Investigating Neighborhood Modeling and Asymmetry Preservation in Digraph Representation Learning

Honglu Zhou, Advith Chegu, Samuel Sohn, Mubbasir Kapadia
Department of Computer Science, Rutgers University, Piscataway, NJ, USA.
{honglu.zhou, ac1771}@rutgers.edu, {sss286, mk1353}@cs.rutgers.edu

Abstract
Graph Neural Networks (GNNs) traditionally exhibit poor performance for directed graphs (digraphs) due to notable challenges in 1) modeling neighborhoods and 2) preserving asymmetry. In this paper, we address these challenges in traditional GNNs by leveraging hyperbolic collaborative learning from multi-ordered and partitioned neighborhoods, and regularizers inspired by socio-psychological factors. Our resulting formalism, Digraph Hyperbolic Network (D-HYPR) learns node representations in hyperbolic space to avoid structural and semantic distortion of real-world digraphs. We conduct comprehensive experimentation on 4 tasks: link prediction, node classification, sign prediction, and embedding visualization. D-HYPR statistically significantly outperforms the current state of the art on a majority of tasks and datasets, while achieving competitive performance otherwise. Our code and data will be available.

Introduction
Directionality is a fundamental characteristic inherent in a multitude of real-world graphs including social networks, webpage networks, and citation networks (Ou et al. 2016). Digraph Representation Learning (DRL) (Tong et al. 2020a) has numerous applications, e.g., community question answering (Sun et al. 2020a 2019), job recommendation (Luo et al. 2019), paper influence prediction (Zhang 2019), software attributes analysis (Zhong, Sun, and Xu 2018), and etc (Zhang et al. 2019). Kenning et al. 2021). Despite the importance of these applications, few methods exist which includes factorization-based (Ou et al. 2016; Sun et al. 2019) and random walk-based (Zhou et al. 2017, Khosla et al. 2019) approaches. However, these methods do not scale to large digraphs (Sun et al. 2019), are sensitive to outliers and noise (Khosla et al. 2019), and do not generalize well across tasks (Ou et al. 2016; Zhou et al. 2017).

Graph Neural Networks (GNNs) have achieved immense success on a wide range of tasks (Zhou et al. 2018; Yan, Xiong, and Lin 2018; Chen et al. 2019; Wang et al. 2021; Nawhal and Mort 2021; Dwivedi et al. 2020). Still, popular GNNs have primarily focused on representation learning for undirected graphs, and been validated solely by experiments on undirected graphs. There are two notable challenges that hinder their effectiveness to digraphs.

Challenge 1: Neighborhood Modelling. The neighbors of a node present distinct characteristics and/or semantics. E.g., connections of in-neighbors and out-neighbors can belong to different types or time periods. In social networks, in-neighbors are commonly known as followers and out-neighbors are the ones that the user follows. In citation networks, in-neighbors of a node could be exiting works cited by the paper by the time the camera-ready version of the paper is submitted, whereas connections of out-neighbors happen after the paper comes out (Getoor 2005). Existing techniques (Kipf and Welling 2016b; Chami et al. 2019; Zhu et al. 2020; Chen et al. 2021) transform digraphs to undirected graphs or only consider the direct out-neighbors in graph convolution for running experiments. Thus, they lose characteristics of the original structure, and result in misleading message passing. GNNs must properly model neighborhoods accounting for properties of digraphs.

Challenge 2: Asymmetry Preservation. Due to the inherent symmetry of the popular measures, such as inner product or distance in the embedding space which produces the same scores for the node pair \((i, j)\) and \((j, i)\), inner-product- or distance-based learning objectives used by prior models are unsuitable for capturing the asymmetric connectivity probabilities for node pairs in digraphs (Salha et al. 2019). Applications that are based on link prediction (Luo et al. 2019) or graph topology learning (Halcrow et al. 2020; Ramezani-Mayani et al. 2019) are particularly affected when models fail to preserve structural asymmetry for digraphs.

Recently, spectral-based GCN approaches (Tong et al. 2020a) have been proposed to address Challenge 1 with respect to modelling neighborhoods for digraphs. However, the learned filters from these methods depend on the Laplacian eigenbasis, which is tied to a graph’s structure. Models trained on a specific structure cannot be directly applied to graphs with different structures (Wu et al. 2020). These models also fail to preserve node-pair asymmetry for Challenge 2, missing a solution for tasks like link prediction by solely experimenting on the node classification task. On the other hand, gravity-inspired graph autoencoders (Salha et al. 2019) have been proposed for addressing Challenge 2, but experimental evidences are only provided for their ability to tackle the directed link prediction problem.

