# Graph Representation Learning in Hyperbolic Space via Dual-Masked

**Rui Gong[1,2]*, Zuyun Jiang[3], Daren Zha[1,2]**
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]China Telecom Digital Intelligence Technology Co, Beijing, China
{gongrui, zhadaren}@iie.ac.cn, jiangzy6@chinatelecom.cn

## Abstract

Graph representation learning (GRL) in hyperbolic space has gradually emerged as a promising approach. Meanwhile, masking and reconstruction-based (MR-based) methods lead to state-of-the-art self-supervised graph representation. However, existing MR-based methods do not fully consider deep node and structural information. Inspired by the recent active and emerging field of self-supervised learning, we propose a novel node and edge dual-masked self-supervised graph representation learning framework in hyperbolic space, named HDM-GAE. We have designed a graph dual-masked module and a hyperbolic structural self-attention encoder module to mask nodes or edges and perform node aggregation within hyperbolic space, respectively. Comprehensive experiments and ablation studies on real-world multi-category datasets, demonstrate the superiority of our method in downstream tasks such as node classification and link prediction.

## 1 Introduction

Previous research on graph representation learning (GRL) has primarily focused on methods within Euclidean space. These methods include traditional graph representation techniques based on matrix factorization and random walks. Subsequently, the introduction of graph convolutional neural networks (GCNs) significantly advanced graph representation learning. These GCNs methods (Kipf and Welling, 2017; Hamilton et al., 2017; Hu et al., 2021) are based on message-passing mechanisms and demonstrate powerful representational capabilities. However, despite the impressive performance of these Euclidean space-based graph representation methods in many tasks, they may face challenges (Papadopoulos et al., 2012) when dealing with complex graph structures and high-dimensional data.

Embedding graphs into hyperbolic space (Sala et al., 2018; Nickel and Kiela, 2017, 2018) offers advantages in terms of metrics or distances, particularly for hierarchical data. Despite its representational strength, designing and training neural networks in hyperbolic space remains a challenge. Traditional hyperbolic convolution and attention mechanisms, which follow the manifold-tangent space-manifold approach, may distort the global structure of the hyperbolic manifold (Huang et al., 2017). Integrating masking mechanisms into research within hyperbolic space holds significant feasibility. Masking mechanisms can be utilized for self-supervised learning by obscuring parts of the graph structure or node features to generate self-supervised tasks, thereby forcing the model to learn more meaningful representations. Introducing this mechanism into hyperbolic space can fully leverage the geometric advantages of hyperbolic space, further enhancing the effectiveness of graph representation learning.

Therefore, combining masking mechanisms with hyperbolic space in research is not only theoretically novel but also holds broad application prospects. In summary, the main contributions of this paper are as follows:

- We developed a hyperbolic dual-masking self-supervised learning architecture for graph representation learning, which fully utilizes the geometric advantages of hyperbolic space and the masking mechanism in self-supervised learning. To the best of our knowledge, this is the first study in the field of graph representation learning based on hyperbolic representation and dual-masking mechanisms.
- We designed several new modules: A graph dual-masking module that simultaneously masks and reconstructs both nodes and edges. Additionally, we proposed a hyperbolic structural self-attention encoding module. The at-

*Corresponding author



*Proceedings of the 31st International Conference on Computational Linguistics*, pages 637–646
January 19–24, 2025. 

tention coefficients are derived from the hyperbolic distances between node embeddings. Through this module, node representations can be directly aggregated in hyperbolic space, ensuring the preservation of manifold characteristics and minimizing distortion.

- Our experimental results validate the effectiveness of the proposed method in graph representation learning. Specifically, we conducted experiments on six real-world datasets, demonstrating the superiority of our approach in downstream node classification and link prediction tasks. Further ablation studies provided in-depth explanations of how each proposed component contributes to the model's success.

## 2 Preliminaries

In this section, we first discuss the formal definitions of problems related to graph representation learning. Then, we introduce some key fundamental concepts of hyperbolic geometry.

### 2.1 Problem Definition

Graph representation learning (GRL) is the process of automatically discovering representations for the nodes, edges, or entire graphs in a low-dimensional space while preserving the graph's inherent structural and semantic properties. Formally, relevant descriptions can be defined as follows.

Consider a graph $G = (V, E)$, where $V$ represents the set of nodes and $E \subseteq V \times V$ represents the set of edges. Additionally, let $A \in \{0, 1\}^{|V|\times|V|}$ be the adjacency matrix of $G$, where $A_{ij} = 1$ if there is an edge between nodes $v_i$ and $v_j$, and $A_{ij} = 0$ otherwise. Furthermore, let $X \in \mathbb{R}^{|V|\times d_x}$ be the node attribute matrix, where each row $X_i$ corresponds to a $d_x$-dimensional feature vector of node $v_i$.

Node classification (NC) is a task where each node $v_i \in V$ is assigned a label from a predefined set of classes $\mathcal{Y}$. Formally, given a graph $G = (V, E)$, and a subset of nodes $V_L \subseteq V$ with known labels $\{y_i \mid v_i \in V_L\}$, the objective is to learn a function $g : V \to \mathcal{Y}$ that can predict the labels for the unlabeled nodes $V_U = V \setminus V_L$. Link prediction (LP) involves predicting the existence of an edge between two nodes in a graph. Given a graph $G = (V, E)$, its adjacency matrix $A$, and node attribute matrix $X$, the task is to determine the likelihood of an edge $(v_i, v_j) \in V \times V$ existing in $G$.

