Dual Space Graph Contrastive Learning

Haoran Yang[*]
haoran.yang-2@student.uts.edu.au
University of Technology Sydney
Sydney, NSW, Australia

Hongxu Chen[*] [†]
hongxu.chen@uts.edu.au
University of Technology Sydney
Sydney, NSW, Australia

Shirui Pan
shirui.pan@monash.edu
Monash University
Melbourne, VIC, Australia

Lin Li
cathylilin@whut.edu.cn
Wuhan University of Technology
Wuhan, Hubei, China

Philip S. Yu
psyu@cs.uic.edu
University of Illinois at Chicago
Chicago, Illinois, U.S.A

Guandong Xu[†]
guandong.xu@uts.edu.au
University of Technology Sydney
Sydney, NSW, Australia

ABSTRACT

Unsupervised graph representation learning has emerged as a powerful tool to address real-world problems and achieves huge success in the graph learning domain. Graph contrastive learning is one of the unsupervised graph representation learning methods, which recently attracts attention from researchers and has achieved state-of-the-art performances on various tasks. The key to the success of graph contrastive learning is to construct proper contrasting pairs to acquire the underlying structural semantics of the graph. However, this key part is not fully explored currently, most of the ways generating contrasting pairs focus on augmenting or perturbating graph structures to obtain different views of the input graph. But such strategies could degrade the performances via adding noise into the graph, which may narrow down the field of the applications of graph contrastive learning. In this paper, we propose a novel graph contrastive learning method, namely **D**ual **S**pace **G**raph **C**ontrastive (DSGC) Learning, to conduct graph contrastive learning among views generated in different spaces including the hyperbolic space and the Euclidean space. Since both spaces have their own advantages to represent graph data in the embedding spaces, we hope to utilize graph contrastive learning to bridge the spaces and leverage advantages from both sides. The comparison experiment results show that DSGC achieves competitive or better performances among all the datasets. In addition, we conduct extensive experiments to analyze the impact of different graph encoders on DSGC, giving insights about how to better leverage the advantages of contrastive learning between different spaces.

CCS CONCEPTS

• **Computing methodologies → Semantic networks**; *Unsupervised learning*.

[*]Both authors contributed equally to this research.
[†]Corresponding author.

KEYWORDS

graph contrastive learning, hyperbolic space, graph embedding

ACM Reference Format:
Haoran Yang, Hongxu Chen, Shirui Pan, Lin Li, Philip S. Yu, and Guandong Xu. 2022. Dual Space Graph Contrastive Learning. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3485447.3512211

1 INTRODUCTION

Graph neural networks (GNNs) [5, 9, 12, 25, 28] leverage the expressive power of graph data in modeling complex interactions among various entities under real-world scenarios including social networks [12] and molecules [7]. GNNs are able to process variable-size graph data and learn low-dimensional embedding via an iterative process of transferring and aggregating the semantics from topological neighbors. They have achieved huge success on various tasks. GNNs require a large amount of manually labelled data to learn informative representations via the supervised learning protocol. However, it is not always realistic to obtain enough effective labels due to expensive labour work and privacy concerns in some context. Thus, unsupervised approaches such as reconstruction based methods [24] and graph contrastive learning methods [10, 21, 31] are coupled with GNNs to tackle the problem in a self-supervised manner. Recently, graph contrastive learning has emerged as a successful method for unsupervised graph representation learning and achieved state-of-the-art on various tasks including node classification, graph classification as well as applications in recommender systems [8, 30] and drug discovery [27].

The key idea of graph contrastive learning is to construct proper contrasting pairs and try to learn the underlying important structural semantics of the input graph. However, existing approaches to generate contrasting pairs are quite limited, the majority of these methods are graph perturbations including node dropping, edge perturbation and attribute masking [31, 33]. As a result, these methods introduce noisy signals to produce corrupted views, which degrade the model performance [10]. Therefore, it is always a confronting challenging problem to design effective ways to generate contrasting pairs. Let us look closer, what graph contrastive learning actually does is that it pushes negative pairs far away from each other and make positive pairs as similar as possible to each other. For negative pairs, it is easier to distinguish them against irrelevant graphs derived from other datasets [21] or other irrelevant graphs in the same training batch [31], which will not involve significant noises. In contrast, the positive samples are usually derived from the input graph via graph perturbation methods, and the noise is inevitable. To avoid introducing too much noise, many researchers instead adopt sub-graph sampling to generate positive samples [19, 21, 31]. The intuition is that each sub-graph has unique focus on certain aspects of the graph semantics and different sub-graphs or local structures can hint the full spectrum of the semantics [31] carried on the graphs. Nevertheless, how to generate sufficient views containing unique and informative features is still an open problem.

Inspired by recent progress of geometric graph mining in hyperbolic space [3, 4], which has achieved satisfactory results under various real-world scenarios, e.g, recommendation systems [15], we innovatively utilize the greater expresiveness of different embedding spaces including the hyperbolic space and the Euclidean space to generate multiple views of the input graph. There are two major advantages of the hyperbolic space. The first is that the hyperbolic space takes smaller space to accommodate a given graph with complex structures, which means hyperbolic embedding has unique expressiveness compared to its counterpart in Euclidean space. The most obvious uniqueness is that Euclidean space expands polynomially, while the hyperbolic space expands exponentially. Therefore, we can leverage that with much lower dimensional embeddings to represent a graph, which will result in purer, compact but powerful embedding spaces. According to the experimental results that showcase the impact of hidden dimensions on performances of our proposed DSGC framework, it is shown that the our model achieves stable and better performances within smaller dimensional hyperbolic embedding spaces. The second advantage is that the hyperbolic space is more capable of capturing hierarchical structures [3, 32] exhibited in graph data, which implies that hyperbolic space is also suitable to be used as a special view for graph contrastive learning paradigm.

