When Does Self-Supervision Help Graph Convolutional Networks?

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

Texas A&M University

* Equal Contribution

This work was presented at ICML 2020
Contents

• Motivation
• Contribution 1. How to incorporate self-supervision (SS) in graph convolutional networks (GCNs)?
• Contribution 2. How to design SS tasks to improve model generalizability?
• Contribution 3. Does SS boost model robustness?
• Conclusions
Motivation

• Semi-supervised (SS) learning is an important field of graph-based applications with abundant unlabeled data available;

• Using unlabeled data, SS is a promising technique in the few-shot scenario for computer vision;

• SS in graph neural networks for graph-structured data is still under-explored with an exception (M3S, AAAI’19).
Contribution 1. How to incorporate SS in GCNs?

- We perform a systematic study on SS + GCNs:
  - 1. How to incorporate SS in GCNs?
    - Pretraining & finetuning;
    - Self-training (M3S, AAAI’19);
    - Multi-task learning.
Contribution 1. How to incorporate SS in GCNs?

• Multi-task learning:
  – Empirically **outperforms** other two schemes;
  – We regard the SS task as a **regularization** term throughout the network training;
  – Act as a **data-driven** regularizer.

<table>
<thead>
<tr>
<th>Pipeline</th>
<th>GCN</th>
<th>P&amp;F</th>
<th>MTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>79.10 ± 0.21</td>
<td>79.19 ± 0.21</td>
<td>80.00 ± 0.74</td>
</tr>
</tbody>
</table>
Contribution 2. How to design SS tasks to improve generalizability?

- We investigate three SS tasks:
  - Feature Extractor (GCNs) $f_{θ}(·, ·)$
  - Linear Transformation $Θ_{L}$
  - Linear Transformation $Θ_{as}$

Table 3: Overview of three self-supervised tasks.

<table>
<thead>
<tr>
<th>Task</th>
<th>Relied Feature</th>
<th>Primary Assumption</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clustering</td>
<td>Nodes</td>
<td>Feature Similarity</td>
<td>Classification</td>
</tr>
<tr>
<td>Partitioning</td>
<td>Edges</td>
<td>Connection Density</td>
<td>Regression</td>
</tr>
<tr>
<td>Completion</td>
<td>Nodes &amp; Edges</td>
<td>Context based Representation</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: The overall framework for self-supervision on GCN through multi-task learning. The target task and auxiliary self-supervised tasks share the same feature extractor $f_{θ}(·, ·)$ with their individual linear transformation parameters $Θ, Θ_{as}$.

- We illustrate that different SS tasks benefit generalizability in different cases.
Contribution 3. Does SS boost robustness?

- We generalize SS into adversarial training:

  
  - Adversarial training:

    \[ Z = f_\theta(X, \hat{A})\Theta, \quad Z' = f_\theta(X', A')\Theta, \]

    \[ \theta^*, \Theta^* = \arg\min_{\theta, \Theta} (\mathcal{L}_{\text{sup}}(\theta, \Theta) + \alpha_3 \mathcal{L}_{\text{adv}}(\theta, \Theta)), \quad (6) \]

  - SS + Adversarial training:

    \[ Z = f_\theta(X, \hat{A})\Theta, \quad Z' = f_\theta(X', A')\Theta, \]

    \[ Z_{\text{ss}} = f_\theta(X_{\text{ss}}, A_{\text{ss}}) \]

    \[ \theta^*, \Theta^*, \Theta_{\text{ss}}^* = \arg\min_{\theta, \Theta, \Theta_{\text{ss}}} \left( \alpha_1 \mathcal{L}_{\text{sup}}(\theta, \Theta) \right. \]

    \[ \left. + \alpha_2 \mathcal{L}_{\text{ss}}(\theta, \Theta_{\text{ss}}) + \alpha_3 \mathcal{L}_{\text{adv}}(\theta, \Theta) \right), \quad (7) \]
Contribution 3. Does SS boost robustness?

- We show that SS also improves GCN robustness without requiring larger models or additional data.
  
  - **Clu** is more effective against feature attacks;
  
  - **Par** is more effective against links attacks;
  
- Strikingly, **Comp** significantly boosts robustness against link attacks and link & feature attacks on Cora.

<table>
<thead>
<tr>
<th>Attacks</th>
<th>None</th>
<th>Links</th>
<th>Feats</th>
<th>Links &amp; Feats</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>80.61 ± 0.21</td>
<td>28.72 ± 0.63</td>
<td>44.06 ± 1.23</td>
<td>8.18 ± 0.27</td>
</tr>
<tr>
<td>AdvT</td>
<td>80.24 ± 0.74</td>
<td>54.58 ± 2.57</td>
<td>75.25 ± 1.76</td>
<td>39.08 ± 3.05</td>
</tr>
<tr>
<td>AdvT+Clu</td>
<td>80.26 ± 0.99</td>
<td>55.54 ± 3.19</td>
<td>76.24 ± 0.99</td>
<td>41.84 ± 3.48</td>
</tr>
<tr>
<td>AdvT+Par</td>
<td>80.42 ± 0.76</td>
<td>56.36 ± 2.57</td>
<td>75.88 ± 0.72</td>
<td>41.57 ± 3.47</td>
</tr>
<tr>
<td>AdvT+Comp</td>
<td>79.64 ± 0.99</td>
<td>59.05 ± 3.29</td>
<td>76.04 ± 0.68</td>
<td>47.14 ± 3.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attacks</th>
<th>None</th>
<th>Links</th>
<th>Feats</th>
<th>Links &amp; Feats</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>71.05 ± 0.56</td>
<td>13.68 ± 1.09</td>
<td>22.08 ± 0.73</td>
<td>3.08 ± 0.17</td>
</tr>
<tr>
<td>AdvT</td>
<td>69.98 ± 1.03</td>
<td>39.32 ± 2.39</td>
<td>63.12 ± 0.62</td>
<td>26.20 ± 2.09</td>
</tr>
<tr>
<td>AdvT+Clu</td>
<td>70.13 ± 0.81</td>
<td>40.32 ± 1.73</td>
<td>63.67 ± 0.45</td>
<td>27.02 ± 1.29</td>
</tr>
<tr>
<td>AdvT+Par</td>
<td>69.96 ± 0.77</td>
<td>41.05 ± 1.91</td>
<td>64.06 ± 0.24</td>
<td>28.70 ± 1.60</td>
</tr>
<tr>
<td>AdvT+Comp</td>
<td>69.98 ± 0.82</td>
<td>40.42 ± 2.09</td>
<td>63.50 ± 0.31</td>
<td>27.16 ± 1.69</td>
</tr>
</tbody>
</table>
Conclusion

• We demonstrate the effectiveness of incorporating self-supervised learning in GCNs through multi-task learning;

• We illustrate that appropriately designed multi-task self-supervision tasks benefit GCN generalizability in different cases;

• We show that multi-task self-supervision also improves robustness against attacks, without requiring larger models or additional data.

TAMU HPRC cluster: Terra (GPU); Software: Anaconda/3-5.0.0.1; Typical job: 8G memory, 9 hours
Thank you for listening.

Code: https://github.com/Shen-Lab/SS-GCNs