### 2.2 Hyperbolic Spaces

A manifold is a generalized description of high-dimensional surfaces. An $n$-dimensional Riemannian manifold $(\mathcal{M}, g)$ is a differentiable manifold $\mathcal{M}$ equipped with a metric $g$. After defining a differential structure on a topological manifold, it can be locally approximated by a linear space near each point, allowing the definition of the tangent space of $\mathcal{T}_{\boldsymbol{p}}\mathcal{M}$ the manifold $\mathcal{M}$ at point $p$.

**Hyperbolic space** $\mathbb{H}^{\mathbf{n},\mathbf{K}}$ is defined as a smooth Riemannian manifold with constant negative curvature $-\mathbf{K}(\mathbf{K} > 0)$. **Lorentz Model** The $n$-dimensional Lorentz Model $\mathbb{L}^{\mathbf{n},\mathbf{K}}$ (also known as the hyperboloid model) is an $n$-dimensional Riemannian manifold with negative curvature $-\mathbf{K}(\mathbf{K} > 0)$ . It is defined as:

$$\mathbb{L}^{\mathbf{n},\mathbf{K}} = \{\mathbf{x} = (x_0, \ldots, x_n) \in \mathbb{R}^{n+1} \mid \langle \mathbf{x}, \mathbf{x}\rangle_{\mathbb{L}} = -1, \; x_0 > 0\} \tag{1}$$

, where $\langle\cdot,\cdot\rangle_{\mathbb{L}}$ is the Lorentzian inner product. For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n+1}$, the Lorentzian inner product is defined as: $\langle \mathbf{x}, \mathbf{y}\rangle_{\mathbb{L}} = -x_0 y_0 + \sum_{i=1}^{n} x_i y_i$

**Exponential and logarithmic mappings** Mappings between hyperbolic space and its tangent space can be defined through the exponential and logarithmic maps. Given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{L}^{\mathbf{n},\mathbf{K}}$ and a vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathbb{L}^{\mathbf{n},\mathbf{K}}$ in the tangent space, where $\boldsymbol{v} \neq 0$ and $\boldsymbol{y} \neq \boldsymbol{x}$ , the exponential map $\exp_{\boldsymbol{x}} : \mathcal{T}_{\boldsymbol{x}}\mathbb{L} \to \mathbb{L}$ and the logarithmic map $\log_{\boldsymbol{x}} : \mathbb{L} \to \mathcal{T}_{\boldsymbol{x}}\mathbb{L}$ can be defined as follows:

$$\exp_{\boldsymbol{x}}^{K}(\boldsymbol{v}) = \cosh\left(\sqrt{K}\|\boldsymbol{v}\|_{\mathbb{L}}\right)\boldsymbol{x} + \boldsymbol{v}\frac{\sinh\left(\sqrt{K}\|\boldsymbol{v}\|_{\mathbb{L}}\right)}{\sqrt{K}\|\boldsymbol{v}\|_{\mathbb{L}}} \tag{2}$$

$$\log_{\boldsymbol{x}}^{K}(\boldsymbol{y}) = d_{\mathbb{L}}^{K}(\boldsymbol{x}, \boldsymbol{y})\frac{\boldsymbol{y} + K\langle \boldsymbol{x}, \boldsymbol{y}\rangle_{\mathbb{L}}\boldsymbol{x}}{\|\boldsymbol{y} + K\langle \boldsymbol{x}, \boldsymbol{y}\rangle_{\mathbb{L}}\boldsymbol{x}\|_{\mathbb{L}}} \tag{3}$$

, where $\|\boldsymbol{v}\|_{\mathbb{L}} = \sqrt{\langle \boldsymbol{v}, \boldsymbol{v}\rangle_{\mathbb{L}}}$ is the norm of $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathbb{L}^{\mathbf{n},\mathbf{K}}$ .

**Isometric isomorphism** In addition to the aforementioned Lorentz model $\mathbb{L}$, there are various hyperbolic models (Peng et al., 2021), such as the Klein model $\mathcal{K}$, the Poincaré ball model $\mathcal{B}$, the hemisphere model $\mathcal{J}$, and the Poincaré half-space model $\mathcal{P}$. It is important to note that different hyperbolic space models have their unique definitions, metrics, exponential and logarithmic mapping methods, but they are mathematically isometric.

Given a point $\mathbf{x} = (x_0, x_1, \ldots, x_n), \mathbf{x} \in \mathbb{L}^{\mathbf{n+1},\mathbf{K}}$ , and its corresponding point $\mathbf{y} =$

$(y_0, y_1, \ldots, y_{n-1}), \mathbf{y} \in \mathcal{K}^{\mathbf{n},\mathbf{K}}$, the bijection between them can be defined as:

$$\begin{aligned} \pi^K_{\mathcal{L}\to\mathcal{K}}(\mathbf{x}) &= \sqrt{K}\frac{(x_1, \cdots, x_n)}{x_0}, \\ \pi^K_{\mathcal{K}\to\mathcal{L}}(\mathbf{y}) &= \frac{K}{\sqrt{K - \|\mathbf{y}\|^2}}(\sqrt{K}, \mathbf{y}) \end{aligned} \tag{4}$$

Similarly, given a point $\mathbf{x} = (x_0, x_1, \ldots, x_n)$, $\mathbf{x} \in \mathbb{L}^{\mathbf{n+1},\mathbf{K}}$, and its corresponding point $\mathbf{b} = (b_0, b_1, \ldots, b_{n-1}), \mathbf{b} \in \mathcal{B}^{\mathbf{n},\mathbf{K}}$, the bijection between them can be defined as:

$$\begin{aligned} \pi^K_{\mathcal{L}\to\mathcal{B}}(\mathbf{x}) &= \frac{[x_1, \cdots, x_n]}{x_0 + \sqrt{K}}, \\ \pi^K_{\mathcal{B}\to\mathcal{L}}(\mathbf{b}) &= \frac{((1 + K\|\mathbf{b}\|^2)\sqrt{K}, 2K\mathbf{b})}{1 - K\|\mathbf{b}\|^2} \end{aligned} \tag{5}$$

These bijections illustrate the relationship between the Lorentz model $\mathbb{L}$ and the Klein model $\mathcal{K}$, as well as the Lorentz model a$\mathbb{L}$nd the Poincaré ball model $\mathcal{B}$, respectively.