Our proposed Digraph HYPERbolic Network (D-HYPR) addresses both challenges with no compromises. To address Challenge 1, D-HYPR utilizes hyperbolic collaborative learning from multi-ordered and partitioned neigh-
For **Challenge 2**, **D-HYPR** takes advantage of self-supervised learning, via asymmetry-preserving regularizers supported by well-established socio-psychological theories (McPherson, Smith-Lovin, and Cook 2001; Mitzenmacher 2004). Specifically, by leveraging collaborative learning from multi-ordered 4 canonical types of partitioned neighborhoods (Fig. 2 (a)), **D-HYPR** achieves larger receptive fields, and generates node embeddings in a more holistic view. In addition, **D-HYPR** learns node representations of real-world digraphs (which exhibit scale-free and/or hierarchical structures) in hyperbolic space to avoid structural and semantic distortion of node neighborhoods, inspired by advances in Hyperbolic Neural Embedding (Suriš, Liu, and Vondríček 2021; Ganea, Bécsy, and Hofmann 2018b; Sun et al. 2021a; Choudhary et al. 2021). In doing so, **D-HYPR** builds upon prior contributions in undirected graphs that leverage both the expressiveness of GNNs and hyperbolic geometry to learn inductive node representations for graphs that may contain cycles (Zhu et al. 2020; Chami et al. 2019). Lastly, motivated by 2 decomposed causes of link formation, homophily and preferential attachment, we employ 2 regularizers in training **D-HYPR**, which are used in a self-supervised fashion to account for each of the two driving forces of link formation. The regularizers help to boost the performance of downstream tasks.

We propose **D-HYPR** (Fig. 2 (b)), a GNN-based formalism that inductively learns digraph embeddings in hyperbolic space. Our contributions are summarized as follows:

- **D-HYPR** takes into account of the unique node neighborhoods in digraphs, by leveraging multi-scale neighborhood collaboration in hyperbolic space.

- **D-HYPR** respects asymmetric relationships of node-pairs. This is achieved by self-supervised learning guided through sociopsychology-inspired regularizers.

- Extensive experiments on 4 tasks (link prediction, node classification, sign prediction, and embedding visualization) demonstrate the superiority of **D-HYPR** against the state of the art (e.g., Fig. 1). **D-HYPR** generates meaningful embeddings in very low dimensions (Table 3). This added benefit is desirable for large-scale real-world applications by efficiently saving space while preserving effectiveness.

**Related Work**

**Graph Representation Learning (GRL).** GRL methods have evolved from matrix factorization (Jiang et al. 2016), graph kernels (Shervashidze et al. 2011), and random walk-based transductive models (Perozzi, Al-Rfou, and Skiena 2014), into GNNs (Kipf and Welling 2016a; Xu et al. 2018; Beani et al. 2021) which have surpassed these prior methods with strong generalization and inductive learning abilities. We refer readers to comprehensive reviews presented in (Cai, Zheng, and Chang 2018; Wu et al. 2020).

**Directed Graph Embedding.** HOPE (Ou et al. 2016) captures asymmetric transitivity, but cannot be generalized to a variety of tasks (Khosla et al. 2019). APP (Zhou et al. 2017) relies on random walks. ATP (Sun et al. 2019) removes cycles in digraphs and leverages factorization. NERD (Khosla et al. 2019) extracts a source walk and a target walk, and employs a shallow neural model. DGNN (Tong et al. 2020b), DiGCN (Tong et al. 2020a) and MagNet (Zhang et al. 2021a) are recent GNNs that extend spectral-based GCN to digraphs, but are tied to a graph’s Laplacian. DAGNN (Thost and Chen 2021) is proposed for directed acyclic graphs (DAGs), by injecting partial ordering into the GNN design.

**Hyperbolic Embedding Learning.** Researchers have started to equip GRL with such geometrically appropriate inductive biases. However, most non-Euclidean embedding techniques (Nickel and Kiela 2017; 2018; Gu et al. 2018; Law and Stan 2020) only account for the graph structure and do not leverage node features. With this regard, hyperbolic GNN approaches emerge (Chami et al. 2019; Liu, Nickel, and Kiela 2019; Zhang et al. 2021b; Zhu et al. 2020; Suzuki, Takahama, and Onoda 2019; Zhang et al. 2021c; Dai et al. 2021) which utilize the strength of hyperbolic geometries for GRL. We build upon the previous works to learn hyperbolic digraph embeddings.

We provide an extended discussion of related work in Appendix.

**Preliminaries**

**Definition 1. Digraph Representation Learning.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. Each edge $e \in \mathcal{E}$ is an ordered pair $e = (i, j)$ between vertex $i$ and $j$. The adjacency matrix of $\mathcal{G}$ can be denoted as $A = \{0, 1\}^{|\mathcal{V}| \times |\mathcal{V}|}$. $\mathcal{G}$ is a digraph when $\exists (i, j), A_{i,j} \neq A_{j,i}$.
Nodes are described by the feature matrix $X^{0,E} \in \mathbb{R}^{|V| \times d}$, i.e., each node $i \in V$ has a $d$-dimensional Euclidean feature $x_i^{0,E}$. The superscript $E$ indicates the vector lies in an Euclidean space, and $H$ denotes a hyperbolic vector. 0 denotes the input layer. The goal of DRL is to learn a mapping $f$ that maps nodes to embedding vectors $(d' \ll |V|)$.

$$f : (V, E, (x_i^{0,E})_{i \in V}) \rightarrow Z \in \mathbb{R}^{|V| \times d'} \quad (1)$$

The embeddings should capture both structural and semantic information and be valuable for downstream tasks.

