Dual Space Graph Contrastive Learning

Haoran Yang[1], Hongxu Chen[1], Shirui Pan[2], Lin Li[3], Philip S. Yu[4], Guandong Xu[1]

[1]University of Technology Sydney
[2]Monash University
[3]Wuhan University of Technology
[4]University of Illinois at Chicago

1 Background

1.1 An illustrative example of contrastive learning

- Limitations of direct-semantic supervision:
  - The underlying data has a much richer structure than what sparse labels or rewards could provide.
  - We cannot rely on direct supervision in high dimensional problems, and the marginal cost of acquiring labels is higher in problems like Reinforcement Learning.
  - It leads to task-specific solutions, rather than knowledge that can be repurposed.

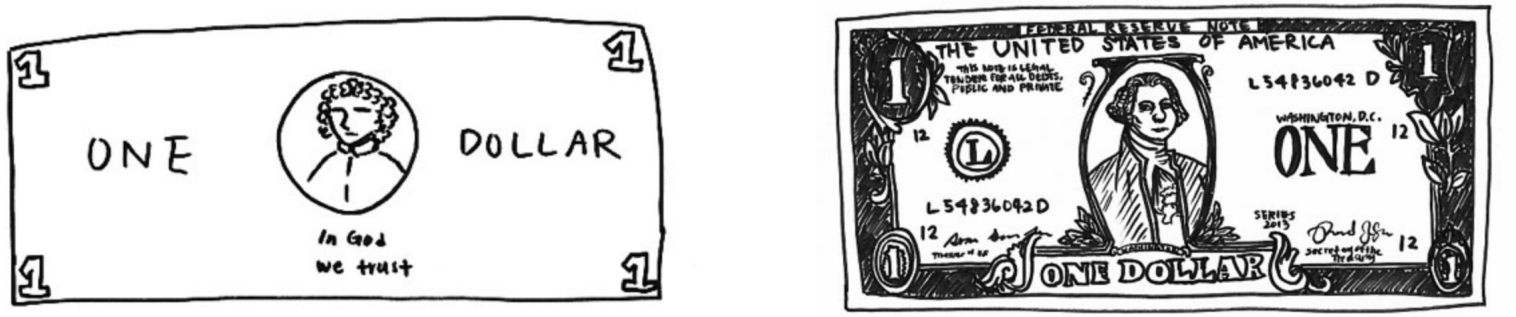


Fig. Left: Drawing of a dollar bill from memory. Right: Drawing subsequently made with a dollar bill present.[1][2]

[1] https://aeon.co/essays/your-brain-does-not-process-information-and-it-is-not-a-computer
[2] https://ankeshanand.com/blog/2020/01/26/contrative-self-supervised-learning.html

1 Background

1.2 Graph Contrastive Learning

➢ For any data point x, contrastive methods aim to learn an encoder f such that:

$$score(f(x), f(x^+) >> score(f(x), f(x^-))$$

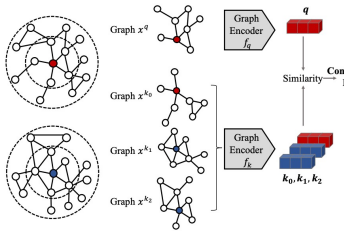
- here x^+ is the postive sample of x, and x^- is a negative sample.
- x is commonly referred to as a 'key', and the set of other positive or negative samples are regarded as a 'dictionary'.
- $score(*)$ is a metric function to measure the similarity of two data points.
- in practice, researchers usually perform dot product as the score function:

$$score(f(x), f(x^*)) = f(x)^T f(x^*)$$

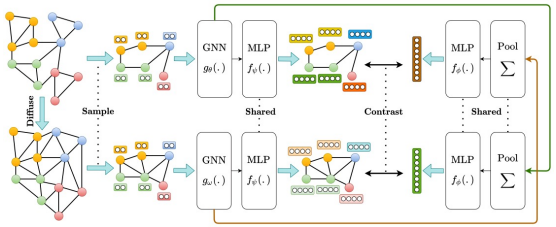
➢ InforNCE[1] is one of the widely used loss functions for contrastive learning:

$$\mathcal{L}_{InfoNCE} = -log \frac{e^{f(x)^T f(x^+)}}{\sum_{i=0}^{N} e^{f(x)^T f(x_i)}}$$