In the method proposed later in this paper, for numerical stability and to reduce distortion (Nickel and Kiela, 2018), we primarily use the Lorentz (hyperboloid) model $\mathbb{L}$ for embedding the network, and the Klein model $\mathcal{K}$ for aggregating node embeddings.

## 3 Methods

We propose a hyperbolic dual-masking self-supervised learning architecture for graph representation learning, leveraging the geometric advantages of hyperbolic space and the masked autoencoding mechanism in self-supervised learning. Figure 1 illustrates the block diagram of the entire architecture.

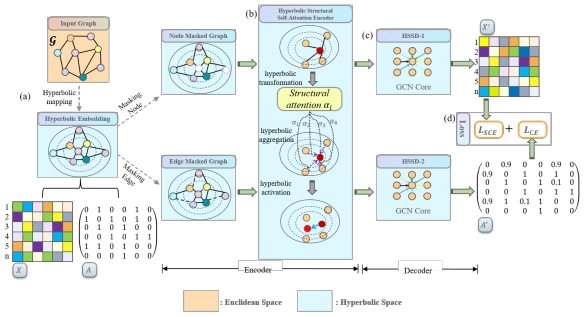


Figure 1: Overview of the proposed HDM-GAE framework, which utilizes hyperbolic space and the dual-masked mechanism to perform masked graph modeling.

### 3.1 Feature Mapping from Euclidean to Hyperbolic Space

In existing graph datasets, input features are typically located in Euclidean space and are often one-hot encoded or $n$-dimensional vectors. Before performing subsequent masking and encoding operations for embedding, it is necessary to map these features to a hyperbolic space using the exponential map.

Specifically, let $x_i^E \in \mathbb{R}^{d-1}$ be the Euclidean space feature of node $v_i$, where $d-1$ represents the input dimension. Define the origin in hyperbolic space $\mathbb{L}$ as $o := \{\sqrt{K}, 0, \ldots, 0\} \in \mathbb{L}^{\mathbf{d},\mathbf{K}}$, which serves as the reference point for tangent space operations. Since $\langle (0, x_i^E), o\rangle = 0$, the input feature $(0, x_i^E)$ can be considered as a point in the tangent space $\mathcal{T}_o\mathbb{L}^{\mathbf{d},\mathbf{K}}$ at the origin $o$. Then, we use the exponential map to generate hyperbolic node representations in the Lorentz model:

$$\begin{aligned} x_i^{\mathbb{L}} &= \exp_o^K((0, x_i^E)) \\ &= \left(\frac{\cosh\left(\sqrt{K}\|\boldsymbol{x_i^E}\|_{\mathbb{L}}\right)}{\sqrt{K}}, x_i^E\frac{\sinh\left(\sqrt{K}\|\boldsymbol{x_i^E}\|_{\mathbb{L}}\right)}{\sqrt{K}\|\boldsymbol{x_i^E}\|_{\mathbb{L}}}\right) \end{aligned} \tag{6}$$

### 3.2 Graph Dual-Mask

In the fields of computer vision and natural language processing, the masking mechanism is very common (Kenton and Toutanova, 2019; Xie et al., 2022). In recent works on graph representation learning, such as GraphMAE (Hou et al., 2022), advanced levels in self-supervised graph representation learning have been achieved by masking and reconstructing graph nodes. However, simply applying masking operations to nodes like pixels in images is flawed because nodes not only have their attribute features but, more importantly, the relationships between nodes, i.e., edges, provide critical information for subsequent tasks in graph representation learning. Therefore, when masking and reconstructing graphs, it is crucial not to ignore both edges and nodes. Our method proposes a dual-masking mechanism for graphs, applying both node masking and edge masking to the graph $\mathcal{G}$.

The input graph with $|V|$ nodes consists of an adjacency matrix $A$ and a hyperbolic node feature matrix $X^{\mathbb{L}}$, denoted as $\mathcal{G} = (A, X^{\mathbb{L}})$. We perform two masking operations on the graph $\mathcal{G}$.

First, we select a node feature masking ratio $p1$. With probability $p1$, we mask the node fea-

ture matrix $X^{\mathbb{L}}$, setting the masked parts to 0, resulting in the masked graph $\mathcal{G}^1_{mask} = (A, X^{\mathbb{L}\prime})$. Simultaneously, we select an edge masking ratio $p2$. With probability $p2$, we choose the edges to be masked, setting the corresponding entries in the adjacency matrix $A$ to 0, resulting in the masked graph $\mathcal{G}^2_{mask} = (A', X^{\mathbb{L}})$. The hyperparameters $p1$ and $p2$ control the proportion of node features and edges being masked, respectively, and have a crucial impact on the training process and performance of the model.

### 3.3 Hyperbolic Structural Self-Attention Encoder (HSSE)

The hyperbolic representation of the graph is then encoded using a structure-based self-attention mechanism. The architecture is shown in Figure 1(b). During the training process, the two masked graphs $\mathcal{G}^1_{mask}$ and $\mathcal{G}^2_{mask}$ are used as input to the entire encoding layer, HSSE. After being processed by the encoding layer, the corresponding graph feature representations $H^1$ and $H^2$ are obtained. This encoding layer can be formally defined as follows: $H^* = HSSE(\mathcal{G}^*_{mask})$.

