Augmentation-Free Self-Supervised Learning on Graphs

Namkyeong Lee[1], Junseok Lee[1], Chanyoung Park[1,2*]

[1] Dept. of Industrial and Systems Engineering, KAIST, Daejeon, Republic of Korea
[2] Graduate School of Artificial Intelligence, KAIST, Daejeon, Republic of Korea
namkyeong96@kaist.ac.kr, junseoklee@kaist.ac.kr, cy.park@kaist.ac.kr

Code:http

Code : https://github.com/Namkyeong/AFGRL

AAAI_2022

Leibniz-Institut
für Sozialwissenschaften

Reported by Xinsheng Wang

1.Introduction

2.Method

3.Experiments

Introduction

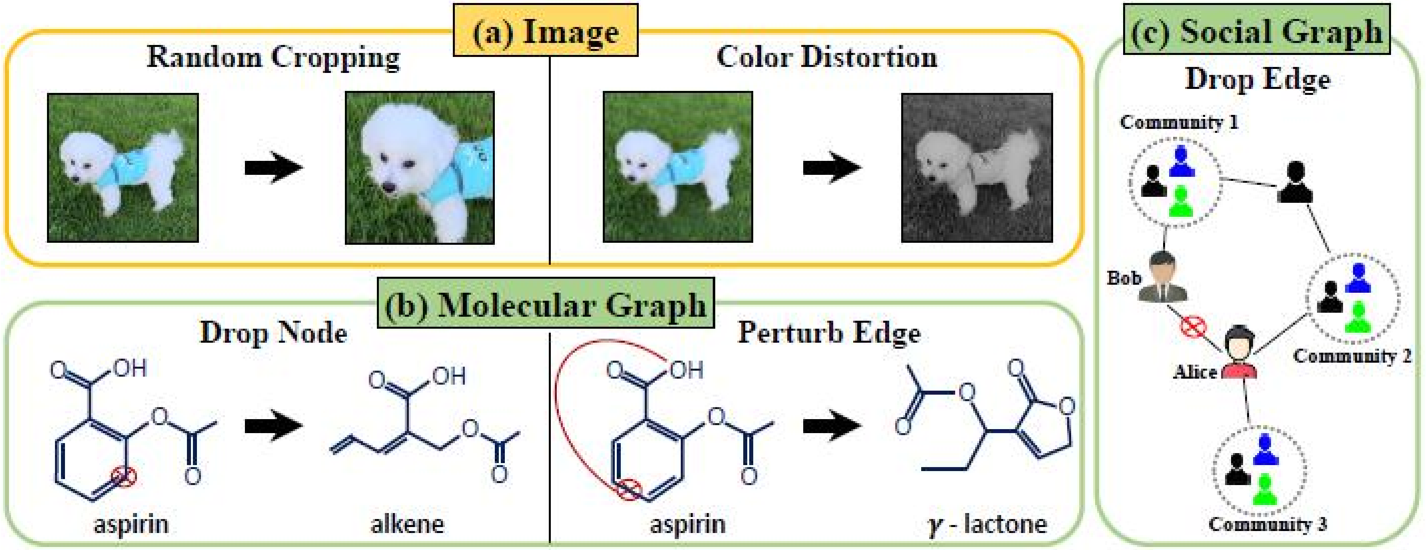


Figure 1: Augmentations on images ((a)) keep the underlying semantics, whereas augmentations on graphs ((b),(c)) may unexpectedly change the semantics.

