GRAND+: Scalable Graph Random Neural Networks

Wenzheng Feng[1], Yuxiao Dong[1], Tinglin Huang[1], Ziqi Yin[2], Xu Cheng[13]
Evgeny Kharlamov[4], Jie Tang[1*†]
[1]Tsinghua University, [2]Beijing Institute of Technology
[3]Tencent Inc., [4]Bosch Center for Artificial Intelligence
fwz17@mails.tsinghua.edu.cn,{yuxiaod,tl091,jietang}@tsinghua.edu.cn,ziqiyin18@gmail.com
alexcheng@tencent.com,evgeny.kharlamov@de.bosch.com

WWW 2022
Code: github.com/THUDM/GRAND-plus

2022.06.04 • ChongQing

Leibniz-Institut für Sozialwissenschaften

Reported by Chenghong Li

Introduction

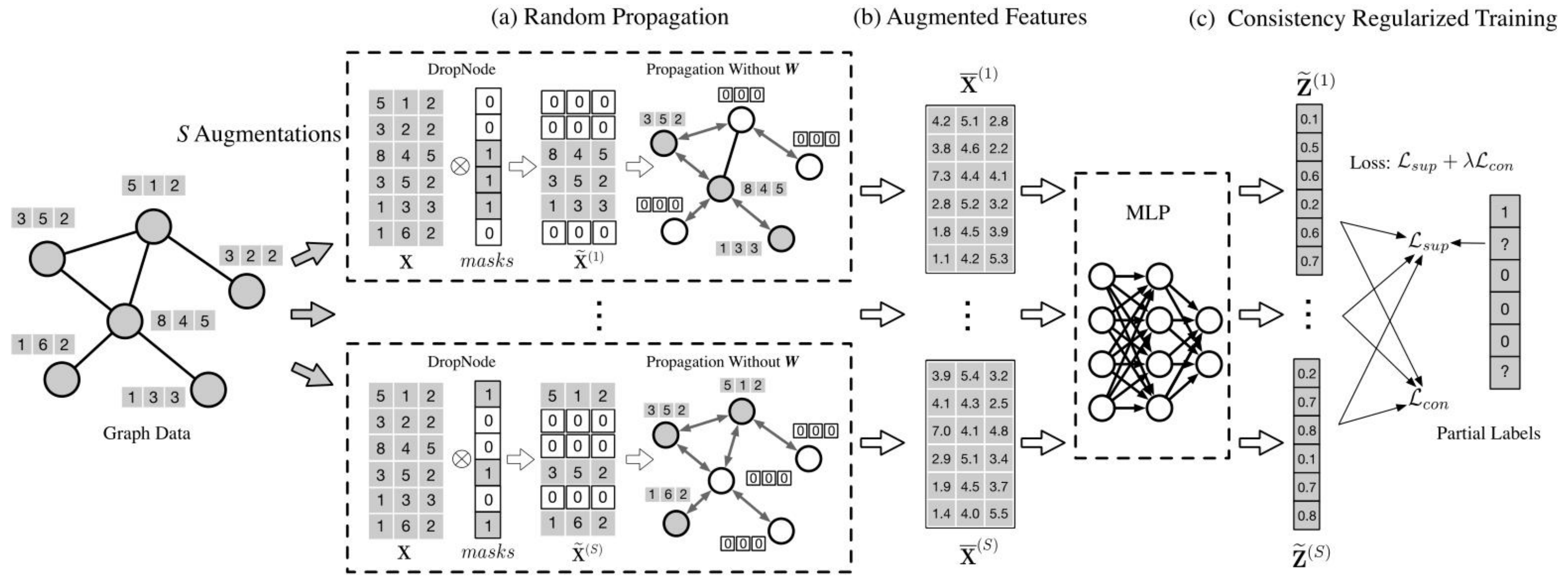


Figure 1: Illustration of GRAND with DropNode as the perturbation method. GRAND designs random propagation (a) to generate multiple graph data augmentations (b), which are further used as consistency regularization (c) for semi-supervised learning.