Three fundamental modules have been designed for hyperbolic space. These modules are versatile and can be applied to graph representation learning, similar to how linear layers, convolutional layers, and activation layers are stacked in neural networks.

**Hyperbolic linear transformation** To perform operations in hyperbolic space, we first project hyperbolic vectors to the tangent space, where vector multiplication can be performed. This is done because the tangent space retains local Euclidean properties, allowing us to apply Euclidean operations. After performing the necessary operations in the tangent space, we map the vectors back to the hyperbolic space. The specific operation is defined as:

$$\mathbf{W} \otimes^K \mathbf{x}_i^{\mathbb{L}} := \exp_o^K(\mathbf{W} \log_o^K(\mathbf{x}_i^{\mathbb{L}})) \tag{7}$$

, where $\mathbf{W}$ is a weight matrix.

For bias addition, we use parallel transport to move a vector $\mathbf{b} \in \mathbb{R}^d$ from the tangent space at the origin $\mathcal{T}_o\mathbb{H}$ to the tangent space at a point $\mathcal{T}_x\mathbb{H}$. We then map this transported vector back to the hyperbolic space:

$$\mathbf{x}_i^{\mathbb{L}} \oplus^K \mathbf{b} := \exp_x^K(P_{o\to x}(\mathbf{b})) \tag{8}$$

, where $P_{o\to x}$ denotes the parallel transport from $o$ to $x$.

**Hyperbolic structural attention aggregation** Computing weighted means is essential in neural network architectures, evident in the pooling layers of CNNs and the message-passing mechanisms of GCNs. However, unlike Euclidean space, hyperbolic vectors cannot be simply averaged since this may yield results outside the manifold. As discussed previously, our method utilizes hyperbolic distance to correlate self-attention coefficients, avoiding the use of hyperbolic MLPs. This is because the hyperbolic distance effectively preserves the intrinsic properties of the graph data.

Given a node $\mathbf{m}_i^{\mathbb{L}}$ and the representation of its neighbor $\mathbf{m}_j^{\mathbb{L}}$, the attention weight $\alpha_{ij}$ is calculated as follows: $\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{v\in\mathcal{N}(i)}\exp(e_{iv})}$, where $e_{ij} = -d^{\mathbf{K}}_{\mathbb{L}}(\mathbf{m}_i^{\mathbb{L}}, \mathbf{m}_j^{\mathbb{L}})$ .

The representation $\tilde{\mathbf{m}}_i^{\mathbb{L}}$ for node $v_i$ is obtained by aggregating the hyperbolic embeddings of its neighbors, weighted by the computed attention coefficients. We use the Einstein gyromidpoint method for this aggregation, ensuring that the process is both translation and rotation invariant. The computation steps are:

$$\begin{aligned} \mathbf{m}_i^{\mathcal{K}} &= \pi^K_{\mathcal{L}\to\mathcal{K}}(\mathbf{m}_i^{\mathbb{L}}), \\ \tilde{\mathbf{m}}_i^{\mathbb{L}} &= \pi^K_{\mathcal{K}\to\mathcal{L}}\left(\frac{\sum_{j\in\widehat{\mathcal{N}}(i)}\alpha_{ij}\gamma_j\mathbf{m}_j^{\mathcal{K}}}{\sum_{j\in\widehat{\mathcal{N}}(i)}\alpha_{ij}\gamma_j}\right) \end{aligned} \tag{9}$$

, where $\gamma_j = \frac{K}{\sqrt{K-\|\mathbf{m}_j^{\mathcal{K}}\|^2_L}}$ are Lorentz factors. $\widehat{\mathcal{N}}(i)$ is the set of nodes consisting of the $i$-th node and its neighbor nodes. Therefore, the hyperbolic structure attention aggregation mechanism applies self-attention aggregation of neighboring nodes to the hyperbolic embeddings of nodes, which can be seen as selectively transmitting messages between nodes.

**Hyperbolic activation mechanism** Nonlinear activation functions play an important role in GCNs, preventing multi-layer networks from collapsing into single-layer networks. However, directly applying commonly used nonlinear activation functions, such as ReLU, to Lorentzian representations can break the manifold constraints of the Lorentz model. It is known that nonlinear activation functions applied in the Poincaré ball model $\mathcal{B}$ can preserve manifold properties: for any $b \in \mathcal{B}$, $\sigma(b) \in \mathcal{B}$. Inspired by this, we project the hyper-

bolic aggregation $\tilde{\mathbf{m}}_i^{\mathbb{L}}$ to the Poincaré ball model to apply the nonlinear activation function, and then project the result back to the Lorentz model:

$$\tilde{\mathbf{h}}_i^{\mathbb{L}} = \pi_{\mathcal{B}\to\mathcal{L}}^K \left(\sigma\left(\pi_{\mathcal{L}\to\mathcal{B}}^K(\tilde{\mathbf{m}}_i^{\mathbb{L}})\right)\right) \tag{10}$$

After the model is trained, an unmasked graph $\mathcal{G}$ is input into the encoder, and the graph representation can be obtained as follows: $H = HSSE(\mathcal{G})$. The representation produced by this layer can be applied to specific downstream tasks.

### 3.4 Hyperbolic Structural Self-Attention Decoder (HSSD)

The decoder aims to map the latent representations obtained from the encoder back to the input, and its design depends on the semantic level (He et al., 2022) of the target input. In graphs, the decoder reconstructs multi-dimensional node features with relatively less informational content. Traditional graph autoencoder methods (GAEs) either do not use a neural decoder or use a simple MLP for decoding, but their expressive capacity is relatively limited. This often leads to the latent representations learned by the encoder being almost identical to the input features. However, learning such trivial latent representations is of no value, as the goal of encoding is to embed the input features into meaningful compressed knowledge.