Method

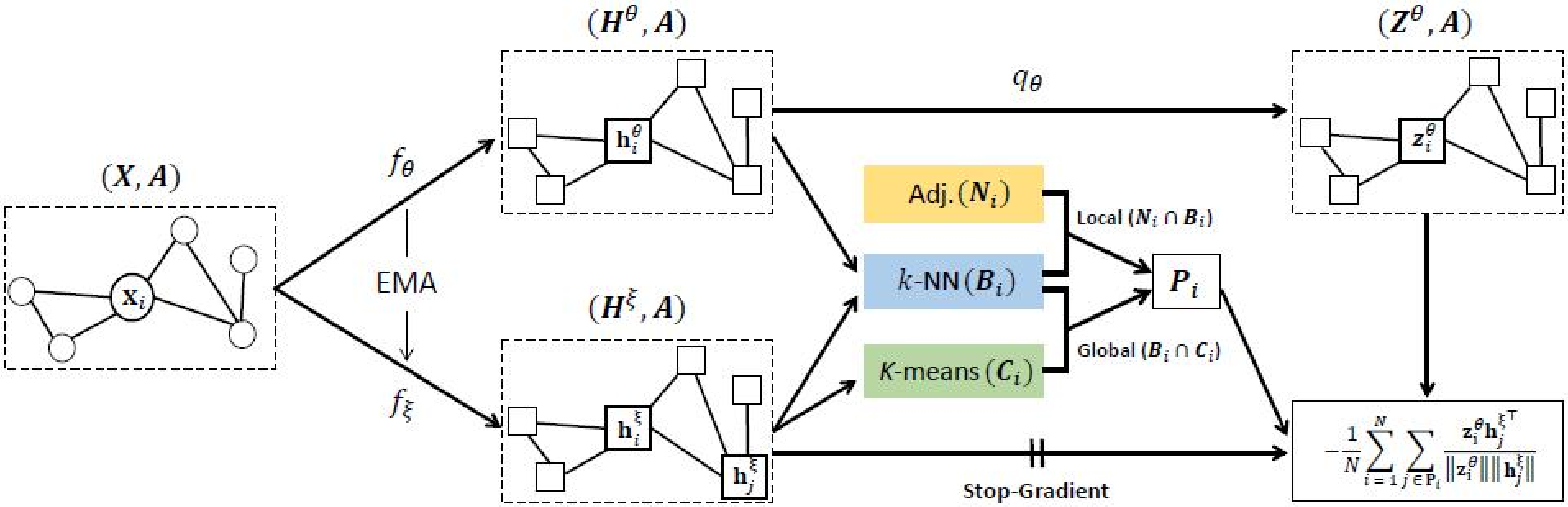