Despite the advantages of hyperbolic space embedding, one cannot ignore the power of the traditional Euclidean space [9, 12, 25, 28]. Compared to the hyperbolic space, vector calculation in Euclidean space is more efficient as distance metrics in Euclidean space does not need inverse trigonometric functions. Moreover, metrics in Euclidean space is able to differentiate the relative distances among the data points (i.e., satisfying the triangle inequality), and the superiority is evidenced by various famous clustering methods, including k-NN [6] and HDBSCAN [2]. Thus, introducing both the hyperbolic and Euclidean spaces together to generate distinct views could illustrate the hierarchical topology and relative distances among the nodes in the original graph, which are beneficial to graph contrastive learning. To fully leverage such advantages, we propose to select different graph samplers to sample sub-graphs in different spaces. Here we argue that random sub-graph extraction cannot fully leverage the advantages of contrasting between Euclidean space and the hyperbolic space. As mentioned previously, different spaces have different advantages, it is better to fully leverage both of them. Hence, we should have different strategies to sample sub-graphs. For Euclidean space, we need to sample a sub-graph that is able to demonstrate the skeleton of the input graph (i.e., illustrating relative locations among nodes). For hyperbolic space, a sub-graph carrying hierarchical information and topology structure is expected. Further, we note that all the current works [10, 19, 21, 31] on graph contrastive learning adopt the same graph encoder to encode different views. We question that such a strategy will limit the distinctions among various views, which may undermine the performances of graph contrastive learning. To avoid this, we also propose different graph encoders for different views.

The main contributions of this paper are summarized as follows:

- We first utilize graph representations in different spaces to construct contrsting pairs, which could leverages 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 experiemnts as feasible way to generate different views of the input graph.
- We conduct extensive experiments to show the superiority of the proposed DSGC on all the datasets comparing to all the baselines. Detailed study is executed to analyze the strategy about selecting different graph encoders for the graph representation learning in different spaces. We also illustrate if the model can achieve better performances with lower hidden dimensions to verify whether DSGC leverages the advantages of the hyperbolic space.

2 RELATED WORK

2.1 Graph Contrastive Learning

Graph contrastive learning methods are currently state-of-the-art in unsupervised graph representation learning. Deep Graph Infomax (DGI) [26] first introduced Deep Infomax [11] into graph learning and achieve satisfying results via maximize the mutual information between local structure and global context. Some other works generalized such a strategy and try to measure the mutual information between different instances. For example, GCC [21] conducted contrastive learning among different sub-graphs extracted from different input graphs, which can be regarded as the mutual information estimation among different local structures. GraphCL [31] is another type, which are different from GCC. It maximized mutual inforamtion between the input graph and the augmented graph, which is contrasting between two different global views. We note that most current graph contrastive learning methods including [10, 20, 33] all conduct contrastive learning between corrupted or augmented graphs or sub-graphs. The methods to generate different view of the input graph for contrastive learning are not rich, most of them generate different views via perturbating graph structures.

2.2 Hyperbolic Representation Learning

Recently, the hyperbolic space has been explored as a latent space, whcih is suitable for various tasks. It has shown its impressive performance comparing to the Euclidean space and achieves better performances in many domains including graph representation learning [4, 16, 23] and recommednation systems [14, 32]. One key advantage of the hyperbolic space is that it expands faster than the Euclidan space. The Euclidean space expands polynomially, while the hyperbolic space expands exponentially. It suggests that the hyperbolic space would take smaller space to embed a group of data than the Euclidean space does. Another property of the hyperbolic space is that it can preserve the hierarchies of data. Many real-world graph data including social networks and e-commercial networks exhibit hierarchical tree-like structures. The hyperbolic space is suitable to serve as a tool to describe such graph data [1, 17]. However, we think both the hyperbolic space and the Euclidean space have their own advantages, and we should try to leverage both of them instead of solely utilizing one of them.

3 PRELIMINARIES

Dual space graph contrastive learning aims to capture multi-level graph semantics in different spaces. We mainly focus on contrasting between Euclidean space and non-Euclidean spaces. Specifically, we contrast graph semantics in Euclidean space and hyperbolic space, which are both widely used spaces for graph representation learning. To better understand the proposed method, in this section, we give introductions to preliminaries about hyperbolic space and the transformation between Euclidean space and hyperbolic space.

3.1 Hyperbolic Space

Various hyperbolic models are created by researchers to solve problems under different scenarios. The Poincaré ball model $\mathbb{D}$, the Klein model $\mathbb{K}$, the half-plane model $\mathbb{P}$, the hyperboloid model $\mathbb{H}$, and the hemisphere model $\mathbb{J}$ are five well-known hyperbolic models [3]. All of them have different propertities. Among them, the Poincaré ball model is the most suitable one for representation learning since it can be tuned by gradient-based optimization [18]. The definition domain of the Poincaré ball model with a constant negative curvature $-c$ is:

$$\mathbb{D} = \{(x_1, ..., x_n) : x_1^2 + \cdots + x_n^2 < \frac{1}{c}\}. \tag{1}$$

It is an n-dimensional open ball in $\mathbf{R}^n$. The straight lines in Euclidean space are corresponding to the geodesics in the Poincaré ball. Given two points in the Poincaré ball $\mathbf{u}, \mathbf{v}$, the similarity between these two points is the reciprocal of the length of geodesics linking the two points, which is defined as:

$$d_{\mathbb{D}}(\mathbf{u}, \mathbf{v}) = \frac{1}{arcosh(1 + \frac{2||\mathbf{u}-\mathbf{v}||^2}{(1-||\mathbf{u}||^2)(1-||\mathbf{v}||^2)})} \tag{2}$$

when the constant negative curvature is -1, where $||\cdot||$ is the Euclidean norm and $arcosh(\cdot)$ is the inverse hyperbolic cosine function. Note that, there is only one center in the Poincaré ball, which is the origin. Let us take the origin as the root node, the leaf nodes will spread from origin layer by layer, capturing the tree topology and hierarchical structure information of the graph [32].

3.2 Space Mapping

We cannot directly apply Euclidean gradient desecent optimization methods to the hyperbolic space, because the gradient in the hyperbolic space is different from the Euclidean gradient. It means that existing graph representation learning methods in Euclidean space is not suitable for graph learning in the hyperbolic space. To leverage existing methods for graph learning, there is a solution, which is mapping the hyperbolic embeddings to the Euclidean embeddings. Here, we introduce the mechanism of such mappings.

