

Supplementary Materials of GraphLearner: Graph Node Clustering with Fully Learnable Augmentation

Anonymous Authors

1 DATASETS DESCRIPTION

In this work, we utilize six widely used graph benchmark, i.e., CORA, CITESEER, AMAP, UAT, EAT, BAT. Here are the basic information about these datasets.

- The CORA dataset, comprised of 2708 scientific publications, is categorized into seven distinct classes. The dataset is intricately connected through a citation network that consists of 5429 interlinks. Each publication within the dataset is characterized by a binary word vector, which indicates the presence or absence of the corresponding word from a dictionary, which itself comprises 1433 unique words.
- CITESEER dataset, another citation network, includes 4732 links and 3327 scientific publications, segmented into six categories. Each publication in the dataset is represented by a binary word vector, indicating the presence or absence of the corresponding word in a dictionary, encompassing 3703 unique words.
- AMAP dataset, a co-purchasing network extracted from Amazon, represents products as nodes. The features of these nodes are encoded through bag-of-words reviews. Edges within this network signify the co-purchasing frequency of two products.
- UAT dataset, consisting of traffic data collected from the Bureau of Transportation Statistics from January to October 2016, contains 1190 nodes and 13599 edges.
- BAT dataset, a collection of airport data gathered from the National Civil Aviation Agency (ANAC) from January to December 2016, contains 131 nodes and 1038 edges.
- EAT dataset, an assembly of airport data collected from the European Union’s Statistics Office from January to November 2016, includes 399 nodes and 5995 edges.

2 DETAILS OF THE PROPOSED METHOD

In this section, we introduce the detailed implementation of our method with PyTorch-style pseudo codes in Algorithm 1.

3 ADDITIONAL EXPERIMENTS

3.1 Additional Comparison Experiments

Due to the limitation of the pages, in this section, we have conducted additional experiments to further the superiority of our proposed GraphLearner. Specifically, two categories methods are compared in this section, i.e. deep clustering methods (DCN [10], DEC [9], AdaGAE [5]), and deep graph clustering methods (DFCN [6], GDCL [11]). The experiment results are shown in Table.2. We could observe as follows.

- Deep clustering methods are not comparable with our proposed methods. We conjecture that those methods overlook the graph structure.

Algorithm 1 PyTorch-style Pseudo Code of Our Method.

```
# X: Original Attribute
# A: Original Structure
# AG: Attribute Augmentor
# SG: Structure Augmentor
# P: High-confidence Pseudo Labels
# sim: Similarity Function
# simclr: simclr loss
# alpha: trade-off parameter
for epoch in range(epoch_num):
    # Attribute Matrix and Adjacency Matrix
    Aug_X = AG(X)
    Aug_S = SG(A)

    # Net Encoding
    F1 = F.normalization((X, A), dim=1, p=2)
    F2 = F.normalization((Gen_X, Gen_A), dim=1, p=2)
    # Clustering and High-confidence Pseudo Label
    clu_res, P = clustering((F1+F2/2))

    # Cross-view Similarity Matrix
    M = F1 @ F2.T
    # Pseudo Label Matrix
    Q = (P==P.T).int()

    loss_c = simclr(F1, F2)
    loss = loss_c

    # Structure Refine
    Gen_A = Aug_S * M
    Gen_A = Aug_S * Q

    loss_a = -(MSE(X, Aug_X) + MSE(A, Aug_S))
    loss = loss_c + alpha * loss_a
    # optimization
    loss.backward()
    optimizer.step()
    clu_res = clustering((F1+F2/2))
    return clu_res
```

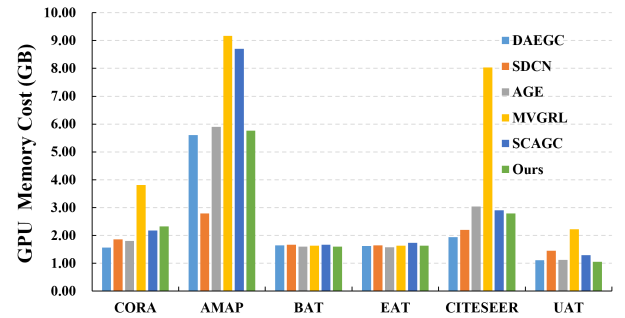


Figure 1: GPU memory costs on six datasets with five methods.