**Definition 2. The Poincaré Ball Model.** The Poincaré ball model (Ganea, Bécsigneul, and Hofmann 2018b) $(\mathbb{D}_c^n, g_c)$ is defined by the $n$-dimensional manifold $\mathbb{D}_c^n = \{ x \in \mathbb{R}^n : c \| x \| < 1 \}$ equipped with the Riemannian metric: $g_c = \lambda_c^{-2} g^{E}$, where $\lambda_c := \frac{2}{1-c \| x \|^2}$ is the Euclidean metric tensor, and $c > 0$ (we call $-c$ as the curvature). $\mathbb{D}_c^n$ is the open ball of radius $1/\sqrt{c}$. The connections between hyperbolic space and tangent space are established by the exponential map $\exp_c : \mathbb{T}_c \mathbb{D}_c^n \rightarrow \mathbb{D}_c^n$ and logarithmic map $\log_c : \mathbb{D}_c^n \rightarrow \mathbb{T}_c \mathbb{D}_c^n$:

$$\exp_c(y) = x \oplus_c (\tanh(\sqrt{c} \frac{\| v \|}{2}) \frac{v}{\sqrt{\| v \|}}) \quad (2)$$

$$\log_c(y) = \frac{2}{\sqrt{c} \lambda_c} \tanh^{-1}(\sqrt{c} \| x \oplus_c y \|) - x \oplus_c y \quad (3)$$

where $x, y \in \mathbb{D}_c^n$, $v \in \mathbb{T}_c \mathbb{D}_c^n$, and $\oplus_c$ is Möbius addition,

$$x \oplus_c y := (1 + 2c(x \cdot y) + c \| y \|^2) x + (1 - c \| x \|^2) y \quad (4)$$

The Möbius scalar multiplication (Equa. 5) and Möbius matrix multiplication of $x \in \mathbb{D}_c^n \setminus \{0\}$ (Equa. 6) are,

$$r \odot_c x := \frac{1}{\sqrt{c}} \tanh(r \tanh^{-1}(\sqrt{c} \| x \|)) \frac{x}{\| x \|} \quad (5)$$

$$M \odot_c x := (1/\sqrt{c}) \tanh(\| Mx \| / \| x \|) \tanh^{-1}(\sqrt{c} \| x \|)) \frac{Mx}{\| Mx \|} \quad (6)$$

where $r \in \mathbb{R}$ and $M \in \mathbb{R}^{m \times n}$. The induced distance function on $(\mathbb{D}_c^n, g_c)$ is given by

$$d_{\mathbb{D}_c^n}(x, y) = (2/\sqrt{c}) \tanh^{-1}(\sqrt{c} \| x \oplus_c y \|) \quad (7)$$

**Graph Embedding Layers**

**Hyperbolic Feature Transformation**

Hyperbolic feature transformation is performed by

$$h_{i}^{\ell,H} = W^\ell \odot_{d-1} x_i^{\ell-1,H} + b \quad (8)$$

where $W^\ell \in \mathbb{R}^{d \times d'}$ is the weight matrix, and $b \in \mathbb{R}^{d'}$ denotes the bias (both are learned). We employ unique trainable curvature at each layer to obtain suitable hyperbolic space to account for different depths of the neural network.

**Hyperbolic Neighbor Aggregation.** We then leverage the bridging between the hyperbolic space and the tangent space to perform neighbor aggregation (Zhang et al. 2021b; Zhu et al. 2020), resulting in $h_{i}^{\ell,H} \in \mathbb{D}_c^{d-1}$,

$$h_{i}^{\ell,H} = \exp^{\ell-1}_{0} \sum_{j \in (i)\cap \mathcal{N}(i)} e_{ij} \log^{\ell-1}_{0}(m_{j}^{\ell,H}) \quad (9)$$

**Neighborhood Collaborative Learning**

Caused by semantics of directed edges, neighbors of a node can be implicitly partitioned into non-disjoint groups. Consideration of these neighborhoods is critical for learning a holistic node embedding in a digraph. This is because each neighborhood can reflect a distinct aspect relating to the node (Ma et al. 2019a). E.g., in a social networking platform, a popular user’s in-neighbors and out-neighbors can have completely different degrees of relationships cohesion to the popular user. Furthermore, users who share common in-neighbors with this user and those who share common out-neighbors can indicate additional contexts.