There are two mappings, which are called exponential mapping and logarithmic mapping, respectively. The mapping from tangent space $\mathcal{T}_{\mathbf{o}}\mathbb{D}_c$, which is an Euclidean space with $\mathbf{o}$ as the origin, to the hyperbolic space $\mathbb{D}_c$ with the constant negative curvature $-c$ is exponential mapping, and the mapping from hyperbolic space $\mathbb{D}_c$ to tangent space $\mathcal{T}_{\mathbf{o}}\mathbb{D}_c$ is logarithmic mapping [4]. Taking origin $\mathbf{o}$ as the target point ensures the simplicity and symmetry of the mapping [32]. With $\mathbf{t} \in \mathcal{T}_{\mathbf{o}}\mathbb{D}_c$ and $\mathbf{u} \in \mathbb{D}_c$, the exponential mapping $exp_{\mathbf{o}}^c : \mathcal{T}_{\mathbf{o}}\mathbb{D}_c \rightarrow \mathbb{D}_c$ and the logarithmic mapping $log_{\mathbf{o}}^c : \mathbb{D}_c \rightarrow \mathcal{T}_{\mathbf{o}}\mathbb{D}_c$ are defined as:

$$exp_{\mathbf{o}}^c(\mathbf{t}) = tanh(\sqrt{c}||\mathbf{t}||)\frac{\mathbf{t}}{\sqrt{c}||\mathbf{t}||}, \tag{3}$$

$$log_{\mathbf{o}}^c(\mathbf{u}) = artanh(\sqrt{c}||\mathbf{u}||)\frac{\mathbf{u}}{\sqrt{c}||\mathbf{u}||}. \tag{4}$$

With two mapping functions above, we can map hyperbolic embeddings to Euclidean space for further process. Matrix multiplication, bias addition, and activation are three common embedding processing methods in neural network models.

$$\mathbf{y} = \sigma(\mathbf{W} \cdot \mathbf{u} + \mathbf{b}). \tag{5}$$

The weight matrix and bias are usually defined in the Euclidean space, which means that they cannot be directly processed together with hyperbolic embeddings. To achieve that, we must first map hyperbolic embeddings to Euclidean embeddings. For hyperbolic matrix multiplication, we have:

$$\mathbf{W} \otimes \mathbf{u} = exp_{\mathbf{o}}^c(\mathbf{W} \cdot log_{\mathbf{o}}^c(\mathbf{u})), \tag{6}$$

for hyperbolic bias addition, we have:

$$\mathbf{u} \oplus \mathbf{b} = exp_{\mathbf{o}}^c(log_{\mathbf{o}}^c(\mathbf{u}) + \mathbf{b}), \tag{7}$$

and for activation, we have:

$$\mathbf{y} = exp_{\mathbf{o}}^c(\sigma(log_{\mathbf{o}}^c(\mathbf{W} \otimes \mathbf{u} \oplus \mathbf{b}))). \tag{8}$$

Leverage the transformations mentioned above, we can utilize the existing graph learning methods in the Euclidean space to process the graph learning problems in the hyperbolic space.

4 METHODOLOGY

In this section, we introduce details of the proposed **D**ual **S**pace **G**raph **C**ontrastive learning, a semi-supervised graph representation learning framework, namely **DSGC**.

4.1 Overview

The overview of the proposed framwork is shown in Figure 1. The entire framework can be devided into two parts, one for labeled graph and one for unlabeled graph. In each part, there are two different views, which are Euclidean and hyperbolic, respectively. More than the supervision signals from labels, we leverage graph contrastive learning between Euclidean space and hyperbolic space to obtain more informative self-supervised signals to enhance the ability to learn graph representations. Algorithm 1 summarizes how we combine the inforamtion from Euclidean space and hyperbolic space via updating model parameters according to the training objective of DSGC.

The figure and the algorithm provide basic understandings about our proposed framework DSGC. In the following subsections, we will give you more details.

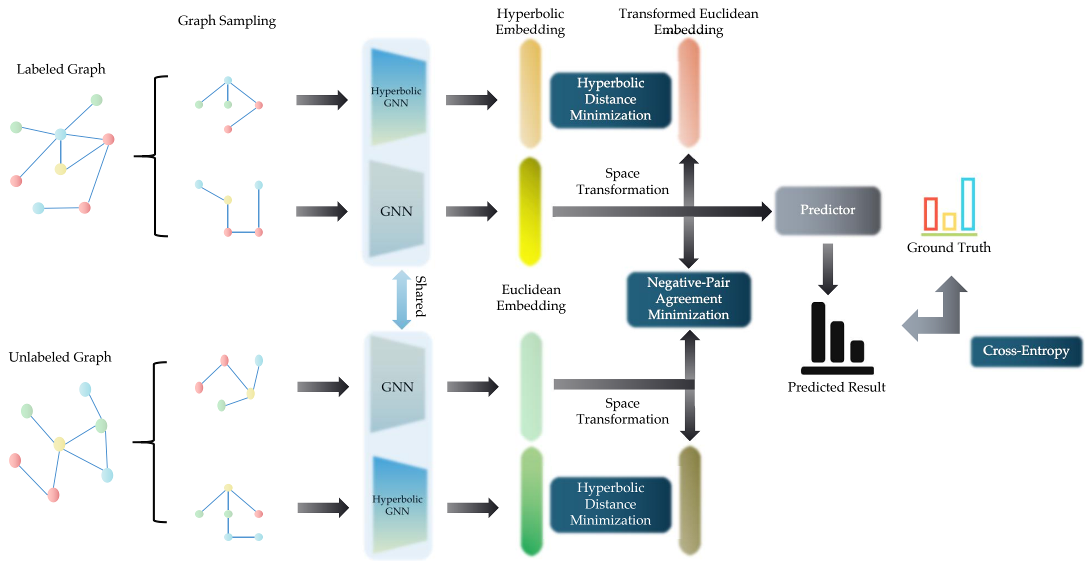


Figure 1: The overview of the proposed method. DSGC first adopts two different graph samplers to sample sub-graphs for the two different spaces on all graphs. Then, the generated sub-graphs will be fed into different graph encoders and processed in different spaces. After obtaining embeddings of all the graphs, we conduct graph contrastive learning, which is maximizing the similarity between the Euclidean embedding and the hyperbolic embedding of the same graph and minimizing the similarity between the hyperbolic embeddings of different graphs. To introduce the supervised signals into the model, the Euclidean embeddings of the labeled graphs will be fed into downstream predictor to predict the labels and the cross-entropy loss function will be utilized to update the model.

