Self-supervised Graph Learning for Recommendation
Outline

- Background
- Model: SGL
- Experiments
- Conclusion & Future Work
Recap GCN for CF

Abstract Paradigm

- Neighbor Aggregation
  
  (1) Representation aggregation layers
  \[
  Z^{(l)} = H(Z^{(l-1)}, G) \\
  a_u^{(l)} = f_{\text{aggregate}} \left( \{z_i^{(l-1)} | i \in N_u \} \right), \\
  z_u^{(l)} = f_{\text{combine}}(z_u^{(l-1)}, a_u^{(l)}),
  \]

  (2) Readout layer
  \[
  z_u = f_{\text{readout}} \left( \{z_u^{(l)} | l = [0, \cdots, L] \} \right)
  \]

- Supervised Learning Loss
  \[
  L_{\text{main}} = \sum_{(u, i, j) \in O} -\log \sigma(\hat{y}_{ui} - \hat{y}_{uj}) \\
  \hat{y}_{ui} = z_u^T z_i
  \]

He et al. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. SIGIR 2020
Limitations in existing GCNs:

- **Sparse Supervision Signal**
  - The supervision signal comes from the observed interactions ➔ extremely sparse

- **Skewed Data Distribution**
  - Power-law distribution
  - High-degree items exert larger impact on the representation learning

- **Noises in Interactions**
  - Implicit feedback makes the learning more vulnerable to interaction noises

---

**Self-supervised Learning**

- Obtain “labels” from the data itself
- Predict part of the data from other parts
Augmentation on graph structure:

- The features of users and items are discrete
- Users and items in the graph are inherently connected and dependent on each other
Data Augmentations

\[ Z_1^{(l)} = H\left( Z_1^{(l-1)}, s_1(\mathcal{G}) \right), \quad Z_2^{(l)} = H\left( Z_2^{(l-1)}, s_2(\mathcal{G}) \right), \quad s_1, s_2 \sim S \]

◆ Node Dropout (ND)

\[ s_1(\mathcal{G}) = (M' \odot \mathcal{V}, \mathcal{E}), \quad s_2(\mathcal{G}) = (M'' \odot \mathcal{V}, \mathcal{E}) \quad M', M'' \in \{0, 1\}^{\lvert \mathcal{V} \rvert} \]

- Identify the influential nodes from differently augmented views
- Make the representation learning less sensitive to structure changes

◆ Edge Dropout (ED)

\[ s_1(\mathcal{G}) = (\mathcal{V}, M' \odot \mathcal{E}), \quad s_2(\mathcal{G}) = (\mathcal{V}, M'' \odot \mathcal{E}) \quad M', M'' \in \{0, 1\}^{\lvert \mathcal{E} \rvert} \]

- Capture the useful patterns of the local structures of a node
- Endow the representations more robustness against the presence of single interactions, especially the noisy interactions.

◆ Random Walk (RW)

\[ s_1(\mathcal{G}) = (\mathcal{V}, M_1^{(l)} \odot \mathcal{E}), \quad s_2(\mathcal{G}) = (\mathcal{V}, M_2^{(l)} \odot \mathcal{E}) \quad M_1^{(l)}, M_2^{(l)} \in \{0, 1\}^{\lvert \mathcal{E} \rvert} \]

- constructing an individual subgraph for each node with random walk
- Layer-sensitive local structure
Objective Function

- **Contrastive Loss --- InfoNCE**
  - maximize the agreement of positive pairs
  - minimize that of negative pairs

\[
\mathcal{L}^{\text{user}}_{ssl} = \sum_{u \in \mathcal{U}} - \log \frac{\exp(s(z'_u, z''_u)/\tau)}{\sum_{v \in \mathcal{U}} \exp(s(z'_u, z''_v)/\tau)}
\]

\[
\mathcal{L}_{ssl} = \mathcal{L}^{\text{user}}_{ssl} + \mathcal{L}_{item}
\]

- **Supervised Loss --- BPR**

\[
\mathcal{L}_{main} = \sum_{(u,i,j) \in \mathcal{O}} -\log \sigma(\hat{y}_{ui} - \hat{y}_{uj})
\]

- **Multi-task Training**

\[
\mathcal{L} = \mathcal{L}_{main} + \lambda_1 \mathcal{L}_{ssl} + \lambda_2 \|\Theta\|_2^2
\]
Hard Negative Mining

- Gradient of the self-supervised

\[
\frac{\partial L_{\text{ssl}}^\text{user}}{\partial z_u'} (u) = \frac{1}{\tau \|z_u'\|} \left\{ c(u) + \sum_{v \in U \setminus \{u\}} c(v) \right\}
\]

\[
P_{uv} = \frac{\exp(s_u'^T s_v''/\tau)}{\sum_{v \in U} \exp(s_u'^T s_v''/\tau)}; s_u' = \frac{z_u'}{\|z_u'\|} \quad \text{and} \quad s_v'' = \frac{z_v''}{\|z_v''\|}
\]

\[
c(u) = \left( s_v'' - (s_u'^T s_v'') s_u' \right) (P_{uu} - 1),
\]

\[
c(v) = \left( s_v'' - (s_u'^T s_v'') s_u' \right) P_{uv},
\]