- GraphLearner could achieve better performance than deep graph clustering methods. The reason is that contrastive learning enhances the supervision information capture capability of our method.

Table 1: The hyper-parameter of the GraphLearner on six datasets.

Hyper-parameter	CORA	CITeseer	AMAP	BAT	EAT	UAT
Learning Rate	1e-5	1e-4	1e-5	1e-2	1e-3	1e-2
τ	0.5	0.8	0.8	0.5	0.5	0.6
α	0.5	0.5	0.5	0.5	0.5	0.5

Table 2: Additional comparison experiments on four benchmark datasets. The clustering performance is evaluated by four metrics with mean value and standard deviation.

Dataset	Metric	Deep Clustering			Deep Graph Clustering		GraphLearner Ours
		DCN	DEC	AdaGAE	DFCN	GDCL	
		ICML 2017	ICML 2016	TPAMI 2021	AAAI 2021	IJCAI 2021	
CORA	ACC	49.38±0.91	46.50±0.26	50.06±1.58	36.33±0.49	70.83±0.47	74.91±1.78
	NMI	25.65±0.65	23.54±0.34	32.19±1.34	19.36±0.87	56.60±0.36	58.16±0.83
	ARI	21.63±0.58	15.13±0.42	28.25±0.98	04.67±2.10	48.05±0.72	53.82±2.25
	F1	43.71±1.05	39.23±0.17	53.53±1.24	26.16±0.50	52.88±0.97	73.33±1.86
AMAP	ACC	48.25±0.08	47.22±0.08	67.70±0.54	76.82±0.23	43.75±0.78	77.24±0.87
	NMI	38.76±0.30	37.35±0.05	55.96±0.87	66.23±1.21	37.32±0.28	67.12±0.92
	ARI	20.80±0.47	18.59±0.04	46.20±0.45	58.28±0.74	21.57±0.51	58.14±0.82
	F1	47.87±0.20	46.71±0.12	62.95±0.74	71.25±0.31	38.37±0.29	52.77±2.61
BAT	ACC	47.79±3.95	42.09±2.21	43.51±0.48	55.73±0.06	45.42±0.54	75.50±0.87
	NMI	18.03±7.73	14.10±1.99	15.84±0.78	48.77±0.51	31.70±0.42	50.58±0.90
	ARI	13.75±6.05	07.99±1.21	07.80±0.41	37.76±0.23	19.33±0.57	47.45±1.53
	F1	47.87±0.20	46.71±0.12	62.95±0.74	71.25±0.31	38.37±0.29	75.40±0.88
UAT	ACC	46.82±1.14	45.61±1.84	52.10±0.87	33.61±0.09	48.70±0.06	55.31±2.42
	NMI	17.18±1.60	16.63±2.39	26.02±0.71	26.49±0.41	25.10±0.01	24.40±1.69
	ARI	13.59±2.02	13.14±1.97	24.47±0.13	11.87±0.23	21.76±0.01	22.14±1.67
	F1	47.87±0.20	46.71±0.12	62.95±0.74	71.25±0.31	38.37±0.29	52.77±2.61
CITeseer	ACC	57.08±0.13	55.89±0.20	54.01±1.11	69.50±0.20	66.39±0.65	70.12±0.36
	NMI	27.64±0.08	28.34±0.30	27.79±0.47	43.90±0.20	39.52±0.38	43.56±0.35
	ARI	29.31±0.14	28.12±0.36	24.19±0.85	45.50±0.30	41.07±0.96	44.85±0.69
	F1	53.80±0.11	52.62±0.17	51.11±0.64	64.30±0.20	61.12±0.70	65.01±0.39
EAT	ACC	38.85±2.32	36.47±1.60	32.83±1.24	49.37±0.19	33.46±0.18	57.22±0.73
	NMI	06.92±2.80	04.96±1.74	04.36±1.87	32.90±0.41	13.22±0.33	33.47±0.34
	ARI	05.11±2.65	03.60±1.87	02.47±0.54	23.25±0.18	04.31±0.29	26.21±0.81
	F1	38.75±2.25	34.84±1.28	32.39±0.47	42.95±0.04	25.02±0.21	57.53±0.67

3.2 Memory Cost

In this subsection, we conduct experiments to test GPU memory costs of our proposed GraphLearner. We compare with five methods (i.e., DAEGC [7], SDCN [1], AGE [2], MVGRL [4], SCAGC [8]) on six datasets. From the results in Fig. 1, we observe that the memory costs of our GraphLearner are also comparable with other algorithms.