Meanwhile, because the HSSE stage involves encoding graphs with masked nodes and graphs with masked edges, the latent embeddings inevitably contain noise, especially when the hyperparameters $p1$, $p2$ are large. Therefore, the proposed method uses a more expressive GCN as the decoder and takes the encoder outputs $H^1$ and $H^2$ of $\mathcal{G}^1_{mask}$ and $\mathcal{G}^2_{mask}$ as inputs to two separate decoders, respectively, to learn different decoder parameters. This addresses the issues mentioned above and enhances reconstruction capabilities. The formulas are defined as follows: $\hat{H}_1 = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H_1W_1\right), \hat{H}_2 = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}'\tilde{D}^{-\frac{1}{2}}H_2W_2\right)$, where $\tilde{A} = A+I$, $\tilde{D}_{ii} = \sum_j \tilde{A}_{i,j}$, $I$ is a unit matrix, and $W_i$ are learnable weight matrices. $\sigma$ represents an activation function.

Please note that the decoders are only used during the self-supervised training phase to perform the task of graph feature reconstruction. Thus, the decoder architecture is not inherently tied to the encoder and can utilize any type of GNN.

### 3.5 Loss

For the above graph representation $\hat{H}_1$ obtained from the decoder HSSD, we expect its error from the hyperbolic node feature matrix $X^{\mathbb{L}}$ of the original graph $\mathcal{G}$ to be as small as possible. To get better results, we utilize the **scaled cosine error** as the evaluation criterion for reconstructing node features. The formula is given by Equation(11):

$$\mathcal{L}_{sce} = \frac{1}{|V_{mask}|}\sum_{v_i\in V_{mask}}\left(1 - \frac{\hat{x}_{1,i}^T x_i^{\mathbb{L}}}{\|\hat{x}_{1,i}\|\cdot\|x_i^{\mathbb{L}}\|}\right)^{\gamma}, \quad \gamma \geq 1 \tag{11}$$

, where $V_{mask}$ denotes the set of masked nodes, and the scaling factor $\gamma$ is a hyperparameter that can be tuned on different datasets. For high-confidence predictions, the corresponding cosine error is typically less than 1 and decays to zero faster when the scaling factor $\gamma > 1$.

For the graph representation $\hat{H}_2$ obtained from the decoder HSSD, the connection probability between node embeddings is first calculated using the Fermi-Dirac decoder (Krioukov et al., 2010), given by the formula:

$$p_{fd}(\hat{x}_{2,i}, \hat{x}_{2,j}) := \left[\exp\left(\left(d_{\mathbb{L}}(\hat{x}_{2,i}, \hat{x}_{2,j}) - u\right)/t\right)\right]^{-1} \tag{12}$$

, where $u$ and $t$ are hyper-parameters.

For clarity, the subscript $*_2$ of the referring graph representation $\hat{H}_2$ is omitted in the subsequent formulas. Our goal is to reconstruct the adjacency matrix $A$ of the original graph $\mathcal{G}$. Therefore, the **cross-entropy loss** $\mathcal{L}_{ce}$ is used to maximize the probability of linked nodes in the original graph $\mathcal{G}$, while minimizing the probability of unlinked nodes:

$$\begin{aligned}\mathcal{L}_{ce} = &\frac{1}{|V|}\sum_{e_{ij}=1} -\log\left(p_{fd}(\hat{x}_i, \hat{x}_j)\right) \\ &-\frac{1}{|V|}\sum_{e_{ij}=0}\left(1 - \log\left(p_{fd}(\hat{x}_i, \hat{x}_j)\right)\right)\end{aligned} \tag{13}$$

, where $e_{ij}$ denotes the value of the corresponding position of the neighbor matrix A in the original graph, and $e_{ij} = 1$ represents the existence of an edge between nodes $v_i$ and $v_j$, otherwise it represents the nonexistence of an edge. Due to the sparsity of real-world graph data and considering the training cost, we use negative sampling as an approximation to improve the training efficiency. In the actual training, the same number of edges are sampled for both positive and negative edges.

According to the above argument, the final loss of the overall architecture is the weighted sum of the feature loss $\mathcal{L}_{sce}$ and the structural loss $\mathcal{L}_{ce}$, as defined in equation:$\mathcal{L} = \mathcal{L}_{sce} + \alpha\mathcal{L}_{ce}$, where $\alpha$ is a non-negative hyperparameter used to control the balance between the two losses. This combined loss ensures that the model learns both accurate node features and graph structures, enhancing the overall performance of the graph representation learning.

## 4 Experiments and Analysis

To validate the effectiveness of HDM-GAE, this study compares HDM-GAE with state-of-the-art (SOTA) methods in link prediction and node classification tasks and further ablation experiments.

### 4.1 Link Prediction

The link prediction task aims to predict the existence of edges between nodes in a given network. This task has applications in various fields of critical importance. The goal is to utilize the structural information of the network to accurately infer missing or future links.

| Dataset | PPI | Blog. | Flickr | PubMed | WikiCS | CiteSeer |
|---|---|---|---|---|---|---|
| Nodes | 17598 | 5196 | 7575 | 19717 | 11701 | 3327 |
| Edges | 5429 | 173468 | 242146 | 44338 | 297110 | 4732 |
| Features | 17 | 8189 | 12047 | 500 | 300 | 3703 |
| Classes | 4 | 6 | 9 | 3 | 10 | 6 |

Table 1: Summary of the datasets

**Datasets** In our experiments, we utilize three widely used datasets for link prediction. PPI (Protein-Protein Interaction) is a dataset of the human PPI network. BlogCatalog represents the social network of bloggers listed on the BlogCatalog website (Tan et al., 2023). Flickr represents the social network of users on the photo-sharing platform Flickr (Tan et al., 2023). Table 1 provides detailed statistics for these datasets.