Contribution from negative sample

\[
L_2 \text{ norm of } c(v) \quad \|c(v)\|_2 \propto \sqrt{1 - (s_u'^T s_v'')^2 \exp(s_u'^T s_v''/\tau)}
\]

\[
g(x) = \sqrt{1 - x^2} \exp\left(\frac{x}{\tau}\right) \quad x \text{ is the cosine similarity between } s_u' \text{ and } s_v''
\]

\[
\begin{cases} 
-1 \leq x < 0 & \text{Easy negative} \\
0 < x \leq 1 & \text{Hard negative} \quad \Rightarrow \text{offer much larger gradients to guide the optimization}
\end{cases}
\]

Wu et al. Self-supervised Graph Learning for Recommendation. SIGIR 2021
Experiment Settings

• **Datasets:**
  - Yelp2018, Amazon-Book, Alibaba-iFashion

• **Evaluation Metrics:**
  - recall@20, ndcg@20

• **Dataset partition:** randomly select 80% data for training set, and 20% data for testing set.

**Table 2: Statistics of the datasets.**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Users</th>
<th>#Items</th>
<th>#Interactions</th>
<th>Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yelp2018</td>
<td>31,668</td>
<td>38,048</td>
<td>1,561,406</td>
<td>0.00130</td>
</tr>
<tr>
<td>Amazon-Book</td>
<td>52,643</td>
<td>91,599</td>
<td>2,984,108</td>
<td>0.00062</td>
</tr>
<tr>
<td>Alibaba-iFashion</td>
<td>300,000</td>
<td>81,614</td>
<td>1,607,813</td>
<td>0.00007</td>
</tr>
</tbody>
</table>
## Experiment Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Yelp2018</th>
<th>Amazon-Book</th>
<th>Alibaba-iFashion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>Recall</td>
<td>NDCG</td>
<td>Recall</td>
</tr>
<tr>
<td>NGCF</td>
<td>0.0579</td>
<td>0.0477</td>
<td>0.0344</td>
</tr>
<tr>
<td>LightGCN</td>
<td>0.0639</td>
<td>0.0525</td>
<td>0.0411</td>
</tr>
<tr>
<td>Mult-VAE</td>
<td>0.0584</td>
<td>0.0450</td>
<td>0.0407</td>
</tr>
<tr>
<td>DNN+SSL</td>
<td>0.0483</td>
<td>0.0382</td>
<td>0.0438</td>
</tr>
<tr>
<td><strong>SGL-ED</strong></td>
<td><strong>0.0675</strong></td>
<td><strong>0.0555</strong></td>
<td><strong>0.0478</strong></td>
</tr>
<tr>
<td>%Improvement</td>
<td>5.63%</td>
<td>5.71%</td>
<td>9.13%</td>
</tr>
<tr>
<td>p-value</td>
<td>5.92e-8</td>
<td>1.89e-8</td>
<td>5.07e-10</td>
</tr>
</tbody>
</table>

- SGL achieves significant improvements over the state-of-the-art baselines → **outstanding performance**
**Experiment Results**