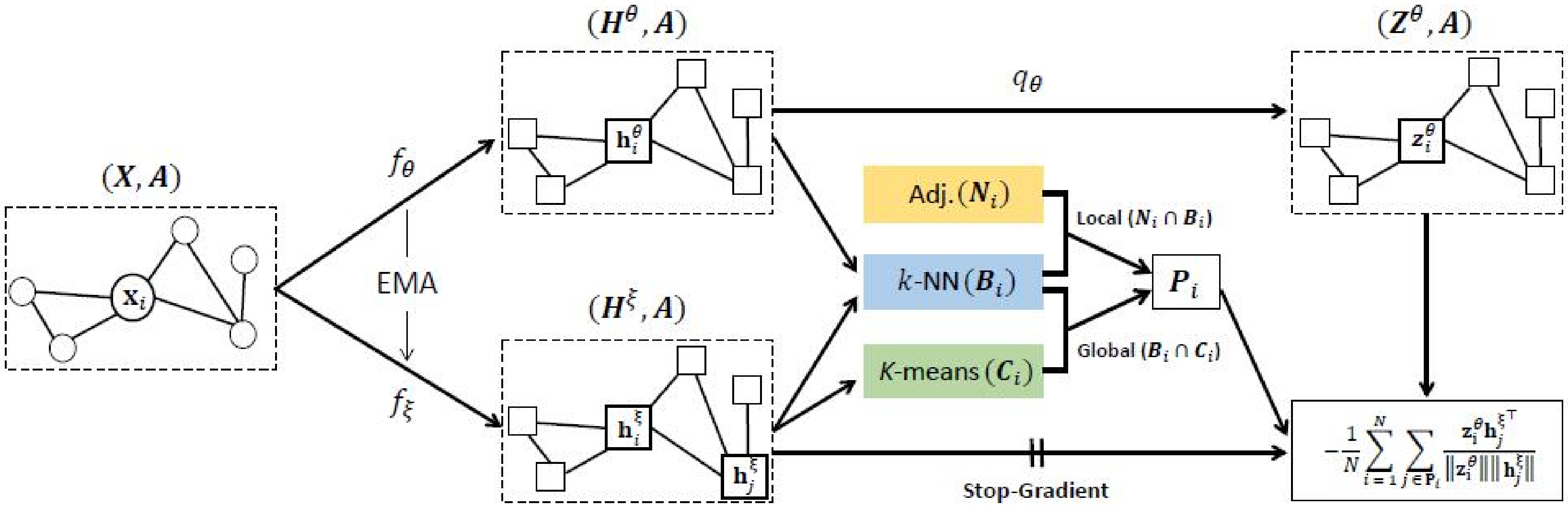
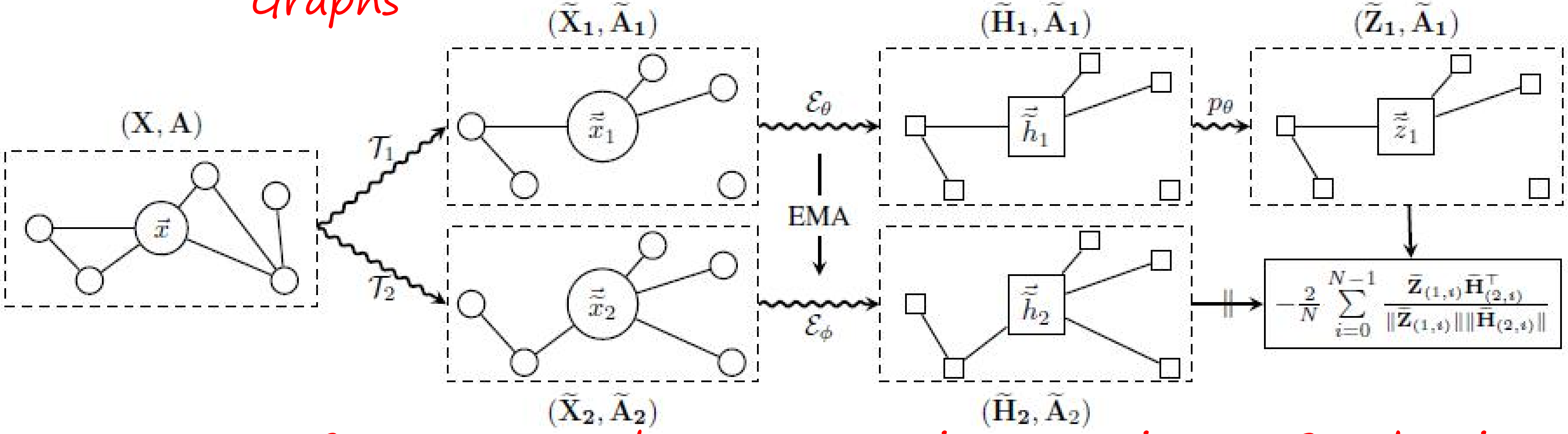