4.2 Graph Sampling

The success of graph contrastive learning mainly relies on how to generate proper contrasting pairs. As what we discussed in the introduction, we adopt sub-graph sampling, as many works do [21, 31], to generate the initial different views. Many methods can derive sub-graphs from a graph, including Depth First Search, Breath First Search, RandomWalk. Plus, sub-graph sampling is widely used as a preprocessing method for graph representation learning since it can discard noises in graphs [9]. Give a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, where $\mathcal{V}$ denotes node set, $\mathcal{E}$ denotes edge sets, x denotes features, and a graph sampler $S(\cdot)$, for the graph $\mathcal{G}$, we can have its sub-graph as follows:

$$SG = S(\mathcal{G}) \tag{9}$$

In our work, specifically, we take *DiffusionSampler* [22] to acquire a more complete and unbiased view comparing to other RandomWalk based sampling methods to serve as a skeleton of the input graph, which will be fed into Euclidean space to process and learn. For hyperbolic space, we utilize *CommunityStructureExpansionSampler* [22] to extract hierachical inforamtion and tree-topology structures of the input graph.

4.3 Graph Encoder

Graph encoders are critical to graph representation learning since we cannot convert input graphs to embeddings without them. GNN models are a satisfying branch of graph encoder leveraging the advantages of neural networks. In our proposed DSGC, we apply various GNN models to learn graph embeddings. Researchers have developed various GNN models, which can be categorized as different types, for example, convolutional GNN models in spectral domain including GCN [12], message passing based GNN models including GraphSAGE [9], attention mechanism based GNN models including GAT [25] and , and structure information aware GNN models including GIN [28]. Note that, for hyperbolic GNN models, they actually have the same structures as the Euclidean GNN models. The different point is that the output embeddings of hyperbolic GNN models will be mapped to hyperbolic space, and the hyperbolic embeddings will formulate loss function in hyperbolic space. The hyperbolic loss function trains the hyperbolic GNN models to enforce them to output proper hyperbolic embeddings.

Give a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, x)$, and a graph encoder $g(\cdot)$, for the graph $\mathcal{G}$, we can have its embeddings in Euclidean space and hyperbolic space, respectively, as follows:

$$\mathcal{H}^E = g_E(\mathcal{G}). \tag{10}$$

$$\mathcal{H}^H = exp_{\mathbf{o}}^c(g_H(\mathcal{G})). \tag{11}$$

For graph-level learning tasks, we need an extra *readout* function to summarizes the updated node embeddings to output the graph representation.

Graph embeddings are critical to the whole framework due to their strong representative ability. We now can feed them to the following components and move forward to the next step.

4.4 Space Transformation

As mentioned in the abstract, different spaces have their own advantages and features, it is unwise to pick up only one of them to