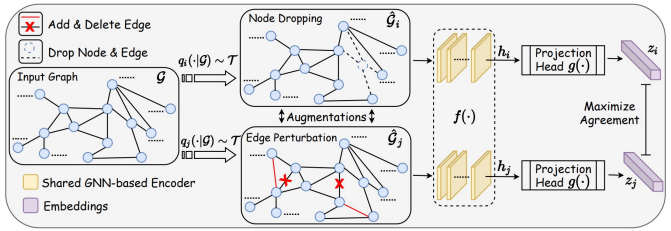
- the set $\{x_0, x_1, \ldots, x_N\}$ contains all the positive and negative samples.
- this formular is similar to cross-entropy for N-way softmax classification tasks, and commonly called InfoNCE loss in contrastive learning literature.



GCC[1]



MVGRL[2]



GraphCL[3]

Fig. Examples of graph contrastive learning protocols.

[1] Jiezhong Qiu，Qibin Chen，Yuxiao Dong，Jing Zhang，Hongxia Yang，Ming Ding，Kuansan Wang，Jie Tang，"GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training"，KDD 2020
[2] Kaveh Hassani, Amir Hosein Khasahmadi，"Contrastive Multi-View Representation Learning on Graphs"，ICML 2020
[3] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, Yang Shen，"Graph Contrastive Learning with Augmentations"，NeurIPS 2020

2 Challenges and Motivations

- Challenges:
  - Adding noises to generate contrasting pairs degrades the performance[1].
  - Sub-graph sampling is an alternative option.
  - How to generate sufficient views containing unique and informative features is still an open problem.

- Motivations:
  - MVGRL[1] adopted graph diffusion following sub-graph sampling to acquire informative contrasting views.
  - The huge success of graph representation learning in the hyperbolic spaces.
  - Utilising different embedding spaces (e.g., the Euclidean space and the hyperbolic space) to generate different views for graph contrastive learning.

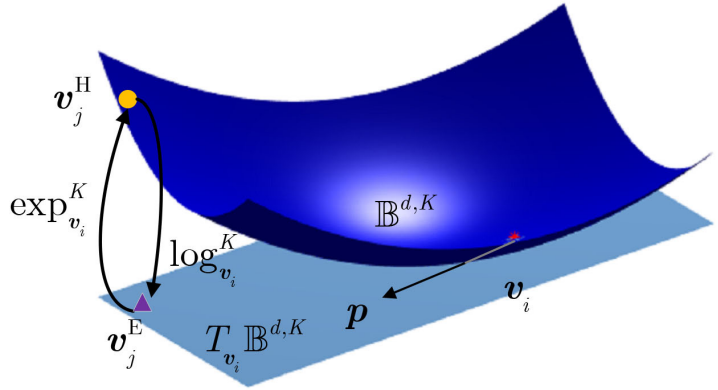


Fig. Mapping between the Euclidean space and the hyperbolic space[1].

[1] Kaveh Hassani, Amir Hosein Khasahmadi, "Contrastive Multi-View Representation Learning on Graphs" , ICML 2020
[2] Zheng Wu, Hongchan Chen, Jianpeng Zhang, "Hyperbolic Graph Attention Networks for Link Prediction in Knowledge Graph", Journal of Electronics & Information Technology.

3 Methodology

3.1 Sampling different sub-graphs for different spaces

- Differences between the Euclidean space and the hyperbolic space:
  - For the Euclidean space: relative locations, skeleton.
  - For the hyperbolic space: hierarchical structures.

- Different samplers:
  - *DiffusionSampler* for the Euclidean space.
  - *CommunitySructureExpansionSampler* for the hyperbolic space.
  - The samplers are implemented by ***little ball of fur***[1].

[1] https://github.com/benedekrozemberczki/littleballoffur

3 Methodology

3.2 Space transformation between the Euclidean space and the hyperbolic space

- Why we need space transformation:
  - Views generated in different spaces are needed to be compared.
  - Vectors in different spaces cannot be compared directly before mapping to the same space.