3.3 Effectiveness of the Similarity and Pseudo-label Matrix Refinement strategies

In this subsection, we implement experiments to verify the effectiveness of our refinement strategies, i.e., similarity matrix optimization and pseudo-label matrix optimization. Here, we adopt the model without any optimization strategy as the baseline. For simplicity, we denote “NP+NS”, “NP”, “NS”, and “P+S” as the baseline, baseline

with similarity matrix optimization, baseline with pseudo-label matrix optimization, and ours, respectively. From the results in Fig. 2, we observe that the performance of the GraphLearner will decrease when any one of the aforementioned components is dropped. Overall, extensive experiments could demonstrate the effectiveness of our optimization strategies.

3.4 Sensitivity Analysis of Hyper-parameter Threshold τ

To investigate the influence of the hyper-parameter threshold τ , we conduct the experiments on four datasets as shown in Fig. 3. From the results, we observe that the model obtains promising performance with the τ increasing. The reason is that the pseudo labels are more reliable with a high threshold.

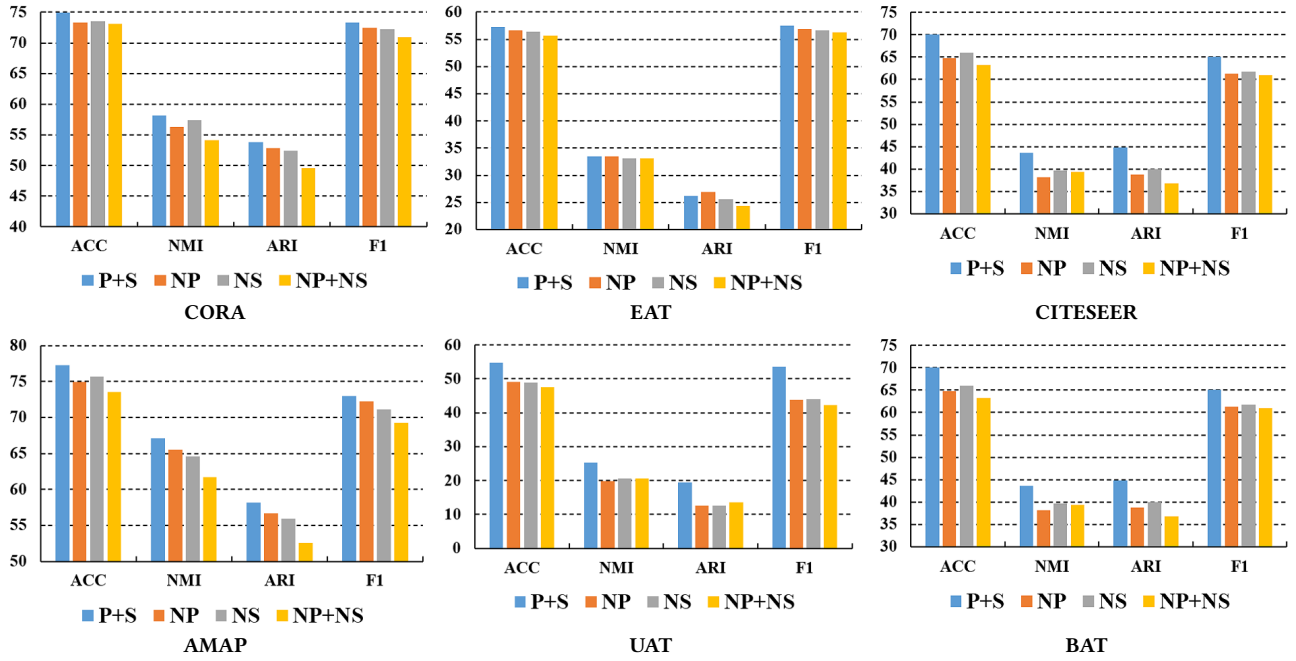


Figure 2: Ablation studies over the effectiveness of the proposed similarity matrix and pseudo labels matrix refinement strategy on six benchmark datasets. “NP+NS”, “S”, “P”, and “P+S” denotes the baseline, baseline with similarity matrix optimization, baseline with pseudo-label matrix optimization, and ours, respectively.

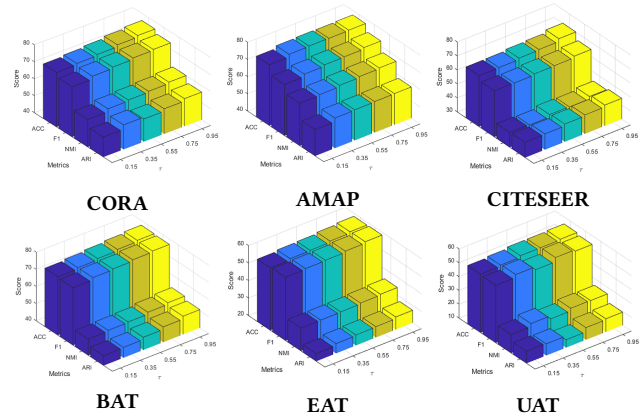


Figure 3: Sensitivity analysis of the hyper-parameter τ .

3.5 Changing the selection of k in KMeans

In this subsection, we discuss the selection of k in Kmeans [3]. Specifically, we conduct the experiments on the CORA dataset with our proposed GraphLearner. CORA consists of 7 classes. We change the class of CORA from 3 to 8. From the Table.3, we could observe that with the variable of the k , the clustering performance is comparable on CORA dataset. The experimental results demonstrate that the selection of k is not an important influencing factor in our GraphLearner. It will be a interesting problem to explore the