Our method leverages this inductive bias exhibited in real-world digraphs, through collaborative learning among the aforementioned 4 canonical types of neighborhoods in hyperbolic space. In addition, D-HYPR achieves larger receptive fields by taking account of the impact of multi-ordered neighbors (Yang et al. 2017; Tong et al. 2020b). D-HYPR generates a representation for each type of neighborhoods,
Figure 2: (a) Multi-ordered and partitioned neighborhoods. We define four types of k-order proximity matrix (shown in the figure with \(k = 1, 2\) with respect to node 1) to incorporate the pertinent subsets of neighbors and multi-scale information. (b) Methodology overview. D-HYPR learns node representation from each neighborhood in hyperbolic space with Hyperbolic Graph Embedding Layers. Hyperbolic Neighborhood Aggregation further enables a closer collaboration of neighborhoods. D-HYPR respects asymmetric relationships of nodes with the hyperbolic Fermi-Dirac and Gravity regularizers.

Each serving as one representation slice that eventually comrises the final comprehensive node embedding through collaboratively learning in the hyperbolic space.

**Neighbor Partition.** 4 types of k-order proximity matrix are defined (Fig. 2(a)). Formally, k-order proximity in terms of:

1. **diffusion in** \(A_{\text{in}}^k\),
   \[
   A_{\text{in}}^k(i, j) = \mathbf{1} \cdot \sum_{p \in V} A_{\text{in}}^{k-1}(i, p) \cdot A_{\text{in}}^1(p, j)
   \]  
   (11)
   where \(A_{\text{in}}^1 = A^T\), \(\mathbf{1}\) is the indicator function. \(A_{\text{in}}^k(i, j) = \mathbf{1}\) if there is a directed path from node \(j\) to node \(i\) of length exactly \(k\).

2. **diffusion out** \(A_{\text{out}}^k\),
   \[
   A_{\text{out}}^k(i, j) = \sum_{p \in V} A_{\text{out}}^{k-1}(i, p) \cdot A_{\text{out}}^1(p, j)
   \]  
   (12)
   where \(A_{\text{out}}^1 = A\). \(A_{\text{out}}^k(i, j) = \mathbf{1}\) if there is a directed path from node \(i\) to node \(j\) of length exactly \(k\).

3. **common in** \(A_{\text{in}}^k\),
   \[
   A_{\text{in}}^k(i, j) = \sum_{p \in V} A_{\text{in}}^k(i, p) \cdot A_{\text{in}}^k(p, j)
   \]  
   (13)
   where \(i \neq j \neq p\). \(A_{\text{in}}^k(i, j) = \mathbf{1}\) if node \(i\) and node \(j\) have a common in-neighbor \(k\) hops away.

4. **common out** \(A_{\text{out}}^k\),
   \[
   A_{\text{out}}^k(i, j) = \sum_{p \in V} A_{\text{out}}^k(i, p) \cdot A_{\text{out}}^k(p, j)
   \]  
   (14)
   where \(i \neq j \neq p\). \(A_{\text{out}}^k(i, j) = \mathbf{1}\) if node \(i\) and node \(j\) have a common out-neighbor \(k\) hops away.

**Multi-Scale Neighborhood Learning.** For a given non-zero integer \(K\), we compute the 4 types of \(k\)-order proximity matrix for \(k = 1\) to \(K\) (as a preprocessing step). This allows for the capture of multi-scale node proximity (Abu-El-Haija et al. 2020). These \(k\)-order proximity matrices replace the original adjacency matrix \(A\) to provide a wider range of neighborhood to Hyperbolic Graph Embedding Layers.

**Neighborhood Collaboration.** We then apply Hyperbolic Neighborhood Aggregation to enable a closer collaboration among the neighborhoods. Here, we view the \(4K\) output hyperbolic vectors from the Hyperbolic Graph Embedding Layers as representations of \(4K\) neighbors of the anchor node \(i\). We consider \(A_{\text{fuse}}^k\), which is the average of these \(4K\) vectors, as the initial representation of node \(i\) before hyperbolic neighborhood collaboration. We then apply Eq. (9). We use the learned curvature \(-c_i\) from the last hyperbolic graph embedding layer \(l\), and use equal aggregation weights \(\frac{1}{4K+1}\) (\(A_{\text{fuse}}^k\) itself is included as a neighbor in order to enforce skip connection). The resulting output, \(z_i^{l,H}\), is the final hyperbolic embedding of node \(i\). Hyperbolic Neighborhood Aggregation can encourage a better utilization of neighborhoods by synthesizing intermediate representations learned in a neighborhood-level in hyperbolic space.

**Self-Supervised Learning with Asymmetry-Preserving Regularizers**

Homophily and preferential attachment are two driving forces of link formation according to sociopsychological theories. Homophily (McPherson, Smith-Lovin, and Cook 2001) explains the similarity factor in link formation and is portrayed as “birds of a feather flock together”, whereas
preferential attachment (Mitzenmacher, 2004) characterizes the connectivity factor: the link formation likelihood is asymmetric and determined by individual connectivity. To model these two decomposed causes of link formation, we utilize 2 regularizers to predict directed edges when training D-HYPR, thus allow it to respect asymmetry in link formation of digraphs by learning it as a self-supervised task.