- Transformation mechanisms:
  - Exponential mapping (from the Euclidean space to the hyperbolic space)
  - Logarithmic mapping (from the hyperbolic space to the Euclidean space)

$$exp_{\mathbf{o}}^{c}(\mathbf{t}) = tanh(\sqrt{c}||\mathbf{t}||)\frac{\mathbf{t}}{\sqrt{c}||\mathbf{t}||},$$

$$log_{\mathbf{o}}^{c}(\mathbf{u}) = artanh(\sqrt{c}||\mathbf{u}||)\frac{\mathbf{u}}{\sqrt{c}||\mathbf{u}||}.$$

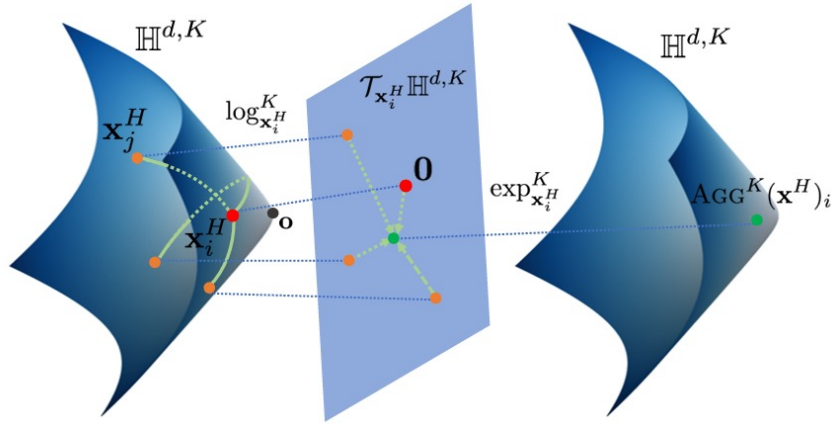
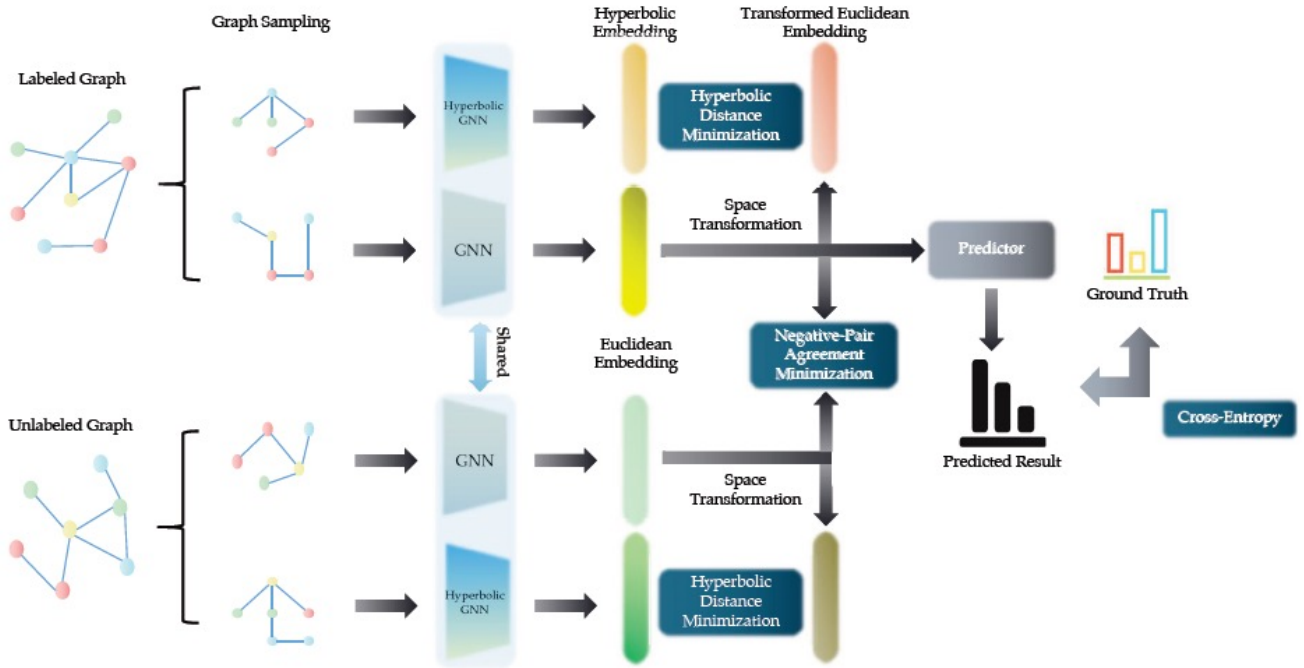


Fig. Transformation between the hyperbolic space and the Euclidean space[1].

