098</td>
<td>1,724,573,718</td>
</tr>
<tr>
<td>$m$</td>
<td>74,744,358,622</td>
<td>124,141,874,032</td>
</tr>
</tbody>
</table>

Figure 3: HITS@K ($K = 1, 10, 50$) of LIGHTNE w.r.t. the number of samples.
Conclusion

• Propose LightNE, a **cost-effective, scalable, and high quality** network embedding system that scales to graphs with hundreds of billions of edges on a **single machine**.

• Introduce 4 techniques to network embedding for the first time:
  1. A new downsampling method to reduce the sample complexity of NetSMF.
  2. A parallel graph processing stack GBBS for memory efficiency and scalability;
  3. Sparse parallel hash table to maintain the matrix sparsifier in memory
  4. Intel MKL for efficient randomized SVD and spectral propagation.
Thanks!

LightNE: A Lightweight Graph Processing System for Network Embedding

Jiezhong Qiu, Laxman Dhulipala, Jie Tang, Richard Peng, Chi Wang

https://github.com/xptree/LightNE.