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 $\alpha_H$; sampling rate for graph processed in Euclidean space $\alpha_E$; 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 $\lambda_u$;
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: $\mathcal{H}_l^H = exp_{\mathbf{o}}^c(g_H(SG_l^H))$, $\mathcal{H}_l^E = g_E(SG_l^E)$, $\mathcal{H}_u^H = exp_{\mathbf{o}}^c(g_H(SG_u^H))$, $\mathcal{H}_u^E = g_E(SG_u^E)$;
4: Mapping Euclidean embeddings of sub-graphs to hyperbolic space: $\mathcal{H}_l^{E\to H} = exp_{\mathbf{o}}^c(\mathcal{H}_l^E)$, $\mathcal{H}_u^{E\to H} = exp_{\mathbf{o}}^c(\mathcal{H}_u^E)$;
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: $\mathcal{L}_{NCE}^l = -log \frac{\mathbf{e}^{d_{\mathbb{D}}^l(\mathcal{H}_l^H, \mathcal{H}_l^{E\to H})}}{\mathbf{e}^{d_{\mathbb{D}}^l(\mathcal{H}_l^H, \mathcal{H}_l^{E\to H})} + \mathbf{e}^{d_{\mathbb{D}}(\mathcal{H}_l^{E\to H}, \mathcal{H}_u^H)}}$, $\mathcal{L}_{NCE}^u = -log \frac{\mathbf{e}^{d_{\mathbb{D}}^u(\mathcal{H}_u^H, \mathcal{H}_u^{E\to H})}}{\mathbf{e}^{d_{\mathbb{D}}^u(\mathcal{H}_u^H, \mathcal{H}_u^{E\to H})} + \mathbf{e}^{d_{\mathbb{D}}(\mathcal{H}_l^H, \mathcal{H}_u^{E\to H})}}$;
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 $\mathcal{L} = C(p, p_l) + \mathcal{L}_{NCE}^l + \lambda_u \mathcal{L}_{NCE}^u$.
```

conduct tasks while discarding others. Instead, we could acquire sufficient semantics from contrasting pairs in different spaces via graph contrastive learning, leveraging advantages from both Euclidean space and hyperbolic space. To achieve such a goal, we conduct graph contrstive learning between Euclidean embeddings and hyperbolic embeddings to obtain informative semantics of different views of the input graph in different spaces, because there are different representation abilities among various spaces . However, embeddings in different spaces cannot be compared directly unless mapping one of them to another space in which another embedding is. Related preliminaries about space mapping and transforamtion between Eucldiean space and hyperbolic space are introduced in Section 3.2.

To compare the Euclidean embedding to the hyperbolic embedding, first, we should map graph embedding in Euclidean space to hyperbolic space via exponential mapping:

$$\mathcal{H}^{E\to H} = exp_{\mathbf{o}}^c(\mathcal{H}^E), \tag{12}$$

and, then, we will conduct contrastive learning between $\mathcal{H}^H$ and $\mathcal{H}^{E\to H}$ via minimizing the distance between these two hyperbolic embeddings. Details of contrastive objective will be given in the following section about loss functions.

4.5 Probobalistic Predictor

The task of probobalistic predictor is to predict the label of the input graph. The predicted label will be fed into cross-entropy loss function to update model parameters to introduce supervision signals. Here, we take Multi-Layer Perceptron (MLP) as the predictor to map the Eucldiean graph embedding to low-dimension space to acquire probobalistic distribution regarding all labels:

$$p = \delta(P(\mathcal{H}_l^E)), p \in \mathbf{R}^K, \tag{13}$$

where K denotes that there are K different labels in the graph dataset and $\delta(\cdot)$ denotes *sigmoid* function.

4.6 Loss Functions

In this section, we aim to formulate a loss function to train the framework, which should include self-supervised loss and supervised loss.

Objective of self-supervised learning phase is graph contrastive learning between graph hyperbolic embedding $\mathcal{H}^H$ and transformed graph Euclidean embedding $\mathcal{H}^{E\to H}$ in the hyperbolic space. Since these two embeddings are different views of the same input graph, hence, they formualte a postive pair and should be similar, which means that the distance between the two embeddings should be small enough. So, one of our self-supervised learning objective is to minimize the distance between the two embeddings $d_{\mathbb{D}}(\mathcal{H}^H, \mathcal{H}^{E\to H})$. Note that $d_{\mathbb{D}}(\cdot,\cdot)$ is defined in Section 3.1. Moreover, with a postive pair, we also need negative pairs in oreder to fulfill the graph contrastive learning process. We borrow the idea from GraphCL [31], which is coupling other graphs in the same batch with the input graph to form the negative pairs. More specifically, assume that there are N unlabeled graphs and a labeled graph in a batch. In our settings, we couple the labeled graph with N unlabeled graphs in the same batch to form negative pairs for graph contrastive learning for the labeled graph and couple an unlabeled graph with the labeled graph in the same batch to form negative pair for graph contrastive learning for each unlabeled graph. Consider that we first try to minimize the distance between the postive pair in the hyperbolic space, we need to minimize the agreement or maximize the distance between the negative pair in the hyperbolic space. The two-phase graph contrastive learning procedure mentioned above can be described as:

$$\begin{aligned}\mathcal{L}_{contra} &= \mathcal{L}_{NCE}^l + \mathcal{L}_{NCE}^u \\ &= -log \frac{\mathbf{e}^{d_{\mathbb{D}}^l(\mathcal{H}_l^H, \mathcal{H}_l^{E\to H})/\tau}}{\mathbf{e}^{d_{\mathbb{D}}^l(\mathcal{H}_l^H, \mathcal{H}_l^{E\to H})/\tau} + \sum_{i=1}^N \mathbf{e}^{d_{\mathbb{D}}(\mathcal{H}_l^{E\to H}, \mathcal{H}_{u,i}^H)/\tau}} \\ &\quad - \frac{\lambda_u}{N} \sum_{i=1}^N log \frac{\mathbf{e}^{d_{\mathbb{D}}^u(\mathcal{H}_{u,i}^H, \mathcal{H}_{u,i}^{E\to H})/\tau}}{\mathbf{e}^{d_{\mathbb{D}}^u(\mathcal{H}_{u,i}^H, \mathcal{H}_{u,i}^{E\to H})/\tau} + \mathbf{e}^{d_{\mathbb{D}}(\mathcal{H}_l^H, \mathcal{H}_{u,i}^{E\to H})/\tau}},\end{aligned} \tag{14}$$

where we introduce τ to serve as a temperature hyperparameter.

For supervised learning task, it will be conducted only on labeled graph. In this paper, we mainly focus on graph classification tasks. Therefore, we adopt cross-entropy function to measure the gap between predicted results and the ground-truth. So, the loss function of supervised learning is:

$$\mathcal{L}_{sup} = C(p, p_l), \tag{15}$$

Table 1: Statistics of 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

where $C(\cdot,\cdot)$ is the cross-entropy function, p denotes the predicted results and p_l denotes the ground-truth.

Then, we have the overoll objective for DSGC:

$$\mathcal{L} = \mathcal{L}_{sup} + \omega \cdot \mathcal{L}_{contra}, \tag{16}$$

where ω is the hyper-parameter to control the impact of graph contrastive learning on the model.

5 EXPERIMENTS