Figure 2: The overall architecture of AFGRL. Given a graph, f_θ and f_ξ generate node embeddings $\mathbf{H}^\theta$ and $\mathbf{H}^\xi$ both of which are used to obtain k-NNs for node v_i, i.e., $\mathbf{B}_i$. Combining it with $\mathbf{N}_i$, we obtain local positives, i.e., $\mathbf{B}_i \cap \mathbf{N}_i$. To obtain global positives for node v_i, K-means clustering is performed on $\mathbf{H}^\xi$, and the result $\mathbf{C}_i$ is combined with $\mathbf{B}_i$, i.e., $\mathbf{B}_i \cap \mathbf{C}_i$. Finally, we combine local and global positives to obtain real positives, i.e., $\mathbf{P}_i$. A predictor q_θ projects $\mathbf{H}^\theta$ to $\mathbf{Z}^\theta$, which is used to compute the final loss along with $\mathbf{H}^\xi$. Note that f_θ is updated via gradient descent of the loss, whereas f_ξ is updated via EMA of f_θ.

Method



2022_AAAI_Augmentation-Free Self-Supervised Learning on Graphs



2021_ICLR_Large-Scale Representation Learning on Graphs via

Method

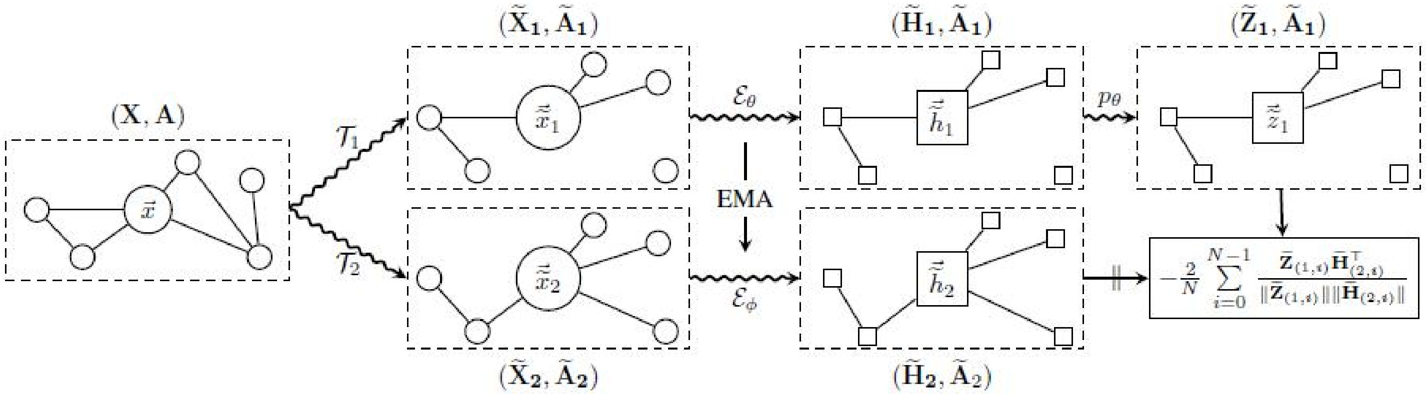


Figure 1: Overview of our proposed BGRL method. The original graph is first used to derive two different semantically similar views using augmentations $\mathcal{T}_{1,2}$. From these, we use encoders $\mathcal{E}_{\theta,\phi}$ to form online and target node embeddings. The predictor p_θ uses the online embedding $\widetilde{\mathbf{H}}_1$ to form a prediction $\widetilde{\mathbf{Z}}_1$ of the target embedding $\widetilde{\mathbf{H}}_2$. The final objective is then computed as the cosine similarity between $\widetilde{\mathbf{Z}}_1$ and $\widetilde{\mathbf{H}}_2$, flowing gradients only through $\widetilde{\mathbf{Z}}_1$. The target parameters ϕ are updated as an exponentially moving average of θ.

$$\mathbf{G}_1 = (\widetilde{\mathbf{X}}_1, \widetilde{\mathbf{A}}_1) \text{ and } \mathbf{G}_2 = (\widetilde{\mathbf{X}}_2, \widetilde{\mathbf{A}}_2)$$

node feature masking and **edge masking**

$$\widetilde{\mathbf{H}}_1 := \mathcal{E}_\theta(\widetilde{\mathbf{X}}_1, \widetilde{\mathbf{A}}_1) \qquad \widetilde{\mathbf{H}}_2 := \mathcal{E}_\phi(\widetilde{\mathbf{X}}_2, \widetilde{\mathbf{A}}_2)$$

$$\widetilde{\mathbf{Z}}_1 := p_\theta(\widetilde{\mathbf{H}}_1)$$

$$\ell(\theta, \phi) = -\frac{2}{N} \sum_{i=0}^{N-1} \frac{\widetilde{\mathbf{Z}}_{(1,i)} \widetilde{\mathbf{H}}_{(2,i)}^\top}{\|\widetilde{\mathbf{Z}}_{(1,i)}\| \|\widetilde{\mathbf{H}}_{(2,i)}\|} \tag{1}$$

$$\theta \leftarrow \text{optimize}(\theta,\ \eta,\ \partial_\theta \ell(\theta, \phi)), \tag{2}$$

$$\phi \leftarrow \tau\phi + (1-\tau)\theta, \tag{3}$$

EMA(Exponential Moving Average)