We first adopt the Fermi-Dirac decoder in [Krioukov et al., 2010] as a regularizer to reinforce the learning of an appropriate node-pair distance in the hyperbolic embedding space (which well models homophily). Hyperbolic Fermi-Dirac decoder defines the likelihood of a node pair \((i, j)\) as

\[
p(i, j) = \frac{1}{1 + e^{\left(d_{L^2}^{\gamma}(z_i^{L^2}, z_j^{L^2})^2 - r\right)/t}} + 1
\]

where \(r = 2\) and \(t = 1\) (default), and \(d_{L^2}^{\gamma}(\cdot, \cdot)\) is the hyperbolic distance (Eq. [7]).

We further preserve the individual asymmetric node connectivity by learning an additional 1-dimensional mass for each node. This design is elegantly derived from Newton’s theory of universal gravitation (Newton, 1833): each particle in the universe attracts the other particle through gravity, which is proportional to their masses, and inversely proportional to their distance. The learnable node mass is flexible, and it encompasses many centrality measures including Betweenness (Brandes, 2005), Pagerank (Page et al., 1999) and Katz (Katz, 1953). It is also capable of providing explainable visualizations (Salha et al., 2019). To incorporate this idea to D-HYPR basing in hyperbolic space instead of Euclidean, we map \(z_i^{L^2}\) to the tangent space of the origin with the logarithmic map (i.e., \(z_i^{L^2} = \log_0^L(z_i^{L^2})\)), and then employ an Euclidean linear layer to learn \(m_i \in \mathbb{R}\) (mass of node \(i\)). Likelihood of node pair \((i, j)\) is computed by

\[
p(i, j) = \gamma \left( m_j - \lambda \log \left( d_{L^2}^{\gamma}(z_i^{L^2}, z_j^{L^2})^2 \right) \right),
\]

where \(\gamma\) denotes the sigmoid function, and \(\lambda \in \mathbb{R}\) is a hyper-parameter that weights the relative importance of the symmetric embedding distance to the asymmetric node relationships. \(p(i, j) \neq p(j, i)\).

Eq. (10) and (15) both serve as self-supervised regularizers by minimize the binary cross-entropy loss with negative sampling to estimate the likelihood of each node pair. However, the two are placed in different depths of training. Eq. (15) is used. Thus, even though \(d_{L^2}^{\gamma}(\cdot, \cdot)\) also appears in Eq. (16), we empirically find that Eq. (15) provides auxiliary benefits because it allows the model to better construct the final hyperbolic embedding space. Eq. (16) is more designated to learn mass in order to model the asymmetric node connectivity.

Experimental Results

Extensive experiments are conducted to verify the effectiveness of D-HYPR. Our evaluations are rigorous (repeated runs over different random seeds) and reproducible. Please see more details in Appendix.

Datasets. We use open access digraph datasets of varied size and category (Table 1). In order to obtain reliable results, we create numerous splits for each dataset and task, because recent surveys (Dwivedi et al., 2020) find the split of dataset will greatly affect the method performance.

Tasks & Metrics. We use the following tasks and metrics.

- **Link Prediction (LP).** LP demonstrates a method’s capability in modeling asymmetric node connectivity, as a binary classification task of discriminating the missing edges from the fake ones. Metrics are AUC (Area under the ROC Curve) and AP (Average Precision).
- **Semi-supervised Node Classification (NC).** In NC, each dataset contains only 20 labeled nodes for each class, which requires use of the graph structure for predicting the labels of remaining nodes (Jong et al., 2020a).
- **Link Sign Prediction (SP).** Many real-world graphs are signed networks, e.g., social networks that allow trust and distrust relationships among users. We use the Wiki dataset to evaluate the accuracy in predicting attributes of directed edges representing votes (Kim et al., 2018).
- **Embedding Visualization (EV).** EV investigates the expressiveness of the methods qualitatively. We visualize node representations in projected 2D space using t-SNE (Van der Maaten and Hinton, 2008). The embedding vectors are obtained from the NC task. Hyperbolic embedding is mapped to the Euclidean space before t-SNE.

| Task | # Nodes | # Edges | Nodes | Edge | Degree | Avg. | Max |
|------|---------|---------|-------|------|--------|------|-----|
| LP Reciprocity | | | | | | | |
| Wiki | 7,115 | 103,689 | User | Vote | 7,115 | 0.84% |

Table 1: Ranges of datasets. Reciprocity is a measure of the likelihood of nodes to be mutually linked. Label rate is the fraction of nodes labeled for training.
Table 2: Results of Link Prediction in Digraphs with 32-dimensional node embeddings. In all tables, the best score is bolded, the second best is underlined, and the third best is in italic. Result (in percentage) on each dataset of each method is repeated 100 times (10 runs on 10 different splits). We list the best and the average results (the average is shown in gray). * indicates statistically superior performance of the best to the second best at a significance level of 0.01 using a standard paired t-test.