**Baselines** To evaluate the effectiveness of our proposed method, we compare it with the following state-of-the-art methods. (1) several Euclidean graph embedding methods (Kipf and Welling, 2017; Velickovic et al., 2017; Hamilton et al., 2017), i.e., GCN, GAT, and GraphSage; (2) several hyperbolic graph embedding methods, i.e., HGCN and HGAT; (3) some masked self-supervised graph learning methods, namely GraphMAE and MaskGAE; (4) other SOTA methods, GIC (Mavromatis and Karypis, 2021), BGRL (Thakoor et al., 2021), and S2GAE (Tan et al., 2023). For these methods in the comparison, the same experimental setup is followed if the methods have been used for evaluating this task. For methods that have not been formally tested on the link prediction task, we apply them to this task by training an MLP-based decoder, also known as fine-tuning.

| | AUC | | | AP | | | |
|---|---|---|---|---|---|---|---|
| Dataset | Blog. | Flickr | PPI | Blog. | Flickr | PPI | A.R. |
| GCN | 76.44±0.75 | 86.26±0.63 | 79.02±0.47 | 75.56±0.23 | 81.44±0.31 | 79.29±0.37 | 9.67 |
| GAT | 79.38±0.67 | 87.34±0.53 | 78.74±0.36 | 76.13±0.31 | 80.98±0.62 | 78.72±0.46 | 9.33 |
| GraphSage | 76.42±0.82 | 85.39±0.44 | 78.20±0.27 | 75.22±0.49 | 81.78±0.51 | 79.71±0.27 | 10.17 |
| HGCN | 79.84±0.41 | 88.63±0.72 | 80.84±0.41 | 76,81±0.35 | 81.26±0.39 | 80.18±0.40 | 7.50 |
| HGAT | 80.84±0.39 | 89.72±0.52 | 81.78±0.37 | 77.31±0.24 | 82.41±0.54 | 81.44±0.27 | 5.50 |
| GraphMAE | 77.30±0.78 | 88.69±0.44 | 81.31±0.43 | 78.10±0.56 | 83.19±0.29 | 81.19±0.36 | 6.33 |
| MaskGAE | 81.43±0.87 | 91.64±0.35 | 83.94±0.31 | 78.41±0.34 | 84.28±0.32 | 83.23±0.42 | 2.67 |
| GIC | 76.57±0.78 | 89.01±0.83 | 83.47±0.27 | 76.29±0.42 | 83.91±0.53 | 81.54±0.51 | 6.00 |
| BGRL | 77.62±0.79 | 88.30±0.49 | 83.53±0.38 | 77.46±0.36 | 83.47±0.45 | 83.62±0.23 | 5.17 |
| S2GAE | **83.62±0.64** | 90.14±0.38 | 84.61± 0.61 | **78.73±0.29** | 85.43± 0.30 | 82.35±0.34 | 2.17 |
| Prop. | 82.31±0.72 | **92.19±0.53** | **85.36±0.42** | 78.28±0.41 | **86.28±0.47** | **84.75±0.30** | **1.50** |

Table 2: Results of the link prediction task. In each column, bold scores indicate the best results and underlined scores indicate the second best results. A.R. is the abbreviation for average rank and denotes the average rank of methods.

**Evaluation Metrics and Setup** After training, our proposed model can generate node representations for different downstream tasks by feeding the original graph, without masking, through the encoder HSSE and the corresponding decoder HSSD. For link prediction, given an unseen edge, we estimate the probability of its existence by further feeding the representations of its end nodes into the Fermi-Dirac decoder. We use the Area Under the ROC Curve (AUC) and Average Precision (AP) as evaluation metrics. For each dataset, 85%/5%/10% of the edges are randomly allocated for training/validation/testing. Since the encoding and decoding operations are mainly performed in hyperbolic space, the Riemannian Adam optimizer (Kochurov et al., 2020) is chosen for optimization of the parameters. Some baseline results are cited from (Li et al., 2023; Tan et al., 2022; Thakoor et al., 2021; Jin et al., 2020). To avoid randomness, we repeat the experiments 10 times and report the average results and standard deviations.

**Results and Analysis** The experimental results are shown in Table 2. HDM-GAE outperforms all baselines on the AUC metrics for the Flickr and PPI datasets and is close to the top method on the BlogCatalog dataset. The method also performs well on the AP metrics, leading on two datasets and being only 0.45% behind the optimal method on BlogCatalog. These results demonstrate the effectiveness of our framework. The proposed method outperforms previous hyperbolic geometry-based and mask-based self-coding methods on all datasets and metrics. One possible explanation is that the archi-

tecture of double-masking and feature reconstruction for node features and edge structure features plays an important role in the superior performance of link prediction. The performance gap between HDM-GAE and other methods suggests a significant advantage of hyperbolic geometry embedding. To further demonstrate generalizability, we compute and compare the average rankings based on the AUC and AP scores. HDM-GAE shows a significant overall advantage with a leading average ranking, validating its graph representation learning capability and generalization to the link prediction task.

### 4.2 Node Classification

The node classification task involves assigning labels to nodes in a network based on their characteristics and structural properties.

**Datasets** We utilize three widely used datasets for node classification. PUBMED (Yang et al., 2016) is a standard benchmark for describing citation networks. CiteSeer includes scientific publications from the CiteSeer Digital Library. WikiCS (Tang et al., 2024) consists of Wikipedia pages related to computer science. Table 1 provides more details about these datasets.