2020_NIPS_Bootstrap your own latent: A new approach to self-supervised Lea

Method

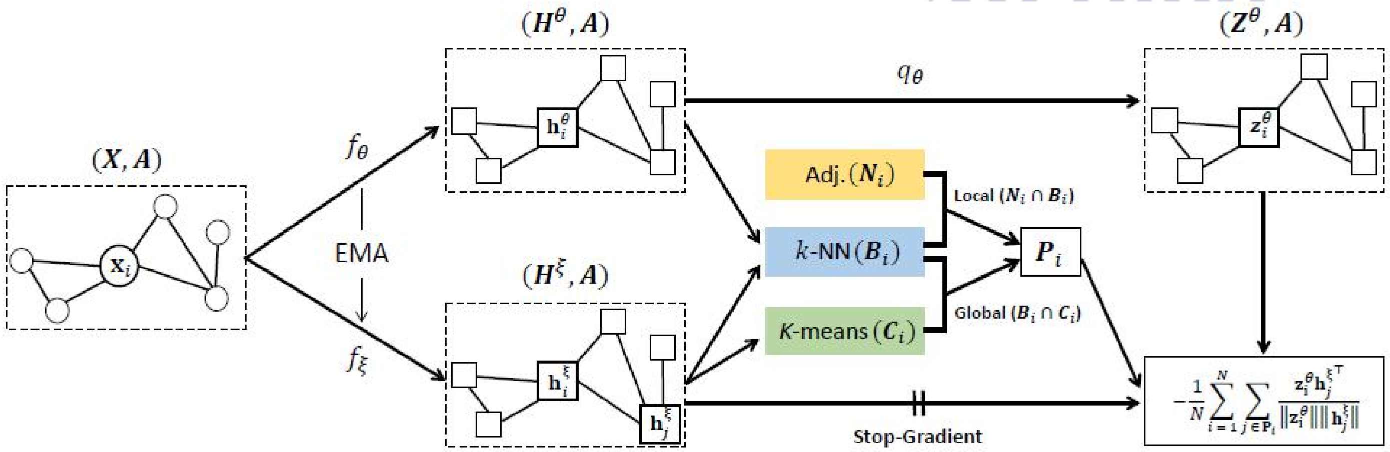


Figure 2: The overall architecture of AFGRL. Given a graph, f_θ and f_ξ generate node embeddings $\mathbf{H}^\theta$ and $\mathbf{H}^\xi$ both of which are used to obtain k-NNs for node v_i, i.e., $\mathbf{B}_i$. Combining it with $\mathbf{N}_i$, we obtain local positives, i.e., $\mathbf{B}_i \cap \mathbf{N}_i$. To obtain global positives for node v_i, K-means clustering is performed on $\mathbf{H}^\xi$, and the result $\mathbf{C}_i$ is combined with $\mathbf{B}_i$, i.e., $\mathbf{B}_i \cap \mathbf{C}_i$. Finally, we combine local and global positives to obtain real positives, i.e., $\mathbf{P}_i$. A predictor q_θ projects $\mathbf{H}^\theta$ to $\mathbf{Z}^\theta$, which is used to compute the final loss along with $\mathbf{H}^\xi$. Note that f_θ is updated via gradient descent of the loss, whereas f_ξ is updated via EMA of f_θ.

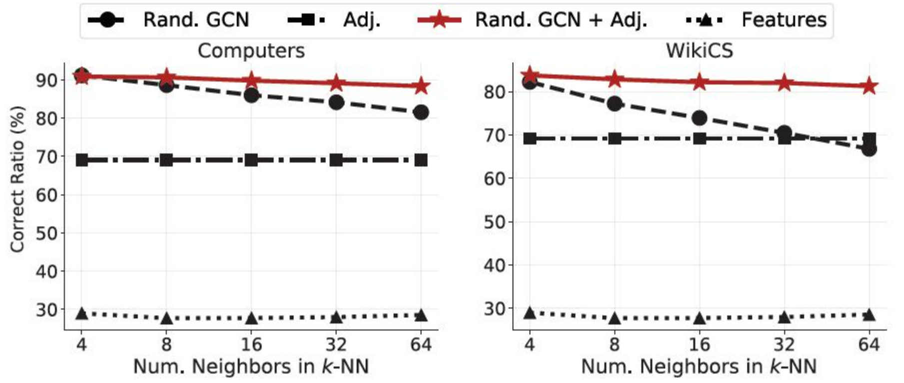


Figure 3: Analysis on the ratio of its neighboring nodes being the same label as the query node across different ks.

$$\mathbf{H}^\theta = f_\theta(\mathbf{X}, \mathbf{A}) \quad \text{online encoder } f_\theta(\cdot)$$

$$\mathbf{H}^\xi = f_\xi(\mathbf{X}, \mathbf{A}) \quad \text{target encoder } f_\xi(\cdot)$$

$$sim(v_i, v_j) = \frac{\mathbf{h}_i^\theta \cdot \mathbf{h}_j^\xi}{\|\mathbf{h}_i^\theta\|\|\mathbf{h}_j^\xi\|}, \forall v_j \in \mathcal{V} \qquad (1)$$

Capturing Local Structural Information

$$\mathbf{H}_{\text{Rand-GCN}} = \text{Rand-GCN}(\mathbf{X}, \mathbf{A})$$

Rand.GCN + Adj

$$\mathbf{B}_i \cap \mathbf{N}_i$$

```
locality = knn_neighbor * adj
```

Capturing Global Semantics

K-means

$$\mathbf{C}_i = \{v_j | v_j \in G_{c(\mathbf{h}_i^\xi)}\}$$

$$\mathbf{B}_i \cap \mathbf{C}_i$$

$$\mathbf{P}_i = (\mathbf{B}_i \cap \mathbf{N}_i) \cup (\mathbf{B}_i \cap \mathbf{C}_i) \qquad (2)$$

$$\mathcal{L}_{\theta,\xi} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{v_j \in \mathbf{P}_i}\frac{\mathbf{z}_i^\theta \mathbf{h}_j^{\xi\top}}{\|\mathbf{z}_i^\theta\|\|\mathbf{h}_j^\xi\|}, \qquad (3)$$