Table 3: Results of Link Prediction in Digraphs with low dimensionality of node embeddings (100 runs the same as in Table 2).
Figure 3: (a) Accuracy on the Semi-supervised Node Classification task by varying the ratio of nodes labeled for training. D-HYPR consistently outperforms the baselines by a large margin. (b) Parameter sensitivity analysis in terms of $\lambda$ and $K$.

Table 4: Experimental results on Semi-supervised Node Classification in Digraphs. Result of each method is averaged over 20 different splits following (Tong et al. 2020a).

| Model     | Cora-ML | CiteSeer |
|-----------|---------|----------|
| MLP       | 61.63 ± 1.8 | 53.18 ± 1.6 |
| ChebNet   | 64.02 ± 1.5 | 56.46 ± 1.4 |
| GCN       | 53.11 ± 0.8 | 54.36 ± 0.5 |
| SGC       | 51.14 ± 0.6 | 44.07 ± 3.5 |
| APPNP     | 70.07 ± 1.1 | 65.39 ± 0.9 |
| InfoMax   | 58.00 ± 2.4 | 60.51 ± 1.7 |
| GAT       | 72.06 ± 0.9 | 63.19 ± 0.7 |
| HNN       | 71.91 ± 0.9 | 63.03 ± 0.6 |
| HGNN      | 62.49 ± 2.6 | 56.10 ± 2.2 |
| SIGN      | 76.45 ± 1.5 | 59.02 ± 2.3 |
| DGCN      | 66.47 ± 0.9 | 60.69 ± 0.4 |
| DGGCN     | 75.02 ± 0.5 | 66.06 ± 0.4 |

D-HYPR (ours) 82.19 ± 1.3 70.66 ± 1.2 Relative Gains (%) 2.38 6.88

Table 5: Experimental results on Wiki.

HGCN has the closest results to D-HYPR on CiteSeer. More visualizations are available in Appendix.

Parameter Sensitivity. D-HYPR uses 2 novel hyper-parameters: $\lambda$ (in Eq. 16) weights the relative importance of the symmetric embedding distance to the asymmetric node relationships, and $K$ determines the maximum order of the proximity matrix. We first examine how $\lambda$ affects the performance of D-HYPR by varying $\lambda$ from 0.25 to 10 (Fig. 3 (b)). Larger $\lambda$ places more value on the symmetric embedding distance (which models ‘homophily’), whereas a smaller $\lambda$ emphasizes the asymmetric node connectivity (which characterizes ‘preferential attachment’). The performance of D-HYPR first increases with $\lambda$ and then decreases. Using the $\lambda$ that produces the best result, we then vary $K$. As shown in the right side of Fig. 3 (b), better results are obtained when $K$ is larger, which means a wider receptive field and more scale information. However, making $K$ too large can lead to feature dilution.

Ablation Study. We perform an ablation study with the setting that produces the best result in the Parameter Sensitivity study. As shown in Table 6, removing any neighborhood harms the performance. Self-supervision largely helps. Neighborhood collaboration plays a key role, and is facilitated by the Gravity regularizer more than the Fermi-Dirac regularizer. All ablations have a lower accuracy than our full model, showcasing that the ablated components work together to increase the learning abilities.

Table 6: Ablation study that demonstrates the individual contribution of components in D-HYPR.

| Ablation       | Cora-ML | CiteSeer |
|----------------|---------|----------|
| D-HYPR − $A_{cin}$ | 82.11 ± 1.2 | 68.72 ± 1.2 |
| D-HYPR − $A_{dout}$ | 81.33 ± 1.4 | 69.10 ± 0.9 |
| D-HYPR − $A_{cin}$ | 81.86 ± 1.6 | 69.98 ± 1.0 |
| D-HYPR − $A_{dout}$ | 81.74 ± 1.8 | 69.84 ± 1.3 |
| D-HYPR − Collaboration | 82.03 ± 1.1 | 70.13 ± 1.5 |
| D-HYPR − Gravity | 79.21 ± 1.5 | 68.58 ± 1.3 |
| D-HYPR − Fermi-Dirac | 82.05 ± 1.3 | 70.03 ± 1.2 |
| D-HYPR − Self-Supervision | 78.15 ± 2.1 | 67.85 ± 1.9 |
| D-HYPR | 82.19 ± 1.3 | 70.66 ± 1.2 |

Relative Gains (%) 36.14 0.09

In this paper, we propose D-HYPR: Digraph Hyperbolic Network, a GNN-based formalism for Digraph Representation Learning (DRL). D-HYPR leverages the inductive biases exhibited in real-world digraphs; it utilizes multi-scale neighborhood collaboration in hyperbolic space and self-supervised learning with asymmetry-preserving regularizers. Through extensive experimentation, we empirically demonstrate the superior representation capability of D-HYPR. One limitation of D-HYPR is the increased number of parameters which comes with the use of multiple neighborhoods. As future work, we would like to explore automatic and dynamic neighborhood partition, as well as parameter-sharing mechanisms to address the current limitation of D-HYPR. Besides, large-scale novel applications of D-HYPR are revenues worth exploration.