We conduct sufficient experiments and give detailed analysis in this section. We want to answer the following four questions via experiments:

- **Q1: Does the framework have the superiority comparing to the baselines?**
- **Q2: Does the proposed DSGC work with different graph encoders encoding different views of the input graphs?**
- **Q3: What is the impact of the hidden dimension of learned features on the whole framework?**

We give more details about the experiments and answers of the questions mentioned above in the following subsections.

5.1 Datasets

To fully demonstrate the performances of DSGC, we choose three public and wildly-used datasets of three categories including chemical compounds, social networks, and citation networks. which are **MUTAG** [13], **REDDIT-BINARY** [29], and **COLLAB** [29]. **MUTAG** is a collection of nitroaromatic compounds and the goal is to predict their mutagenicity on Salmonella typhimurium. **REDDIT-BINARY** is a balanced dataset where each graph corresponds to an online discussion thread where nodes correspond to users, and there is an edge between two nodes if at least one of them responded to another's comment. **COLLAB** is a scientific collaboration dataset. A graph corresponds to a researcher's ego network.

All the datasets mentioned above are available on the website[1]. The statistics of datasets are shown in Table 1. Note that, to implement sampling algorithms and meet their requirements, we discard the unconnected graphs in all datasets.

5.2 Baselines

To verify the effectiveness of the proposed framework, we compare it with several baselines in two categories. The first one is graph neural network models, including GCN [12], GraphSAGE [9], GAT [25], and GIN [28]. The second one is novel graph contrastive learning methods, including GCC [21] and GraphCL [31]. To ensure the fairness of experiments, for GNN models, we take matrix completion task as the training objective on unlabeled data following the protocol in [24], in which case GNN models would receive enough self-supervised signals from unlabeled data.

[1]https://ls11-www.cs.tu-dortmund.de/staff/morris/graphkerneldatasets

5.3 Experimental Settings

For reproducibility, we introduce the detailed settings of the proposed DSGC. First, to maintain the reliability of the experimental results, we run 10 times on each dataset, each time we have 10% data as the testing set. Then, we follow the protocol of transductive learning and take all the data as the training set and have a samll ratio of labeled data. Note that labeled data in training set does not contain any data in testing set. The metric to measure the performances of all the methods are classification accuracy. Moreover, to implement graph sampling algorithms, we discard all the graphs containing isolated nodes. All the experiments were conducted on NVIDIA TITAN Xp. We utilized PyTorch (version 1.7.0) and PyTorch Geometric (version 1.6.3) to implement our method. The hyper-parameter settings of DSGC for comparison experiment is in the appendix.

5.4 Experimental Results

5.4.1 Does the framework have the supriority comparing to the baselines? The comparison experiment results for all baselines and our proposed methods on all three datasets are shown in Table 2. Generally, the proposed DSGC method outperforms the best baselines. Note that, DSGC has much lower standard error comparing to the baselines, which shows that our proposed method is more stable when facing the datasets having different distributions. We also find that DSGC has more advantages on dataset COLLAB. It shows that the proposed contrastive learning process is able to acquire more unsupervised signals providing the model more informative semantics to achieve the best performances.

As to the baselines, GCC and GraphCL consistently outperform GNN models, which indicates contrastive learning in graph representation learning domain is feasible and could achieve better performances. Moreover, GraphCL has superiority comparing to GCC on dataset REDDIT-BINARY and COLLAB, which are larger than dataset MUTAG. Obviously, GraphCL performs better on datasets with large scales. It is because that GraphCL has a Multi-Layer Perceptron (MLP) as the readout function to aggregate the processed node features, which is more capable to process complex data comparing to average readout function adopted by GCC. Note that, our proposed DSGC also adopted average readout function to have graph embeddings, but DSGC outperforms GraphCL. This phenomenon indicates that the contrastive learning settings in DSGC may be more effective than GraphCL.

With different ratio of labeled data, the differences among the results of GNN models are not significant. It is reasonable for unsupervised learning on the datasets without features. Because, on

Table 2: Comparison experiment results of classification accuracies and standad error of all the comparing methods (the best results are in bold-face).

Dataset	Methods / Label Ratio	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)**

the one hand, the features or the embeddings of the test instances are all trained via the contrastive learning part, supervised signals have no direct impact on this training phase. On the other hand, supervised signals impacts embedding training of the test data via updating graph encoders' parameters. However, the graph encoders we adopt are simple, which have no deep or sophisticated structures. A small portion of labeled data is capable to train the encoders and too much labeled data may force the encoders to be overfitting. But the proposed DSGC method have no such phenomenon, we believe that the dual space contrastive learning protocol here is helpful to address the overfitting problem in the supervised learning phase.

5.4.2 Does the proposed DSGC work with different graph encoders encoding different views of the input graphs? The core idea of graph contrastive learning is how to construct proper contrasting pairs to conduct contrastive learning to obtain informative semantics. In this paper, we propose to utilize the different ability to learn representations of the Euclidean and Hyperbolic space to have the different but related views of the input graph. To fully leverage such different ability, we should select different graph encoders for different spaces. Adopting different graph encoders enriches the semantics among contrasting instances. It is critical to find out how to select graph encoders to fully leverage the advantages of DSGC. In this section, we have an initial exploration about the impact that different pairs of graph encoders have on the proposed DSGC framework. The experiment results are illustrated by Fig. 2, where the x-axis corresponds to the graph encoders for the hyperbolic space and the y-axis corresponds to the graph encoders for the Euclidean space. The block with deep colour represents higher accuracies.