- Performance comparison among different SGL implementations and LightGCN at different layers:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Yelp2018</th>
<th></th>
<th>Amazon-Book</th>
<th></th>
<th>Alibaba-iFashion</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recall</td>
<td>NDCG</td>
<td>Recall</td>
<td>NDCG</td>
<td>Recall</td>
<td>NDCG</td>
</tr>
<tr>
<td>#Layer</td>
<td>Method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LightGCN</td>
<td>0.0631</td>
<td>0.0515</td>
<td>0.0384</td>
<td>0.0298</td>
<td>0.0990</td>
</tr>
<tr>
<td></td>
<td>SGL-ND</td>
<td>0.0643(+1.9%)</td>
<td>0.0529(+2.7%)</td>
<td>0.0432(+12.5%)</td>
<td>0.0334(+12.1%)</td>
<td>0.1133(+14.4%)</td>
</tr>
<tr>
<td></td>
<td>SGL-ED</td>
<td>0.0637(+1.0%)</td>
<td>0.0526(+2.1%)</td>
<td>0.0451(+17.4%)</td>
<td>0.0353(+18.5%)</td>
<td>0.1125(+13.6%)</td>
</tr>
<tr>
<td></td>
<td>SGL-RW</td>
<td>0.0637(+1.0%)</td>
<td>0.0526(+2.1%)</td>
<td>0.0451(+17.4%)</td>
<td>0.0353(+18.5%)</td>
<td>0.1125(+13.6%)</td>
</tr>
<tr>
<td>1 Layer</td>
<td>LightGCN</td>
<td>0.0622</td>
<td>0.0504</td>
<td>0.0411</td>
<td>0.0315</td>
<td>0.1066</td>
</tr>
<tr>
<td></td>
<td>SGL-ND</td>
<td>0.0658(+5.8%)</td>
<td>0.0538(+6.7%)</td>
<td>0.0427(+3.9%)</td>
<td>0.0335(+6.3%)</td>
<td>0.1106(+3.8%)</td>
</tr>
<tr>
<td></td>
<td>SGL-ED</td>
<td>0.0668(+7.4%)</td>
<td>0.0549(+8.9%)</td>
<td>0.0468(+13.9%)</td>
<td>0.0371(+17.8%)</td>
<td>0.1091(+2.3%)</td>
</tr>
<tr>
<td></td>
<td>SGL-RW</td>
<td>0.0644(+3.5%)</td>
<td>0.0530(+5.2%)</td>
<td>0.0453(+10.2%)</td>
<td>0.0358(+13.7%)</td>
<td>0.1091(+2.3%)</td>
</tr>
<tr>
<td>2 Layers</td>
<td>LightGCN</td>
<td>0.0639</td>
<td>0.0525</td>
<td>0.0410</td>
<td>0.0318</td>
<td>0.1078</td>
</tr>
<tr>
<td></td>
<td>SGL-ND</td>
<td>0.0644(+0.8%)</td>
<td>0.0528(+0.6%)</td>
<td>0.0440(+7.3%)</td>
<td>0.0346(+8.8%)</td>
<td>0.1126(4.5%)</td>
</tr>
<tr>
<td></td>
<td>SGL-ED</td>
<td><strong>0.0675(+5.6%)</strong></td>
<td><strong>0.0555(+5.7%)</strong></td>
<td><strong>0.0478(+16.6%)</strong></td>
<td><strong>0.0379(+19.2%)</strong></td>
<td><strong>0.1139(+5.7%)</strong></td>
</tr>
<tr>
<td></td>
<td>SGL-RW</td>
<td>0.0667(+4.4%)</td>
<td>0.0547(+4.2%)</td>
<td>0.0457(+11.5%)</td>
<td>0.0356(+12.0%)</td>
<td>0.1126(+4.5%)</td>
</tr>
<tr>
<td>3 Layers</td>
<td>LightGCN</td>
<td>0.0639</td>
<td>0.0525</td>
<td>0.0410</td>
<td>0.0318</td>
<td>0.1078</td>
</tr>
<tr>
<td></td>
<td>SGL-ND</td>
<td>0.0644(+0.8%)</td>
<td>0.0528(+0.6%)</td>
<td>0.0440(+7.3%)</td>
<td>0.0346(+8.8%)</td>
<td>0.1126(4.5%)</td>
</tr>
<tr>
<td></td>
<td>SGL-ED</td>
<td><strong>0.0675(+5.6%)</strong></td>
<td><strong>0.0555(+5.7%)</strong></td>
<td><strong>0.0478(+16.6%)</strong></td>
<td><strong>0.0379(+19.2%)</strong></td>
<td><strong>0.1139(+5.7%)</strong></td>
</tr>
<tr>
<td></td>
<td>SGL-RW</td>
<td>0.0667(+4.4%)</td>
<td>0.0547(+4.2%)</td>
<td>0.0457(+11.5%)</td>
<td>0.0356(+12.0%)</td>
<td>0.1126(+4.5%)</td>
</tr>
</tbody>
</table>

- Training curves of SGL-ED and LightGCN:

  - InfoNCE vs. BPR
  - Hard negative mining

(a) Yelp2018-L3    (b) Amazon-Book-L3
Benefits of SGL

- Long-tail Recommendation
- Robustness to Noisy Interactions
Study of SGL

• Effect of Temperature

(a) Yelp2018

(b) Amazon-Book

• Effect of Negatives and Pretrain

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Yelp2018</th>
<th>Amazon-Book</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>Recall</td>
<td>NDCG</td>
</tr>
<tr>
<td>SGL-ED-batch</td>
<td>0.0670</td>
<td>0.0549</td>
</tr>
<tr>
<td>SGL-ED-merge</td>
<td>0.0671</td>
<td>0.0547</td>
</tr>
<tr>
<td>SGL-pre</td>
<td>0.0653</td>
<td>0.0533</td>
</tr>
<tr>
<td>SGL-ED</td>
<td><strong>0.0675</strong></td>
<td><strong>0.0555</strong></td>
</tr>
</tbody>
</table>
Conclusion & Future Work

✅ Conclusion

- A model-agnostic framework SGL to supplement the supervised recommendation task with self-supervised learning on user-item graph
- Devise three types of data augmentation from different aspects to construct the auxiliary contrastive task
- Prove in theory that SGL inherently encourages learning from hard negatives

✅ Future Work

- Explore new perspectives, such as counterfactual learning to identify influential data points
- Pre-training and fine-tuning in recommendation?
- Fulfill the potential of SSL to address the long-tail issue
Thanks & QA?

• The code is available at https://github.com/wujcan/SGL