Experiments

	WikiCS	Computers	Photo	Co.CS	Co.Physics
Sup. GCN	77.19 ± 0.12	86.51 ± 0.54	92.42 ± 0.22	93.03 ± 0.31	95.65 ± 0.16
Raw feats.	71.98 ± 0.00	73.81 ± 0.00	78.53 ± 0.00	90.37 ± 0.00	93.58 ± 0.00
node2vec	71.79 ± 0.05	84.39 ± 0.08	89.67 ± 0.12	85.08 ± 0.03	91.19 ± 0.04
DeepWalk	74.35 ± 0.06	85.68 ± 0.06	89.44 ± 0.11	84.61 ± 0.22	91.77 ± 0.15
DW + feats.	77.21 ± 0.03	86.28 ± 0.07	90.05 ± 0.08	87.70 ± 0.04	94.90 ± 0.09
DGI	75.35 ± 0.14	83.95 ± 0.47	91.61 ± 0.22	92.15 ± 0.63	94.51 ± 0.52
GMI	74.85 ± 0.08	82.21 ± 0.31	90.68 ± 0.17	OOM	OOM
MVGRL	77.52 ± 0.08	87.52 ± 0.11	91.74 ± 0.07	92.11 ± 0.12	95.33 ± 0.03
GRACE	**77.97 ± 0.63**	86.50 ± 0.33	92.46 ± 0.18	92.17 ± 0.04	OOM
GCA	77.94 ± 0.67	87.32 ± 0.50	92.39 ± 0.33	92.84 ± 0.15	OOM
BGRL	76.86 ± 0.74	89.69 ± 0.37	93.07 ± 0.38	92.59 ± 0.14	95.48 ± 0.08
AFGRL	77.62 ± 0.49	**89.88 ± 0.33**	**93.22 ± 0.28**	**93.27 ± 0.17**	**95.69 ± 0.10**

Table 2: Performance on node classification (OOM: Out of memory on 24GB RTX3090).

Experiments

		GRACE	GCA	BGRL	AFGRL
WikiCS	NMI	**0.4282**	0.3373	0.3969	0.4132
	Hom.	**0.4423**	0.3525	0.4156	0.4307
Computers	NMI	0.4793	0.5278	0.5364	**0.5520**
	Hom.	0.5222	0.5816	0.5869	**0.6040**
Photo	NMI	0.6513	0.6443	**0.6841**	0.6563
	Hom.	0.6657	0.6575	**0.7004**	0.6743
Co.CS	NMI	0.7562	0.7620	0.7732	**0.7859**
	Hom.	0.7909	0.7965	0.8041	**0.8161**
Co.Physics	NMI	OOM	OOM	0.5568	**0.7289**
	Hom.	OOM	OOM	0.6018	**0.7354**

Table 3: Performance on node clustering in terms of NMI and homogeneity.

Experiments

		GRACE	GCA	BGRL	AFGRL
WikiCS	Sim@5	0.7754	0.7786	0.7739	**0.7811**
	Sim@10	0.7645	**0.7673**	0.7617	0.7660
Computers	Sim@5	0.8738	0.8826	0.8947	**0.8966**
	Sim@10	0.8643	0.8742	0.8855	**0.8890**
Photo	Sim@5	0.9155	0.9112	**0.9245**	0.9236
	Sim@10	0.9106	0.9052	**0.9195**	0.9173
Co.CS	Sim@5	0.9104	0.9126	0.9112	**0.9180**
	Sim@10	0.9059	0.9100	0.9086	**0.9142**
Co.Physics	Sim@5	OOM	OOM	0.9504	**0.9525**
	Sim@10	OOM	OOM	0.9464	**0.9486**

Table 4: Performance on similarity search. (Sim@n: Average ratio among n nearest neighbors sharing the same label as the query node.)

