TEXAS A&M UNIVERSITY
Engineering

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

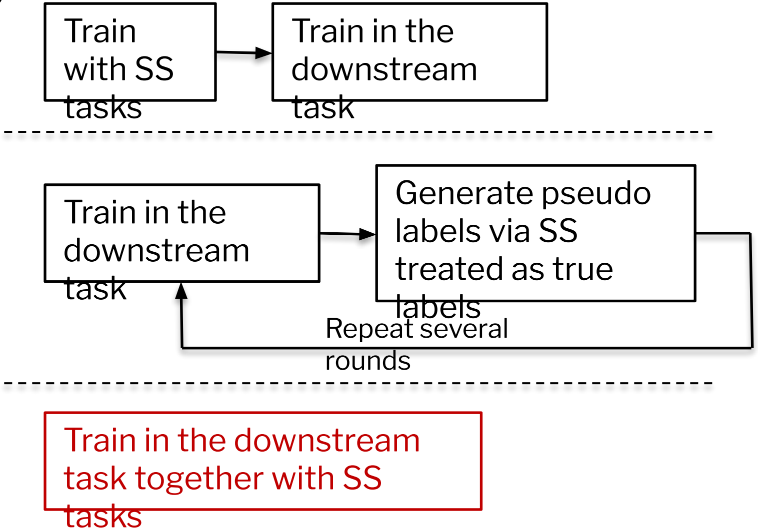
- Motivation
- Contribution 1. How to incorporate self-supervision (SS) in graph convolutional networks (GCNs)?
- Contribution 2. How to design SS tasks to improve model generalizability?
- Contribution 3. Does SS boost model robustness?
- Conclusions

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:

Train in the downstream task together with SS tasks

  – 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 1: Comparing performances of GCN through pretraining & finetuning (P&F) and multi-task learning (MTL) with graph partitioning (see Section 3.3) on the PubMed dataset. Reported numbers correspond to classification accuracy in percent.

Pipeline	GCN	P&F	MTL
Accuracy	79.10 ± 0.21	79.19 ± 0.21	80.00 ± 0.74

Contribution 2. How to design SS tasks to improve generalizability?

- We investigate three SS tasks:

Table 3: Overview of three self-supervised tasks.

Task	Relied Feature	Primary Assumption	Type
Clustering	Nodes	Feature Similarity	Classification
Partitioning	Edges	Connection Density	Classification
Completion	Nodes & Edges	Context based Representation	Regression

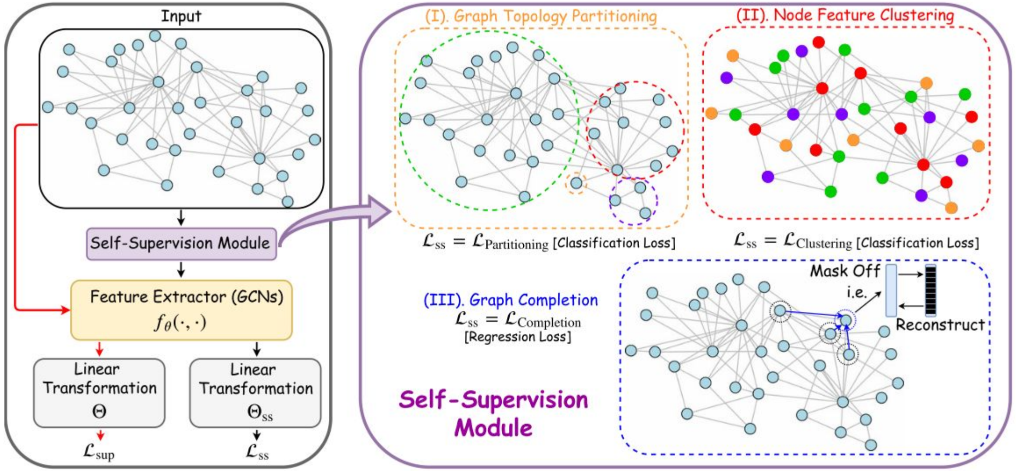


Figure 1: The overall framework for self-supervision on GCN through *multi-task learning*. The target task and auxiliary self-supervised tasks share the same feature extractor $f_\theta(\cdot,\cdot)$ with their individual linear transformation parameters Θ, Θ_{ss}.

- 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:

$$\boldsymbol{Z} = f_\theta(\boldsymbol{X}, \hat{\boldsymbol{A}})\boldsymbol{\Theta}, \quad \boldsymbol{Z}' = f_\theta(\boldsymbol{X}', \boldsymbol{A}')\boldsymbol{\Theta},$$
$$\theta^*, \boldsymbol{\Theta}^* = \arg\min_{\theta, \boldsymbol{\Theta}} \big(\mathcal{L}_{\text{sup}}(\theta, \boldsymbol{\Theta}) + \alpha_3 \mathcal{L}_{\text{adv}}(\theta, \boldsymbol{\Theta})\big), \quad (6)$$

  - SS + Adversarial training:

$$\begin{aligned} \boldsymbol{Z} &= f_\theta(\boldsymbol{X}, \hat{\boldsymbol{A}})\boldsymbol{\Theta}, \quad \boldsymbol{Z}' = f_\theta(\boldsymbol{X}', \boldsymbol{A}')\boldsymbol{\Theta}, \\ \boldsymbol{Z}_{\text{ss}} &= f_\theta(\boldsymbol{X}_{\text{ss}}, \boldsymbol{A}_{\text{ss}}) \\ \theta^*, \boldsymbol{\Theta}^*, \boldsymbol{\Theta}_{\text{ss}}^* &= \arg\min_{\theta, \boldsymbol{\Theta}, \boldsymbol{\Theta}_{\text{ss}}} \big(\alpha_1 \mathcal{L}_{\text{sup}}(\theta, \boldsymbol{\Theta}) \\ &\quad + \alpha_2 \mathcal{L}_{\text{ss}}(\theta, \boldsymbol{\Theta}_{\text{ss}}) + \alpha_3 \mathcal{L}_{\text{adv}}(\theta, \boldsymbol{\Theta})\big), \end{aligned} \quad (7)$$

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 7: Adversarial defense performances on Cora using adversarial training (abbr. AdvT) without or with graph self-supervision. Attacks include those on links, features (abbr. Feats), and both. Red numbers indicate the best two performances in each attack scenario (node classification accuracy; unit: %).

Attacks	None	Links	Feats	Links & Feats
GCN	80.61 ± 0.21	28.72 ± 0.63	44.06 ± 1.23	8.18 ± 0.27
AdvT	80.24 ± 0.74	54.58 ± 2.57	75.25 ± 1.26	39.08 ± 3.05
AdvT+Clu	80.26 ± 0.99	55.54 ± 3.19	76.24 ± 0.99	41.84 ± 3.48
AdvT+Par	80.42 ± 0.76	56.36 ± 2.57	75.88 ± 0.72	41.57 ± 3.47
AdvT+Comp	79.64 ± 0.99	59.05 ± 3.29	76.04 ± 0.68	47.14 ± 3.01

Table 8: Adversarial defense performances on Citeseer using adversarial training without or with graph self-supervision.

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

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

TEXAS A&M UNIVERSITY

Engineering

Thank you for listening.

Paper: https://arxiv.org/abs/2006.09136

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