Graph Convolutional Network.

$$\mathrm{H}^{(l+1)} = \sigma(\hat{\mathrm{A}}\mathrm{H}^{(l)}\mathrm{W}^{(l)}), \tag{1}$$

$$\mathrm{H}^{(0)} = \mathrm{X},\ \hat{\mathrm{A}} = \widetilde{\mathrm{D}}^{-\frac{1}{2}}\widetilde{\mathrm{A}}\widetilde{\mathrm{D}}^{-\frac{1}{2}}$$

Simplified Graph Convolution.

$$\hat{\mathrm{Y}} = \mathrm{softmax}(\hat{\mathrm{A}}^{N}\mathrm{XW}), \tag{2}$$

$$\overline{\mathrm{X}} = \Pi_{\mathrm{sym}}^{\mathrm{avg}} \cdot \mathrm{diag}(\mathbf{z}) \cdot \mathrm{X}, \quad z_i \sim \mathrm{Bernoulli}(1-\delta), \tag{3}$$

$$\frac{1}{M \cdot |U|} \sum_{s \in U} \sum_{m=1}^{M} \left\| \hat{\mathrm{Y}}_s^{(m)} - \overline{\mathrm{Y}}_s \right\|_2^2, \quad \overline{\mathrm{Y}}_s = \sum_{m=1}^{M} \frac{1}{M} \hat{\mathrm{Y}}_s^{(m)}, \tag{4}$$

Method

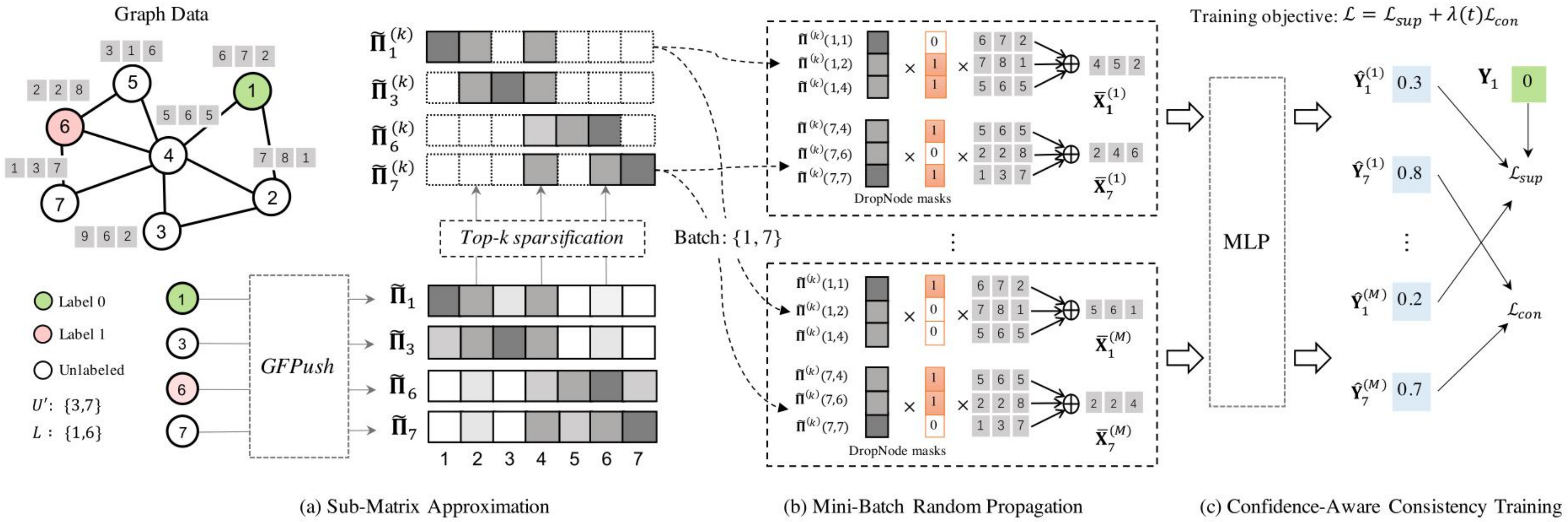
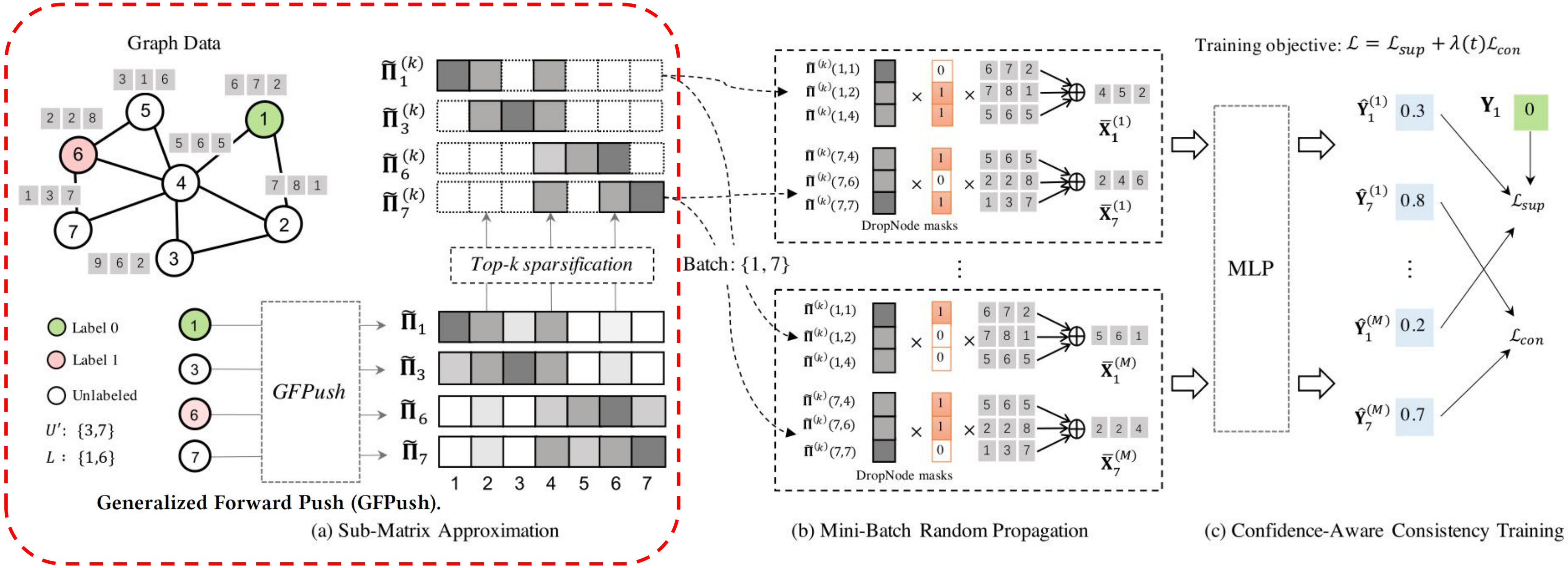