unknown number of the clusters. We will explore this problem in the future.

Table 3: Changing the selection of k in KMeans.

K	NMI	ARI
3	41.90±2.69	33.07±2.89
4	47.06±1.91	37.85±1.98
5	52.05±1.49	43.63±1.60
6	55.00±1.27	50.06±2.35
7	58.16±0.83	53.82±2.25
8	56.32±0.74	52.44±0.45

REFERENCES

- [1] Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. 2020. Structural deep clustering network. In *Proceedings of The Web Conference 2020*. 1400–1410.
- [2] Ganqu Cui, Jie Zhou, Cheng Yang, and Zhiyuan Liu. 2020. Adaptive graph encoder for attributed graph embedding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 976–985.
- [3] John A Hartigan and Manchek A Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)* 28, 1 (1979), 100–108.
- [4] Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*. PMLR, 4116–4126.
- [5] Xuelong Li, Hongyuan Zhang, and Rui Zhang. 2021. Adaptive graph auto-encoder for general data clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).
- [6] Wenxuan Tu, Sihang Zhou, Xinwang Liu, Xifeng Guo, Zhiping Cai, Jieren Cheng, et al. 2020. Deep Fusion Clustering Network. *arXiv preprint arXiv:2012.09600* (2020).
- [7] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Attributed graph clustering: A deep attentional embedding approach. *arXiv preprint arXiv:1906.06532* (2019).
- [8] Wei Xia, Qianqian Wang, Quanxue Gao, Ming Yang, and Xinbo Gao. 2022. Self-consistent Contrastive Attributed Graph Clustering with Pseudo-label Prompt. *IEEE Transactions on Multimedia* (2022). <https://doi.org/10.1109/TMM.2022.3213208>
- [9] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*. PMLR, 478–487.
- [10] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *international conference on machine learning*. PMLR, 3861–3870.
- [11] Han Zhao, Xu Yang, Zhenru Wang, Erkun Yang, and Cheng Deng. 2021. Graph debiased contrastive learning with joint representation clustering. In *Proc. IJCAI*. 3434–3440.