The first insight is that the higher ratio of labeled training data the better overall performances DSGC has. But when ratio is low, it will be tricky to select different pairs of graph encoders for DSGC to achieve better performances. So, choosing different graph encoders to produce contrasting views of input graph to conduct graph contrastive learning is indeed feasible and would be useful when label ratio is lower. In other words, when we face the real-world problems about graph contrastive learning lacking of labels, we could carefully choose the pairs of different encoders to achieve better results.

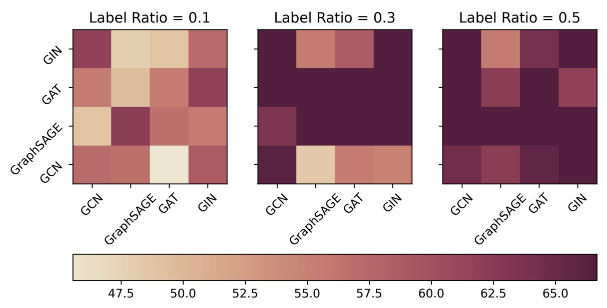
Moreover, according to the figures, we note that the best performances will not only appear when both encoders are the same but also appear when the encoders are different. It means that some different combinations of graph encoders have the potential to achieve the best results. Therefore, under real-world scenarios, it is possibile for us to find out the pairs of encoders with less complexity to improve the efficiency of the models. For example, in the second figure in Fig. 2(c), we can see that GAT-GAT combination achieves one of the best results. However, GIN-GCN combination also has good performances. Note that, GAT are more complex than both GIN and GCN because of its attention mechanism. So, in practice, we should adopt GIN-GCN combination to replace GAT-GAT combination to have higher efficiency.

5.4.3 What is the impact of the hidden dimension of learned features on the whole framework? DSGC not only introduce the advantages of the Euclidean space but also leverages the representation ability of the hyperbolic space. One of the major advantages of the hyperbolic space is that it does not need larger hidden dimensions to acquire representative embeddings [32]. In this section, we explore the performances of DSGC with different hidden dimensions to verify if it leverages such an advantage. To achieve this goal, we conduct experiments on DSGC with different hidden dimensions, we present their performances in each fold on all the datasets in the Fig. 3.

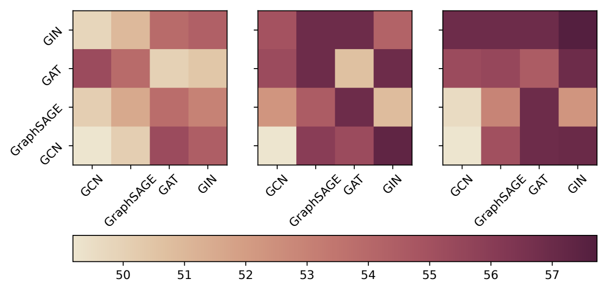
As the figure shows, we note that smaller hidden dimensions, e.g, 8 and 16, do achieve more stable performances, which is that the average of the results of 10 folds and the standard error is continuous on the same dataset with different label ratio. But, when hidden dimension is 8, the performances is unsatisfying. We think it is too small to maintain semantics of graphs even if we have introduced the hyperbolic space. When hidden dimensions are large, e.g., 32 and 64, the results have not shown superiority. Note that, the performances of DSGC with different hidden dimensions are not stable on the dataset COLLAB, and the situations are more severe when hidden dimension is 32 or 64. Therefore, for DSGC, we should adopt relatively small hidden dimensions to have higher efficience and better performances.

6 CONCLUSION

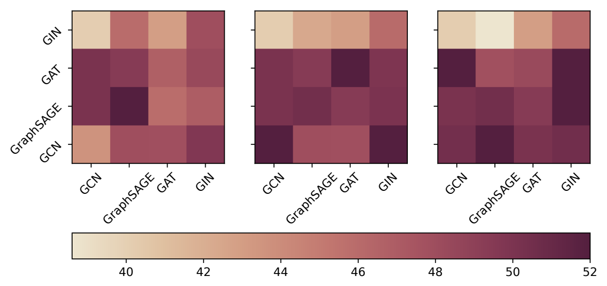
In this paper, we propose a novel graph contrastive learning framework, dual space graph contrastive learning (DSGC), which first utilizes the different representation abilities of different spaces to generate contrasting views. Moreover, to fully leverage advantages



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

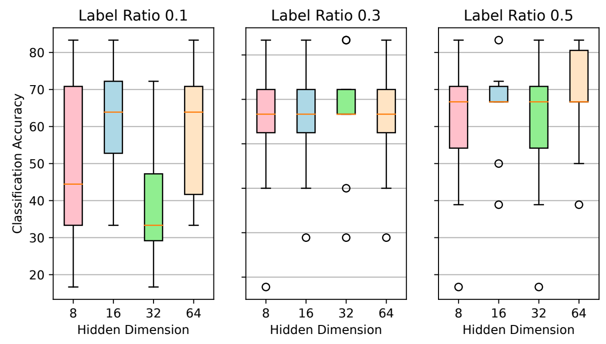


(b) Performances of DSGC with different pairs of graph encoders for the Euclidean and Hyperbolic spaces on dataset REDDIT-BINARY.

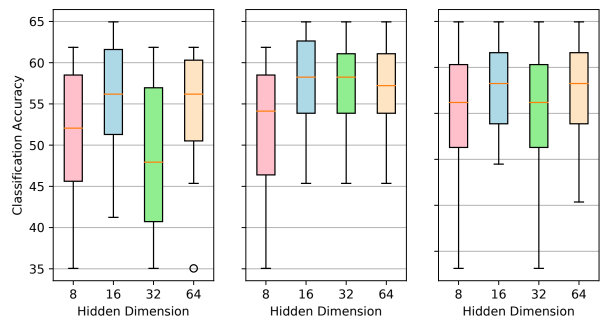


(c) Performances of DSGC with different pairs of graph encoders for the Euclidean and Hyperbolic spaces on dataset COLLAB.

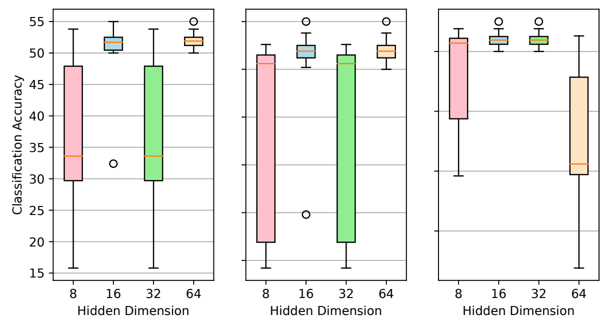
Figure 2: We explore the possibility to apply different graph encoders for graph representation learning in different spaces. The experiment results show that it is feasibile to adopt different graph encoders to acquire the contrasting views of the input graph in differen spaces to conduct graph contrastive learning.



(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.

Figure 3: DSGC's performances of each fold on all three datasets with different hidden dimensions.

of different spaces, we choose different graph sampler to sample sub-graphs and explore the different combinations of graph encoders for different spaces to further reinforce the distinctions among embeddings of the sampled sub-graphs. The feasibility of such strategy is verified by the experiments and we conduct extensive experiments to analyze the properties of DSGC. Our explorations provide a novel way to conduct graph contrastive learning and extend the potential application scenarios of graph contrastive learning.

ACKNOWLEDGMENTS

This work is supported by the Australian Research Council (ARC) under grant No. DP220103717, DP200101374, LP170100891, and LE220100078, and NSF under grants III-1763325, III-1909323, III-2106758, and SaTC-1930941.

REFERENCES

[1] Aaron B. Adcock, Blair D. Sullivan, and Michael W. Mahoney. 2013. Tree-Like Structure in Large Social and Information Networks. In *2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013*, Hui

Xiong, George Karypis, Bhavani M. Thuraisingham, Diane J. Cook, and Xindong Wu (Eds.). IEEE Computer Society, 1–10. https://doi.org/10.1109/ICDM.2013.77

[2] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. 2013. Density-Based Clustering Based on Hierarchical Density Estimates. In *Advances in Knowledge Discovery and Data Mining*, Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 160–172.

[3] James W. Cannon, William J. Floyd, Richard Kenyon, Walter, and R. Parry. 1997. Hyperbolic geometry. In *In Flavors of geometry*. University Press, 59–115.

[4] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic Graph Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 4869–4880. https://proceedings.neurips.cc/paper/2019/hash/0415740eaa4d9decbc8da001d3fd805f-Abstract.html