Experiments

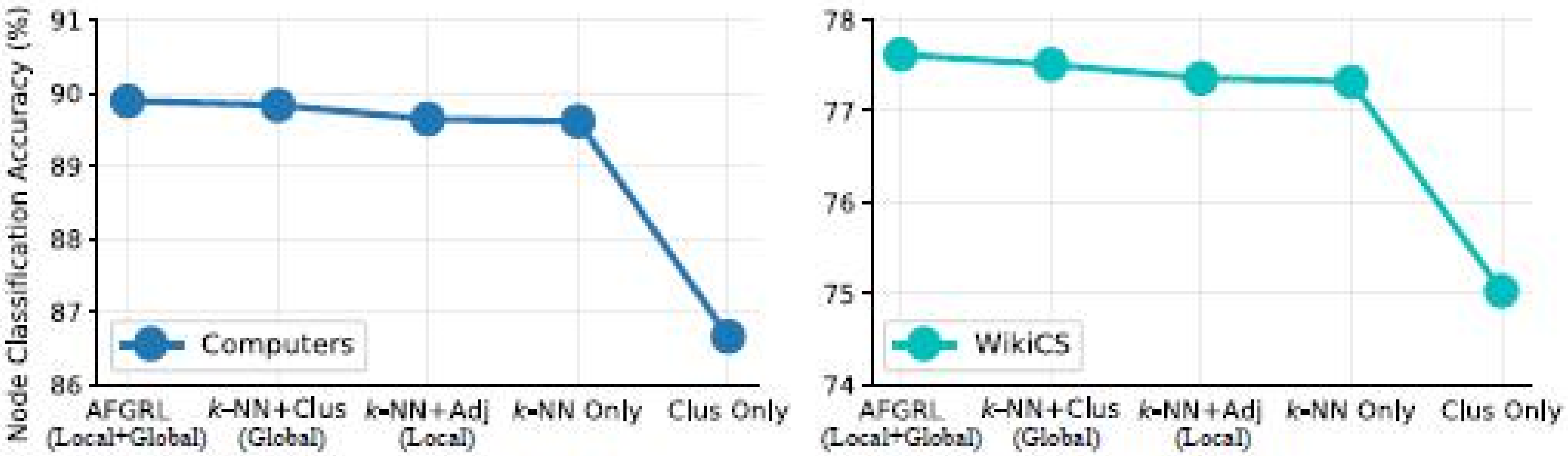


Figure 6: Ablation study on AFGRL.

Experiments

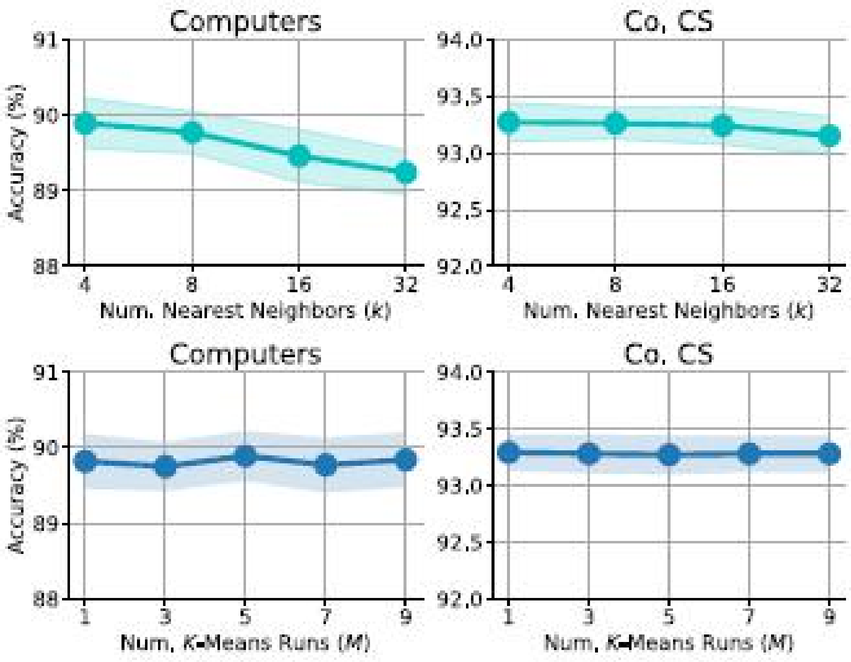


Figure 5: Sensitivity analysis.