Conclusion
References

Abu-El-Haija, S.; Kapoor, A.; Perozzi, B.; and Lee, J. 2020. N-gcn: Multi-scale graph convolution for semi-supervised node classification. In uncertainty in artificial intelligence, 841–851. PMLR.

Bachmann, G.; Bécigneul, G.; and Ganea, O. 2020. Constant curvature graph convolutional networks. In International Conference on Machine Learning, 486–496. PMLR.

Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. 2018. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261.

Beani, D.; Passaro, S.; Létourneau, V.; Hamilton, W.; Corso, G.; and Liò, P. 2021. Directional graph networks. In International Conference on Machine Learning, 748–758. PMLR.

Bläsius, T.; Friedrich, T.; and Katzmann, M. 2021. Force-Directed Embedding of Scale-Free Networks in the Hyperbolic Plane. In 19th International Symposium on Experimental Algorithms (SEA 2021). Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In Neural Information Processing Systems (NIPS), 1–9.

Brandes, U. 2008. On variants of shortest-path betweenness centrality and their generic computation. Social Networks, 30(2): 136–145.

Bronstein, M. M.; Bruna, J.; LeCun, Y.; Szlam, A.; and Vandergheynst, P. 2017. Geometric deep learning: going beyond euclidean data. IEEE Signal Processing Magazine, 34(4): 18–42.

Cai, H.; Zheng, V. W.; and Chang, K. C.-C. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Transactions on Knowledge and Data Engineering, 30(9): 1616–1637.

Chamberlain, B. P.; Clough, J.; and Deisenroth, M. P. 2017. Neural embeddings of graphs in hyperbolic space. arXiv preprint arXiv:1705.10359.

Chami, I.; Gu, A.; Nguyen, D. P.; and Ré, C. 2021. HoroPCA: Hyperbolic Dimensionality Reduction via Horospherical Projections. In International Conference on Machine Learning, 1419–1429. PMLR.

Chami, I.; Wolf, A.; Juan, D.-C.; Sala, F.; Ravi, S.; and Ré, C. 2020. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 6901–6914.

Chami, I.; Ying, Z.; Ré, C.; and Leskovec, J. 2019. Hyperbolic graph convolutional neural networks. In Advances in neural information processing systems, 4868–4879.

Chen, J.; Qin, J.; Shen, Y.; Liu, L.; Zhu, F.; and Shao, L. 2020. Learning Attentive and Hierarchical Representations for 3D Shape Recognition. In Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16, 105–122. Springer.

Chen, M.; Yang, Q.; Tang, X.; et al. 2007. Directed Graph Embedding. In IJCAI, 2707–2712.

Chen, W.; Han, X.; Lin, Y.; Zhao, H.; Liu, Z.; Li, P.; Sun, M.; and Zhou, J. 2021. Fully Hyperbolic Neural Networks. arXiv preprint arXiv:2105.14686.

Chen, Y.; Rohrbach, M.; Yan, Z.; Shuicheng, Y.; Feng, J.; and Kalantidis, Y. 2019. Graph-based global reasoning networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 433–442.

Choudhary, N.; Rao, N.; Katariya, S.; Subbian, K.; and Reddy, C. K. 2021. Self-Supervised Hyperboloid Representations from Logical Queries over Knowledge Graphs. In Proceedings of the Web Conference 2021, 1373–1384.

Dai, J.; Wu, Y.; Gao, Z.; and Jia, Y. 2021. A Hyperbolic-to-Hyperbolic Graph Convolutional Network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 154–163.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016a. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems, 29: 3844–3852.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016b. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems, 29: 3844–3852.

Dhingra, B.; Shallue, C. J.; Norouzi, M.; Dai, A. M.; and Dahl, G. E. 2018. Embedding text in hyperbolic spaces. arXiv preprint arXiv:1806.04313.

Dwi Reid, V. P.; Joshi, C. K.; Laurent, T.; Bengio, Y.; and Bresson, X. 2020. Benchmarking graph neural networks. arXiv preprint arXiv:2003.00982.

Ganea, O.; Bécigneul, G.; and Hofmann, T. 2018a. Hyperbolic entailment cones for learning hierarchical embeddings. In International Conference on Machine Learning, 1646–1655. PMLR.

Ganea, O.; Bécigneul, G.; and Hofmann, T. 2018b. Hyperbolic neural networks. In Advances in neural information processing systems, 5345–5355.

Getoor, L. 2005. Link-based classification. In Advanced methods for knowledge discovery from complex data, 189–207. Springer.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 855–864.

Gu, A.; Sala, F.; Gunel, B.; and Ré, C. 2018. Learning mixed-curvature representations in product spaces. In International Conference on Learning Representations.

Gulcehre, C.; Denil, M.; Malinowski, M.; Razavi, A.; Pascanu, R.; Hermann, K. M.; Battaglia, P.; Bapst, V.; Raposo, D.; Santoro, A.; et al. 2018. Hyperbolic attention networks. arXiv preprint arXiv:1805.09786.