[5] Hongxu Chen, Hongzhi Yin, Weiqing Wang, Hao Wang, Quoc Viet Hung Nguyen, and Xue Li. 2018. PME: Projected Metric Embedding on Heterogeneous Networks for Link Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq (Eds.). ACM, 1177–1186. https://doi.org/10.1145/3219819.3219986

[6] Thomas M. Cover and Peter E. Hart. 1967. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* 13, 1 (1967), 21–27. https://doi.org/10.1109/TIT.1967.1053964

[7] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett (Eds.). 2224–2232. https://proceedings.neurips.cc/paper/2015/hash/f9be311e65d81a9ad8150a60844bb94c-Abstract.html

[8] Naicheng Guo, Xiaolei Liu, Shaoshuai Li, Qiongxu Ma, Yunan Zhao, Bing Han, Lin Zheng, Kai-Xin Gao, and Xiaobo Guo. 2021. HCGR: Hyperbolic Contrastive Graph Representation Learning for Session-based Recommendation. *CoRR* abs/2107.05366 (2021). arXiv:2107.05366 https://arxiv.org/abs/2107.05366

[9] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 1024–1034. https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html

[10] Kaveh Hassani and Amir Hosein Khas Ahmadi. 2020. Contrastive Multi-View Representation Learning on Graphs. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 4116–4126. http://proceedings.mlr.press/v119/hassani20a.html

[11] R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=Bklr3j0cKX

[12] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[13] Nils M. Kriege and Petra Mutzel. 2012. Subgraph Matching Kernels for Attributed Graphs. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress. http://icml.cc/2012/papers/542.pdf

[14] Anchen Li, Bo Yang, Hongxu Chen, and Guandong Xu. 2021. Hyperbolic Neural Collaborative Recommender. *CoRR* abs/2104.07414 (2021). arXiv:2104.07414 https://arxiv.org/abs/2104.07414

[15] Yicong Li, Hongxu Chen, Xiangguo Sun, Zhenchao Sun, Lin Li, Lizhen Cui, Philip S. Yu, and Guandong Xu. 2021. Hyperbolic Hypergraphs for Sequential Recommendation. *CoRR* abs/2108.08134 (2021). arXiv:2108.08134 https://arxiv.org/abs/2108.08134

[16] Yadan Luo, Zi Huang, Hongxu Chen, Yang Yang, and Mahsa Baktashmotlagh. 2020. Interpretable Signed Link Prediction with Signed Infomax Hyperbolic Graph. *CoRR* abs/2011.12517 (2020). arXiv:2011.12517 https://arxiv.org/abs/2011.12517

[17] Maximilian Nickel and Douwe Kiela. 2017. Poincaré Embeddings for Learning Hierarchical Representations. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 6338–6347. https://proceedings.neurips.cc/paper/2017/hash/59dfa2df42d9e3d41f5b02bfc32229dd-Abstract.html

[18] Maximilian Nickel and Douwe Kiela. 2017. Poincaré Embeddings for Learning Hierarchical Representations. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 6338–6347. https://proceedings.neurips.cc/paper/2017/hash/59dfa2df42d9e3d41f5b02bfc32229dd-Abstract.html

[19] Zhen Peng, Yixiang Dong, Minnan Luo, Xiao-Ming Wu, and Qinghua Zheng. 2020. Self-Supervised Graph Representation Learning via Global Context Prediction. *CoRR* abs/2003.01604 (2020). arXiv:2003.01604 https://arxiv.org/abs/2003.01604

[20] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph Representation Learning via Graphical Mutual Information Maximization. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen (Eds.). ACM / IW3C2, 259–270. https://doi.org/10.1145/3366423.3380112

[21] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1150–1160. https://doi.org/10.1145/3394486.3403168

[22] Benedek Rozemberczki and Rik Sarkar. 2020. Fast Sequence-Based Embedding with Diffusion Graphs. *CoRR* abs/2001.07463 (2020). arXiv:2001.07463 https://arxiv.org/abs/2001.07463

[23] Wenzhuo Song, Hongxu Chen, Xueyan Liu, Hongzhe Jiang, and Shengsheng Wang. 2021. Hyperbolic node embedding for signed networks. *Neurocomputing* 421 (2021), 329–339. https://doi.org/10.1016/j.neucom.2020.10.008

[24] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *CoRR* abs/1706.02263 (2017). arXiv:1706.02263 http://arxiv.org/abs/1706.02263

[25] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=rJXMpikCZ

[26] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. 2019. Deep Graph Infomax. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=rklz9iAcKQ

[27] Yingheng Wang, Yaosen Min, Xin Chen, and Ji Wu. 2021. Multi-view Graph Contrastive Representation Learning for Drug-Drug Interaction Prediction. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 2921–2933. https://doi.org/10.1145/3442381.3449786

[28] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=ryGs6iA5Km

[29] Pinar Yanardag and S. V. N. Vishwanathan. 2015. Deep Graph Kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams (Eds.). ACM, 1365–1374. https://doi.org/10.1145/2783258.2783417

[30] Haoran Yang, Hongxu Chen, Lin Li, Philip S. Yu, and Guandong Xu. 2021. Hyper Meta-Path Contrastive Learning for Multi-Behavior Recommendation. *CoRR* abs/2109.02859 (2021). arXiv:2109.02859 https://arxiv.org/abs/2109.02859

[31] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph Contrastive Learning with Augmentations. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/3fe230348e9a12c13120749e3f9fa4cd-Abstract.html

[32] Sixiao Zhang, Hongxu Chen, Xiao Ming, Lizhen Cui, Hongzhi Yin, and Guandong Xu. 2021. Where are we in embedding spaces? A Comprehensive Analysis on Network Embedding Approaches for Recommender Systems. *CoRR* abs/2105.08908 (2021). arXiv:2105.08908 https://arxiv.org/abs/2105.08908

[33] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph Contrastive Learning with Adaptive Augmentation. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 2069–2080. https://doi.org/10.1145/3442381.3449802

A HYPER-PARAMETER SETTINGS

For the reproducibility of our work, we list all the hyper-parameter settings of comparison study in Table 3.

Table 3: Hyper-parameter settings of the proposed DSGC in the comparison study on all the datasets with different ratio of labeled data in training set.

Dataset & Label Ratio / Hyper-parameters	MUATG			REDDIT-BINARY			COLLAB		
	0.1	0.3	0.5	0.1	0.3	0.5	0.1	0.3	0.5
Euclidean Encoder	GraphSAGE	GCN	GCN	GCN	GCN	GIN	GraphSAGE	GraphSAGE	GraphSAGE
Hyperbolic Encoder	GraphSAGE	GCN	GCIN	GAT	GIN	GIN	GCN	GraphSAGE	GIN
Number of Encoder Layers	3	3	3	1	1	1	3	3	1
Temperature	1	1	1	100	100	100	100	100	100
Learning Rate	1e-4	1.7e-4	5e-5	2e-5	1e-4	1e-5	2e-5	2e-5	2e-5
Weight Decay	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5
Number of Training Epoch	200	200	200	200	200	200	200	200	200
Weight of Contrastive Learning (w)	0.01	0.01	0.01	1e-5	0.01	0.01	0.01	1e-4	0.01
Batch Size	8	8	8	16	16	16	32	64	64
Hidden Dimension	16	16	16	16	16	16	16	16	16