[1] Ines Chami, Rex Ying, Christopher Re, Jure Leskovec, "Hyperbolic Graph Convolutional Neural networks", NeurIPS 2019.

3 Methodology

3.3 Overview



Algorithm 1 DSGC algorithm

Input: Labeled graph $\mathcal{G}_l$ and its label p_l; unlabeled graph $\mathcal{G}_u$; graph sampler $S_H(\cdot,\cdot)$; graph sampler $S_E(\cdot,\cdot)$; sampling rate for graph processed in hyperbolic space α_H; sampling rate for graph processed in Euclidean space α_E; trainable hyperbolic GNN model $g_H(\cdot)$; trainable Euclidean GNN model $g_E(\cdot)$; trainable predictor $P(\cdot)$; cross-entropy function $C(*)$; unlabeled loss weight λ_u;

Output: Training objective of DSGC, $\mathcal{L}$;

1: Sampling sub-graphs for labeled graph: $SG_l^H = S_H(\mathcal{G}_l, \alpha_H)$, $SG_l^E = S_E(\mathcal{G}_l, \alpha_E)$;

2: Sampling sub-graphs for unlabeled graph: $SG_u^H = S_H(\mathcal{G}_u, \alpha_H)$, $SG_u^E = S_E(\mathcal{G}_u, \alpha_E)$;

3: Generating embeddings for sampled sub-graphs via GNN models: $\mathcal{H}_l^H = exp_o^c(g_H(SG_l^H))$, $\mathcal{H}_l^E = g_E(SG_l^E)$, $\mathcal{H}_u^H = exp_o^c(g_H(SG_u^H))$, $\mathcal{H}_u^E = g_E(SG_u^E)$;

4: Mapping Euclidean embeddings of sub-graphs to hyperbolic space: $\mathcal{H}_l^{E\to H} = exp_o^c(\mathcal{H}_l^E)$, $\mathcal{H}_u^{E\to H} = exp_o^c(\mathcal{H}_u^E)$;

5: Hyperbolic distance calculation for positive pair: $d_{\mathbb{D}}^l(\mathcal{H}_l^H, \mathcal{H}_l^{E\to H})$, $d_{\mathbb{D}}^u(\mathcal{H}_u^H, \mathcal{H}_u^{E\to H})$;

6: Hyperbolic distance calculation for negative pair: $d_{\mathbb{D}}(\mathcal{H}_l^{E\to H}, \mathcal{H}_u^{E\to H})$;

7: InfoNCE loss for labeled and unlabeled data: $\mathcal{L}_{NCE}^l = -log\frac{e^{d_{\mathbb{D}}^l(\mathcal{H}_l^H, \mathcal{H}_l^{E\to H})}}{e^{d_{\mathbb{D}}^l(\mathcal{H}_l^H, \mathcal{H}_l^{E\to H})} + e^{d_{\mathbb{D}}(\mathcal{H}_l^{E\to H}, \mathcal{H}_u^H)}}$, $\mathcal{L}_{NCE}^u = -log\frac{e^{d_{\mathbb{D}}^u(\mathcal{H}_u^H, \mathcal{H}_u^{E\to H})}}{e^{d_{\mathbb{D}}^u(\mathcal{H}_u^H, \mathcal{H}_u^{E\to H})} + e^{d_{\mathbb{D}}(\mathcal{H}_l^H, \mathcal{H}_u^{E\to H})}}$;

8: Generating probobalistic distribution for labeled graph: $p = \delta(P(\mathcal{H}_l^E))$;

9: Cross-entropy loss function for labeled graph: $C(p, p_l)$;

10: **return** $\mathcal{L} = C(p, p_l) + \mathcal{L}_{NCE}^l + \lambda_u \mathcal{L}_{NCE}^u$.