Figure 1: Illustration of GRAND+. (a) GRAND+ adopts *Generalized Forward Push* (*GFPush*) and *Top-k sparsification* to approximate the corresponding rows of propagation matrix Π for nodes in $L \cup U'$. (b) The obtained sparsified row approximations are then used to perform mini-batch random propagation to generate augmentations for nodes in the batch. (c) Finally, the calculated feature augmentations are fed into an MLP to conduct confidence-aware consistency training, which employs both supervised loss $\mathcal{L}_{sup}$ and confidence-aware consistency loss $\mathcal{L}_{con}$ for model optimization.

Method



(a) Sub-Matrix Approximation (b) Mini-Batch Random Propagation (c) Confidence-Aware Consistency Training

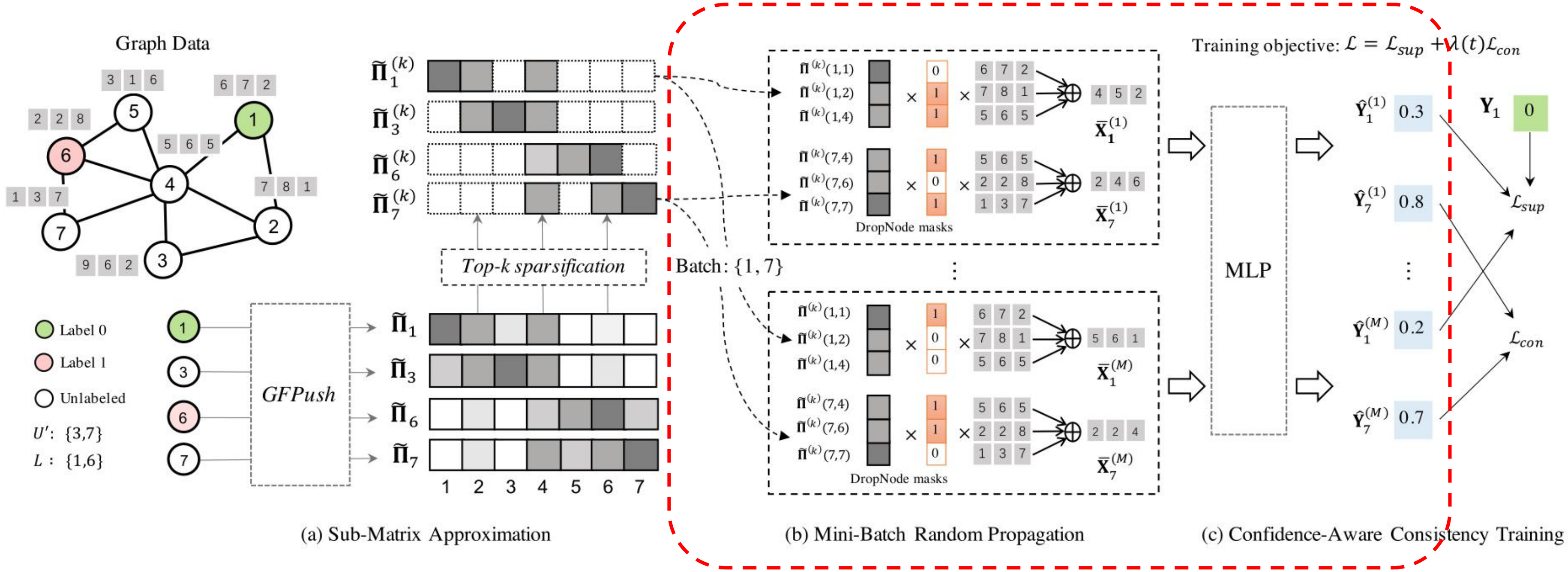
Propagation Matrix.

$$\Pi = \sum_{n=0}^{N} w_n \cdot \mathrm{P}^n, \quad \mathrm{P} = \widetilde{\mathrm{D}}^{-1}\widetilde{\mathrm{A}}, \tag{5}$$

$$\sum_{n=0}^{N} w_n = 1 \text{ and } w_n \geq 0$$

Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local graph partitioning using pagerank vectors. In FOCS'06. IEEE, 475–486.

Method

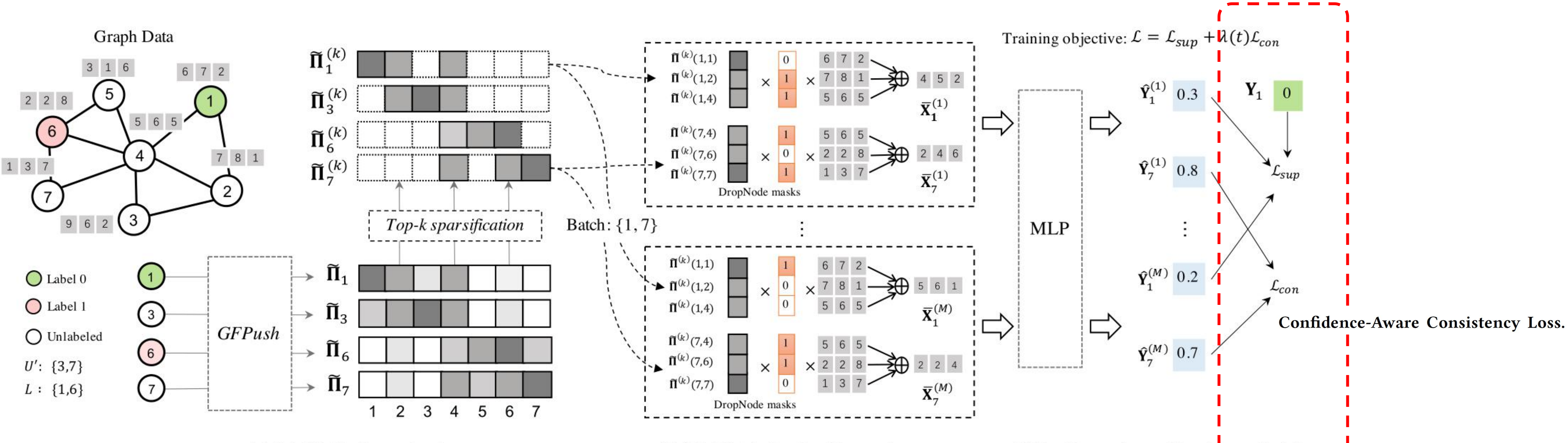


$$\overline{\mathbf{X}}_s = \sum_{v \in \mathcal{N}_s^{(k)}} \mathbf{z}_v \cdot \widetilde{\Pi}^{(k)}(s, v) \cdot \mathbf{X}_v, \quad \mathbf{z}_v \sim \text{Bernoulli}(1-\delta), \tag{7}$$