**Baselines** The selected Baselines are identical to Section 4.1. For these methods in the comparison, the same experimental setup is followed if the methods have been used for evaluating this task. For methods that have not been formally tested on a node classification task, we obtain node representations by training a baseline task. The node representations are then used to train and test a simple L2 regularized logistic regression classifier for evaluating the performance of the node classification task.

**Evaluation Metrics and Setup** We use the accuracy (ACC) metric to evaluate the performance of the node classification method. Accuracy measures the proportion of correctly classified nodes to the total number of nodes. After training, our proposed model can be applied to downstream tasks by generating node representations by feeding the original graph through the encoder HSSE without masking. Specifically, for classification, we directly use the learned node representations to train and test simple l2 regularized logistic regression classifiers to evaluate node-level classification. Public segmentation is used for Citeseer, PubMed, and WikiCS datasets. Some baseline results are quoted from (Tan et al., 2023; Li et al., 2023; Thakoor et al., 2021; Xiao et al., 2022). To avoid randomness, we report the average accuracy with a standard deviation of 10 random initializations.

| | ACC | | | |
|---|---|---|---|---|
| Dataset | CiteSeer | PUBMED | WikiCS | Avg, |
| GCN | 70.75±0.38 | 78.94±0.57 | 77.21±0.14 | 75.63 |
| GAT | 71.92±0.62 | 78.22 ±0.82 | 78.71±0.36 | 76.28 |
| GraphSage | 70.20±1.15 | 77.96±0.54 | 76.93±0.42 | 75.03 |
| HGCN | 72.18±0.32 | 80.27±0.53 | 78.64±0.37 | 77.03 |
| HGAT | 73.06±0.45 | 79.94±0.35 | 79.36±0.43 | 77.45 |
| GraphMAE | 72.49±0.65 | 81.54±0.22 | 78.61±0.51 | 77.55 |
| MaskGAE | 75.21±0.46 | 81.26±0.36 | 80.12±0.41 | 78.86 |
| GIC | **76.94±0.22** | 80.87±0.13 | 79.91±0.26 | 79.24 |
| BGRL | 72.92±0.34 | 79.62±0.18 | 79.41±0.48 | 77.32 |
| S2GAE | 74.23±0.14 | 81.68±0.24 | 80.26±0.31 | 78.72 |
| Prop. | 75.68±0.51 | **82.18±0.37** | **81.85±0.29** | **79.90** |

Table 3: Results of the node classification task. Avg. is the abbreviation for average and denotes the average performance.

**Results and Analysis** The results of the experiment are shown in Table 3. Our method achieves state-of-the-art performance on the PUBMED and WikiCS datasets and outperforms all benchmark methods except GIC on the CiteSeer dataset. It shows that our proposed architecture can effectively combine the properties of hyperbolic geometry and the reconstruction properties of mask self-coding to enhance the model's ability of node attribute feature extraction. The mean value of the scores of each method on all datasets is calculated in the last column of Table 3. It can be seen that our proposed method scores the highest among all the baselines. This indicates the method consistently performs well when dealing with different types of datasets and is indeed an effective architecture for GRL.

### 4.3 Ablation Study

To verify the effectiveness of the main modules in HDM-GAE, we further conducted the following ablation study.

#### 4.3.1 Effect of Hyperbolic and Edge-Mask

We conducted ablation studies to validate the effectiveness of our main modules. We independently removed the hyperbolic geometry and edge mask modules to create simpler architectures. The DM-GAE model implies the removal of the hyperbolic geometry transformation from the overall HDM-GAE architecture, with both attention and aggregation operations running on Euclidean space. The HNM-GAE model removes the edge masks, leaving only node masking and reconstruction. The rest

follows the same experimental setup. For both link prediction and node classification tasks, we display experimental results for a total of four datasets, as shown in Table 4.

| | AUC for Link Prediction | | ACC for Node Class | |
|---|---|---|---|---|
| Dataset | Flickr | PPI | PUBMED | WikiCS |
| DM-GAE | 89.26±0.42 | 82.53±0.22 | 79.38±0.33 | 80.15±0.19 |
| HNM-GAE | 86.74±0.38 | 80.62±0.28 | 81.48±0.29 | 80.87±0.24 |
| HDM-GAE | **92.19±0.53** | **85.36±0.42** | **82.18±0.37** | **81.85±0.29** |

Table 4: Results of ablation studies performed for hyperbolic module and edge mask module.

The ablation experiments demonstrate that the HDM-GAE model outperforms the comparison methods on all datasets, validating the effectiveness of our proposed method. That is, the removal of any module leads to performance degradation. The effect of edge mask removal is particularly significant in the link prediction task, with HNM-GAE performance decreasing by 5.45% on Flickr and 4.74% on PPI. This shows that edge masking and reconstruction are key to capturing structural information and filtering noise, thus improving link prediction. The DM-GAE model without hyperbolic geometry shows a performance degradation of about 1.7-2.9%, particularly on the PubMed dataset for node classification, likely due to its hierarchical structure. These results clarify the role and importance of each module and verify the design concept of HDM-GAE.