4 Experiments

4.1 Datasets and Baselines

- Details of the datasets:

Name \ Statistics	Num. of Graphs	Num. of Classes	Avg. Number of Nodes	Avg. Number of Edges
MUTAG	188	2	17.93	19.79
REDDIT-BINARY	978	2	243.11	288.53
COLLAB	5,000	3	74.49	2457.78

- Selected baselines:
  - Graph neural network models: GCN, GraphSAGE, GAT, GIN
  - Graph contrastive learning methods: GCC and GraphCL

4 Experiments

4.2 Comparison experimental results

Dataset	Label Ratio \ Methods	GCN	GraphSAGE	GAT	GIN	GCC	GraphCL	DSGC
MUTAG	0.1	56.11(std 19.02)	52.78(std 19.14)	58.89(std 18.23)	50.56(std 20.98)	61.67(std 16.15)	57.78(std 13.44)	**62.22(std 15.50)**
	0.3	62.78(std 15.54)	57.22(std 18.05)	62.78(std 14.22)	54.44(std 18.01)	63.89(std 14.06)	62.78(std 12.99)	**66.11(std 12.41)**
	0.5	61.11(std 14.60)	63.33(std 15.57)	58.89(std 17.60	60.56(std 19.88)	65.56(std 13.57)	59.44(std 16.76)	**66.67(std 12.37)**
REDDIT-BINARY	0.1	52.27(std 7.54)	51.55(std 13.07)	53.71(std 12.66)	53.51(std 7.05)	51.65(std 7.36)	54.54(std 7.44)	**55.26(std 6.99)**
	0.3	56.60(std 6.08)	55.88(std 11.35)	54.43(std 7.72)	54.33(std 10.12)	53.40(std 10.68)	56.19(std 5.68)	**57.32(std 5.67)**
	0.5	55.67(std 6.96)	53.61(std 8.33)	57.63(std 7.99)	53.40(std 9.00)	52.37(std 8.81)	**58.14(std 5.73)**	57.73(std 4.35)
COLLAB	0.1	38.98(std 13.78)	38.58(std 14.20)	38.74(std 11.77)	38.48(std 10.88)	37.68(std 13.38)	46.72(std 7.78)	**50.08(std 5.79)**
	0.3	38.54(std 9.07)	42.90(std 13.44)	42.24(std 11.38)	38.56(std 4.62)	37.78(std 13.26)	48.12(std 7.51)	**50.48(std 5.14)**
	0.5	35.14(std 10.13)	36.96(std 12.19)	42.64(std 9.51)	40.24(std 6.41)	38.74(std 6.81)	46.76(std 7.20)	**52.00(std 1.39)**

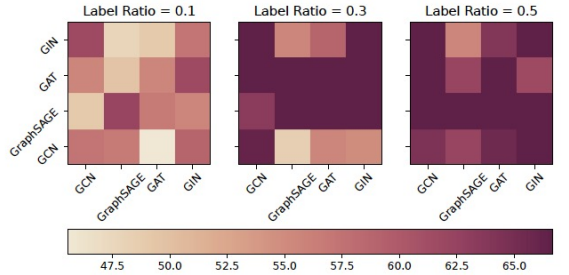
- Insights in the comparison experimental results:
  - The proposed method achieved the best results in most cases.
  - The proposed method had lower standard error, showing the stability.
  - Comparing the proposed method and GraphCL, graph augmentation (sub-graph sampling) followed by embedding in different spaces had more superiority than sole graph augmentation.

4 Experiments

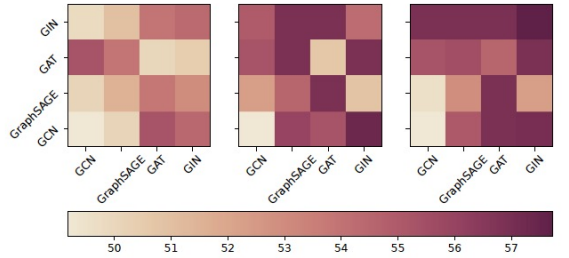
4.2 Using different graph encoders to further augment contrasting views

- How to acquire contrasting views:
  - Sampling different sub-graphs
  - Embedding in different spaces
  - Different graph encoders?