$$\overline{\mathbf{X}}_s = \sum_{v \in \mathcal{N}_s^{(k)}} \mathbf{z}_v \cdot \widetilde{\Pi}^{(k)}(s, v) \cdot \mathbf{H}_v, \quad \mathbf{H}_v = \mathbf{X}_v \cdot \mathbf{W}^{(0)}, \tag{8}$$

$$\hat{\mathbf{Y}}_s^{(m)} = \text{MLP}(\overline{\mathbf{X}}_s^{(m)}, \Theta), \tag{9}$$

Method



(a) Sub-Matrix Approximation (b) Mini-Batch Random Propagation (c) Confidence-Aware Consistency Training

$$\mathcal{L}_{sup} = -\frac{1}{|L_t| \cdot M} \sum_{s \in L_t} \sum_{m=1}^{M} \mathrm{Y}_s \cdot \log(\hat{\mathrm{Y}}_s^{(m)}). \tag{10}$$

$$\widetilde{\mathrm{Y}}(s, j) = \overline{\mathrm{Y}}(s, j)^{\frac{1}{\tau}} \Big/ \sum_{c=0}^{C-1} \overline{\mathrm{Y}}(s, c)^{\frac{1}{\tau}}, \tag{11}$$

$0 < \tau \leq 1$ is a hyperparameter

$$\overline{\mathrm{Y}}_s = \textstyle\sum_{m=1}^{M} \hat{\mathrm{Y}}_s^{(m)} / M.$$

$$\mathcal{L}_{con} = \frac{1}{|U_t| \cdot M} \sum_{s \in U_t} \mathbb{I}(\max(\overline{\mathrm{Y}}_s) \geq \gamma) \sum_{m=1}^{M} \mathcal{D}(\widetilde{\mathrm{Y}}_s, \hat{\mathrm{Y}}_s^{(m)}), \tag{12}$$

$\mathcal{D}$: L_2 *distance* and *KL divergence.*

$$\mathcal{L} = \mathcal{L}_{sup} + \lambda(t)\mathcal{L}_{con}, \tag{13}$$

$$\hat{\mathrm{Y}}^{(inf)} = \mathrm{MLP}\Big(\sum_{n=0}^{N} w_n (\widetilde{\mathrm{D}}^{-1}\widetilde{\mathrm{A}})^n \cdot (1 - \delta) \cdot \mathrm{X}, \Theta\Big), \tag{14}$$

Experiments

Table 1: Dataset statistics.

Dataset	Nodes	Edges	Classes	Features
Cora	2,708	5,429	7	1,433
Citeseer	3,327	4,732	6	3,703
Pubmed	19,717	44,338	3	500
AMiner-CS	593,486	6,217,004	18	100
Reddit	232,965	11,606,919	41	602
Amazon2M	2,449,029	61,859,140	47	100
MAG-Scholar-C	10,541,560	265,219,994	8	2,784,240

Experiments

Table 2: Classification Accuracy (%) on Benchmarks.