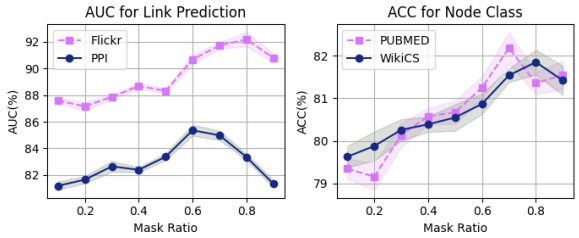


Figure 2: Ablation studies of mask ratio

### 4.3.2 Effect of Mask Ratio

Given the strategy of the mask is crucial for our training task, we further investigate how the masking ratio improves or degrades the model performance. The left and right plots in Fig. 2 show the effect of mask ratio on model performance on the link prediction and node classification tasks, respectively. In most cases, lower mask ratios (e.g., p1 and p2 less than 0.5) are not sufficient to learn useful features, and model performance shows a similar trend of improvement when p1 and p2 are increased within a certain range. The optimal ratio varies from chart to chart. On the PPI dataset, model performance is optimal at the mask ratio of 0.6, and continuing to increase the mask ratio decreases performance. For the dataset PUBMED, model performance is optimal at the mask ratio near 0.7. For the WikiCS dataset and the Flickr dataset, the performance is optimal at the mask ratio of 0.8. We analyze that this difference exhibited by the mask ratio should be able to be linked to the information redundancy in the graph dataset. Large node degrees may lead to information redundancy when only a few nodes and edges are needed to approximately recover the node features. On the contrary, in less redundant datasets, a higher masking rate will not recover the features, thus reducing the performance. In summary, the best performance is achieved when p1 or p2 reaches the interval range of 0.6 to 0.8. Parameters too large or too small may lead to poor performance.

## 5 Conclusions

In this work, we propose a novel self-supervised graph representation learning architecture, named HDM-GAE, for static graph representation learning in hyperbolic spaces. In HDM-GAE, we first map graphs onto hyperbolic spaces, as the geometric advantages of hyperbolic spaces are taken into account. Further, our model forces the model to learn both node features and structure features by jointly performing masking and reconstruction operations on nodes and edges. It fully utilizes the geometric advantage of the hyperbolic space and the masking mechanism in self-supervised learning, which, to the best of our knowledge, is the first study in the field of graph representation learning based on hyperbolic representation and double-masked self-coding mechanism. In addition, the architecture proposes a new graph bi-masking module, a self-attentive coding module for hyperbolic structures with node aggregation directly in hyperbolic space, and so on. Experimental results on six real-world datasets demonstrate the effectiveness of HDM-GAE in graph representation learning and its superiority in downstream node classification and link prediction tasks. For future work, we hope our work will inspire further development of graph embedding and graph mask self-coding techniques in hyperbolic space.

## References


Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. 2022. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 594–604.

Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wan Erjin Zhou, and Yue Gao. 2021. Graph-mlp: Node classification without message passing in graph. *Preprint*, arXiv:2106.04051.

Zhiwu Huang, Ruiping Wang, Shiguang Shan, Luc Van Gool, and Xilin Chen. 2017. Cross euclidean-to-riemannian metric learning with application to face recognition from video. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2827–2840.

Wei Jin, Tyler Derr, Haochen Liu, Yiqi Wang, Suhang Wang, Zitao Liu, and Jiliang Tang. 2020. Self-supervised learning on graphs: Deep insights and new direction. *arXiv preprint arXiv:2006.10141*.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Max Kochurov, Rasul Karimov, and Serge Kozlukov. 2020. Geoopt: Riemannian optimization in pytorch. *arXiv preprint arXiv:2005.02819*.

Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. 2010. Hyperbolic geometry of complex networks. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 82(3):036106.

Jintang Li, Ruofan Wu, Wangbin Sun, Liang Chen, Sheng Tian, Liang Zhu, Changhua Meng, Zibin Zheng, and Weiqiang Wang. 2023. What's behind the mask: Understanding masked graph modeling for graph autoencoders. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1268–1279.

Costas Mavromatis and George Karypis. 2021. Graph infoclust: Maximizing coarse-grain mutual information in graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 541–553. Springer.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30.

Maximillian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International conference on machine learning*, pages 3779–3788. PMLR.

Fragkiskos Papadopoulos, Maksim Kitsak, M Ángeles Serrano, Marián Boguná, and Dmitri Krioukov. 2012. Popularity versus similarity in growing networks. *Nature*, 489(7417):537–540.

Wei Peng, Tuomas Varanka, Abdelrahman Mostafa, Henglin Shi, and Guoying Zhao. 2021. Hyperbolic deep neural networks: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, 44(12):10023–10044.

Frederic Sala, Chris De Sa, Albert Gu, and Christopher Ré. 2018. Representation tradeoffs for hyperbolic embeddings. In *International conference on machine learning*, pages 4460–4469. PMLR.

Qiaoyu Tan, Ninghao Liu, Xiao Huang, Rui Chen, Soo-Hyun Choi, and Xia Hu. 2022. Mgae: Masked autoencoders for self-supervised learning on graphs. *arXiv preprint arXiv:2201.02534*.

Qiaoyu Tan, Ninghao Liu, Xiao Huang, Soo-Hyun Choi, Li Li, Rui Chen, and Xia Hu. 2023. S2gae: Self-supervised graph autoencoders are generalizable learners with graph masking. In *Proceedings of the sixteenth ACM international conference on web search and data mining*, pages 787–795.

Peng Tang, Cheng Xie, and Haoran Duan. 2024. Node and edge dual-masked self-supervised graph representation. *Knowledge and Information Systems*, 66(4):2307–2326.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L Dyer, Remi Munos, Petar Veličković, and Michal Valko. 2021. Large-scale representation learning on graphs via bootstrapping. In *International Conference on Learning Representations*.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. 2017. Graph attention networks. *stat*, 1050(20):10–48550.

Shunxin Xiao, Shiping Wang, Yuanfei Dai, and Wenzhong Guo. 2022. Graph neural networks in node classification: survey and evaluation. *Machine Vision and Applications*, 33(1):4.

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. 2022. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9653–9663.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pages 40–48. PMLR.