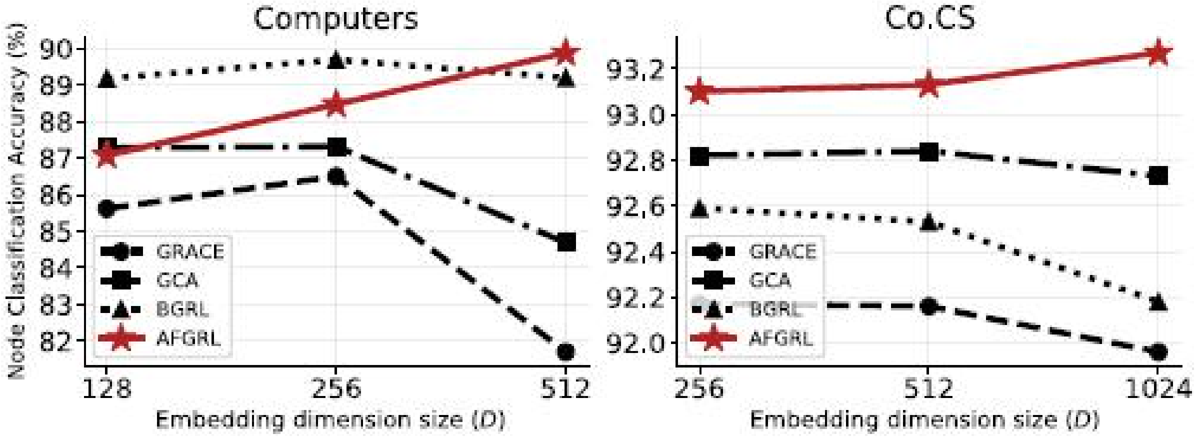
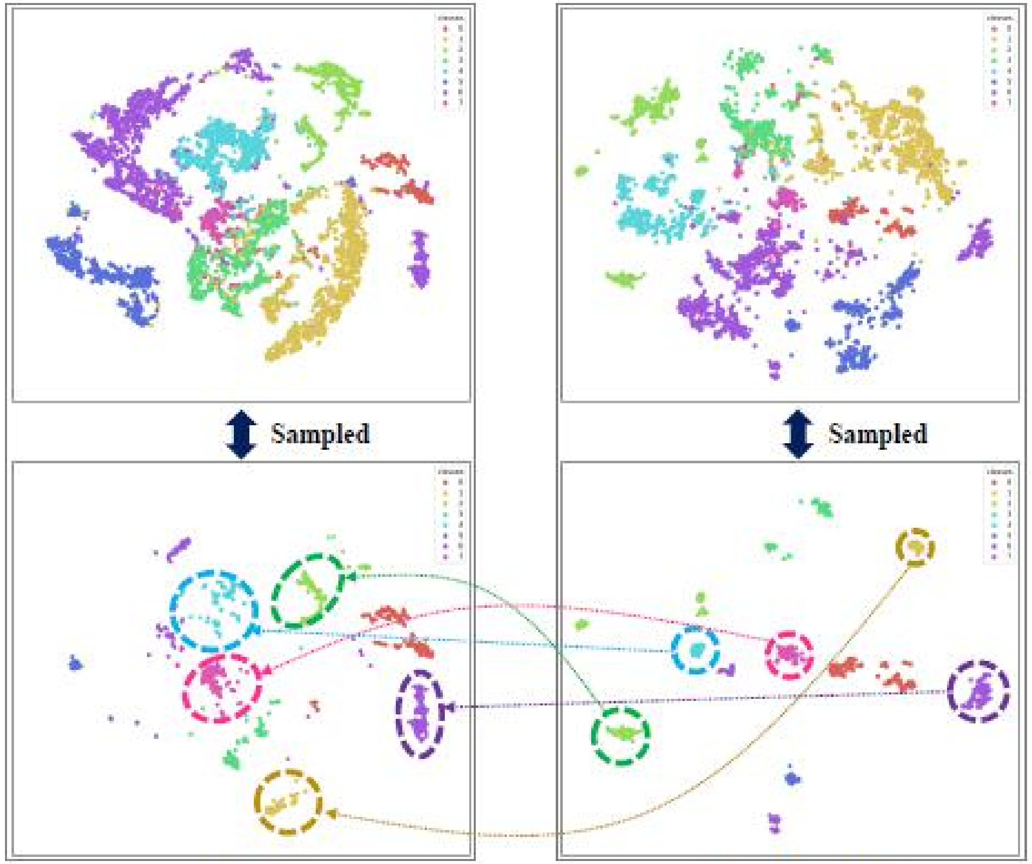


Figure 7: Effect of embedding dimension size (D).

Experiments



(a) GCA (b) AFGRL

Figure 8: t-SNE embeddings of nodes in *Photo* dataset.

Thank you!