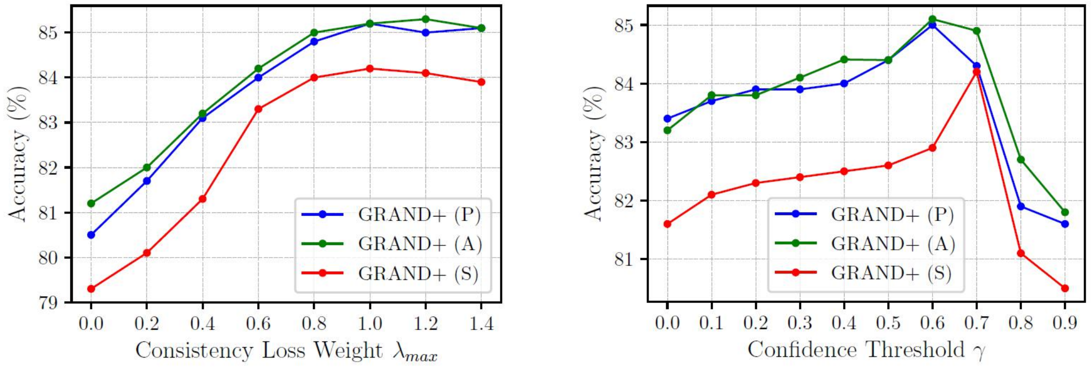
Category	Method	Cora	Citeseer	Pubmed
Full-batch GNNs	GCN	81.5 ± 0.6	71.3 ± 0.4	79.1 ± 0.4
	GAT	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3
	APPNP	84.1 ± 0.3	71.6 ± 0.5	79.7 ± 0.3
	GCNII	85.5 ± 0.5	73.4 ± 0.6	80.3 ± 0.4
	GRAND	85.4 ± 0.4	75.4 ± 0.4	82.7 ± 0.6
Scalable GNNs	FastGCN	81.4 ± 0.5	68.8 ± 0.9	77.6 ± 0.5
	GraphSAINT	81.3 ± 0.4	70.5 ± 0.4	78.2 ± 0.8
	SGC	81.0 ± 0.1	71.8 ± 0.1	79.0 ± 0.1
	GBP	83.9 ± 0.7	72.9 ± 0.5	80.6 ± 0.4
	PPRGo	82.4 ± 0.2	71.3 ± 0.3	80.0 ± 0.4
Our Methods	GRAND+ (P)	**85.8 ± 0.4**	**75.6 ± 0.4**	84.5 ± 1.1
	GRAND+ (A)	85.5 ± 0.4	75.5 ± 0.4	**85.0 ± 0.6**
	GRAND+ (S)	85.0 ± 0.5	74.4 ± 0.5	84.2 ± 0.6

Table 3: Accuracy (%) and Running Time (s) on Large Graphs.

Method	AMiner-CS		Reddit		Amazon2M		MAG.	
	Acc	RT	Acc	RT	Acc	RT	Acc	RT
GRAND	53.1±1.1	750	OOM	–	OOM	–	OOM	–
FastGCN	48.9±1.6	69	89.6±0.6	158	72.9±1.0	239	64.3±5.6	4220
GraphSAINT	51.8±1.3	39	92.1±0.5	39	75.9±1.3	189	75.0±1.7	6009
SGC	50.2±1.2	9	92.5±0.2	31	74.9±0.5	69	–	>24h
GBP	52.7±1.7	21	88.7±1.1	370	70.1±0.9	280	–	>24h
PPRGo	51.2±1.4	11	91.3±0.2	233	67.6±0.5	160	72.9±1.1	434
GRAND+ (P)	53.9±1.8	17	93.3±0.2	183	75.6±0.7	188	77.6±1.2	653
GRAND+ (A)	54.2±1.7	14	**93.5±0.2**	174	75.9±0.7	136	**80.0±1.1**	737
GRAND+ (S)	**54.2±1.6**	10	92.8±0.2	62	**76.2±0.6**	80	77.8±0.9	483

- GRAND+ (P): *Truncated ppr matrix* $\mathbf{\Pi}^{\text{ppr}} = \sum_{n=0}^{N} \alpha(1-\alpha)^n \mathbf{P}^n$.
- GRAND+ (A): *Average pooling matrix* $\mathbf{\Pi}^{\text{avg}} = \sum_{n=0}^{N} \mathbf{P}^n/(N+1)$.
- GRAND+ (S): *Single order matrix* $\mathbf{\Pi}^{\text{single}} = \mathbf{P}^N$.