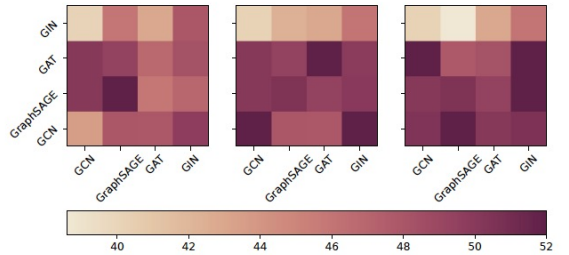
- Experiments with different graph encoders:
  - We assign different graph encoders to encode graph in different spaces.
  - Select proper combinations of graph encoders can achieve better performances.



(a) Performances of DSGC with different pairs of graph encoders for the Euclidean and Hyperbolic spaces on dataset MUTAG.



(b) Performances of DSGC with different pairs of graph encoders for the Euclidean and Hyperbolic spaces on dataset REDDIT-BINARY.

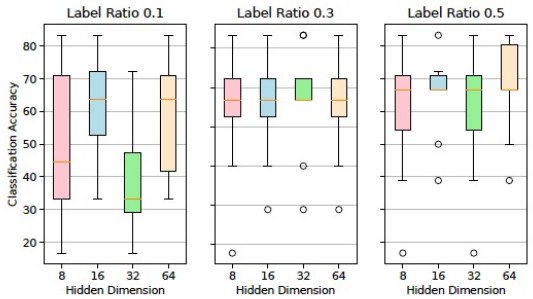


(c) Performances of DSGC with different pairs of graph encoders for the Euclidean and Hyperbolic spaces on dataset COLLAB.

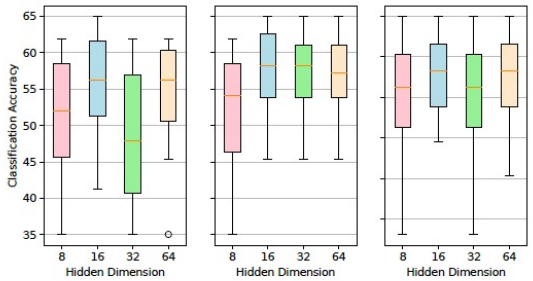
4 Experiments

4.2 Hyper-parameter study – hidden dimension

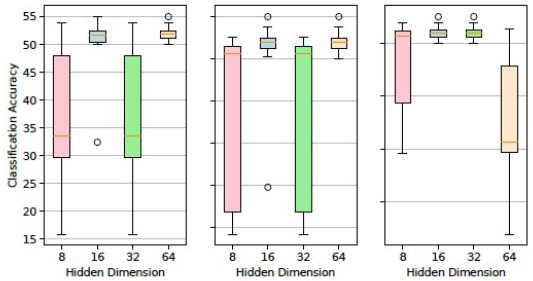
- Study scope:
  - The hyperbolic embeddings contain more semantics compare to the Euclidean embeddings when the hidden dimension is small[1].
- Experiments with different hidden dimensions:
  - We set different hidden dimensions for graph embeddings.
  - According to the experimental results, a relatively small hidden dimension (16) is more stable and has better performances.



(a) Performances of DSGC with different hidden dimension on dataset MUTAG.



(b) Performances of DSGC with different hidden dimension on dataset REDDIT-BINARY.



(c) Performances of DSGC with different hidden dimension on dataset COLLAB.

[1] Sixiao Zhang, Hongxu Chen, Xiao Ming, Lizhen Cui, Hongzhi Yin, Guandong Xu, "Where are we in embedding spaces? A Comprehensive Analysis on Network Embedding Approaches for Recommender Systems", KDD 2021.

5 Summary

- Contributions:
  - We first utilize graph representations in different spaces to construct contrasting pairs, which could leverage the advantages from both the hyperbolic space and the Euclidean space.
  - We innovatively propose to select different graph encoders for representation learning in different spaces, which is also verified by the experiments as feasible way to generate different views of the input graph.
  - Detailed experiments are provided to analyze the proposed method.

- Conclusion:
  - In this paper, we propose a novel graph contrastive learning framework, dual space graph contrastive learning (DSGC), which first utilizes the different representation abilities of different spaces to generate contrasting views. It is feasible to select different combinations of graph encoders for different spaces to further reinforce the distinctions among embeddings of the sampled sub-graphs.

Thanks for Your Listening