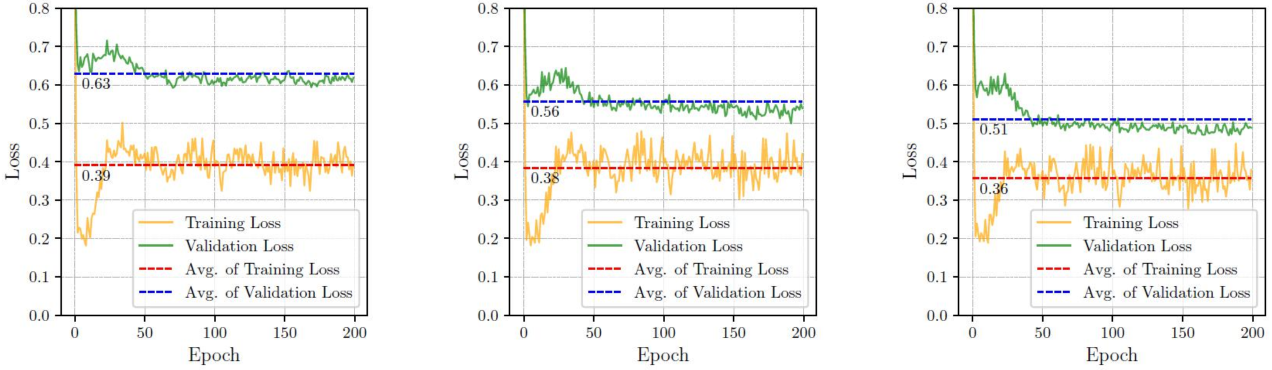
Experiments



(a) Accuracy w.r.t. λ_{max}.

(b) Accuracy w.r.t. γ

Figure 2: Effects of λ_{max} and γ on Pubmed.



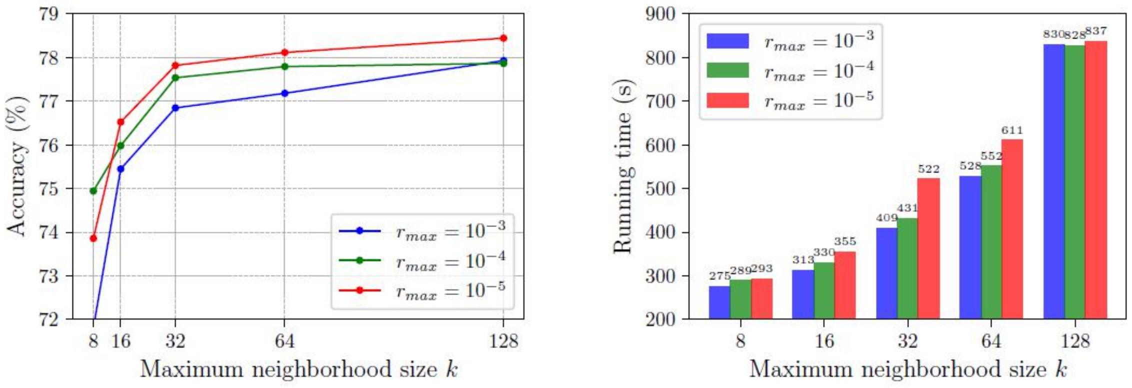
(a) $\lambda_{max} = 0.0$

(b) $\lambda_{max} = 1.0, \gamma = 0.0$

(c) $\lambda_{max} = 1.0, \gamma = 0.6$

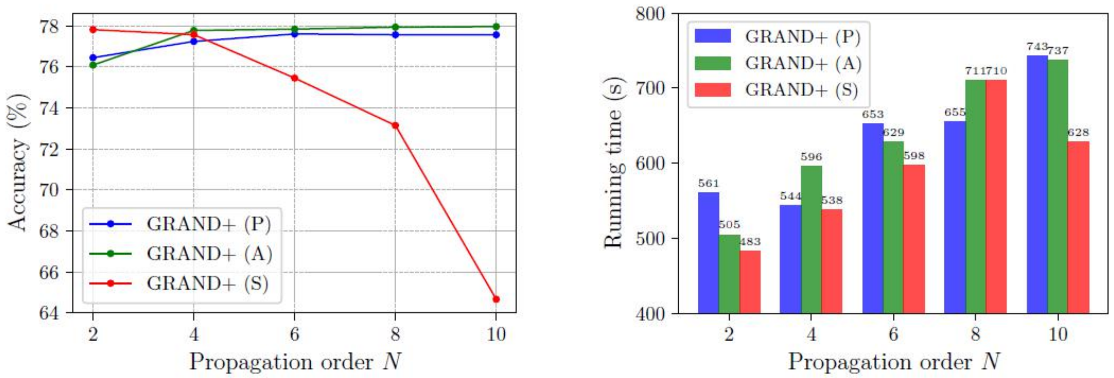
Figure 3: Training and Validation Losses on Pubmed.

Experiments



(a) Classification accuracy. (b) Running time.

Figure 4: GRAND+ w.r.t. k and r_{max} on MAG-Scholar-C.



(a) Classification accuracy. (b) Running time.

Figure 5: Effects of propagation order N on MAG-Scholar-C.

Thanks