Halcrow, J.; Mosoi, A.; Ruth, S.; and Perozzi, B. 2020. GraLe: designing networks for graph learning. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2523–2532.
Katz, L. 1953. A new status index derived from sociometric analysis. *Psychometrika*, 18(1): 39–43.

Khosla, M.; Leonhardt, J.; Nejdl, W.; and Anand, A. 2019. Node representation learning for directed networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 395–411. Springer.

Khrulkov, V.; Mirvakhabova, L.; Ustinova, E.; Oseledets, I.; and Lempitsky, V. 2020. Hyperbolic image embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6418–6428.

Kim, J.; Park, H.; Lee, J.-E.; and Kang, U. 2018. Side: representation learning in signed directed networks. In *Proceedings of the 2018 World Wide Web Conference*, 509–518.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kipf, T. N.; and Welling, M. 2016a. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N.; and Welling, M. 2016b. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Klicpera, J.; Bojchevski, A.; and Günnemann, S. 2018. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*.

Klicpera, J.; Groß, J.; and Günnemann, S. 2020. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*.

Kriege, N. M.; Johansson, F. D.; and Morris, C. 2020. A survey on graph kernels. *Applied Network Science*, 5(1): 1–42.

Krioukov, D.; Papadopoulos, F.; Kitsak, M.; Vahdat, A.; and Boguná, M. 2010. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3): 036106.

Kyriakis, P.; Fostiropoulos, I.; and Bogdan, P. 2020. Learning Hyperbolic Representations of Topological Features. In *International Conference on Learning Representations*.

Law, M. T.; and Stam, J. 2020. Ultrahyperbolic Representation Learning. *Advances in neural information processing systems (NeurIPS 2020)*.

Liang, R.; Fu, W.; Wen, L.; Hao, S.; and Hong, R. 2016. Dimensionality reduction on anchor graph with an efficient locality preserving projection. *Neurocomputing*, 187: 109–118.

Luo, H.; Ma, S.; Selvaraj, A. J. B.; and Sun, Y. 2019. Learning job representation using directed graph embedding. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*, 1–5.

Ma, J.; Cui, P.; Kuang, K.; Wang, X.; and Zhu, W. 2019a. Disentangled graph convolutional networks. In *International Conference on Machine Learning*, 4212–4221. PMLR.

Ma, Y.; Hao, J.; Yang, Y.; Li, H.; Jin, J.; and Chen, G. 2019b. Spectral-based graph convolutional network for directed graphs. *arXiv preprint arXiv:1907.08990*.

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1): 415–444.

Mitzenmacher, M. 2004. A brief history of generative models for power law and lognormal distributions. *Internet mathematics*, 1(2): 226–251.

Monti, F.; Boscaini, D.; Masci, J.; Rodola, E.; Svoboda, J.; and Bronstein, M. M. 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5115–5124.

Nawhal, M.; and Mori, G. 2021. Activity Graph Transformer for Temporal Action Localization. *arXiv preprint arXiv:2101.08540*.

Newton, I. 1833. *Philosophiae naturalis principia mathematica*, volume 2. typis A. et JM Duncan.

Nickel, M.; and Kiela, D. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems (NeurIPS 2017)*, 6338–6347.

Nickel, M.; and Kiela, D. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, 3779–3788. PMLR.

Ou, M.; Cui, P.; Pei, J.; Zhang, Z.; and Zhu, W. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 1105–1114.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710.

Ramezani-Mayiami, M.; Hajimiradsegh, M.; Skretting, K.; Blum, R. S.; and Poor, H. V. 2019. Graph topology learning and signal recovery via Bayesian inference. In *2019 IEEE Data Science Workshop (DSW)*, 52–56. IEEE.
Zhang, Y.; Wang, X.; Shi, C.; Liu, N.; and Song, G. 2021c. Lorentzian Graph Convolutional Networks. In Proceedings of the Web Conference 2021, 1249–1261.
Zhong, S.; Sun, K.; and Xu, H. 2018. Learning Embeddings of Directed Networks with Text-Associated Nodes—with Applications in Software Package Dependency Networks. arXiv preprint arXiv:1809.02270.
Zhou, C.; Liu, Y.; Liu, X.; Liu, Z.; and Gao, J. 2017. Scalable graph embedding for asymmetric proximity. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 31.
Zhou, J.; Cui, G.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; and Sun, M. 2018. Graph neural networks: A review of methods and applications. arXiv preprint arXiv:1812.08434.
Zhu, S.; Pan, S.; Zhou, C.; Wu, J.; Cao, Y.; and Wang, B. 2020. Graph Geometry Interaction Learning. In Advances in Neural Information Processing Systems.
Appendix

Euclidean Graph Neural Network
The spatial-based GNN layer can be interpreted as performing message propagation and updating on nodes. Let \( \mathcal{N}(i) = \{ j : (i, j) \in E \} \) denote the set of neighbors of \( i \in V \). For layer \( \ell \), suppose its input node features from layer \( \ell - 1 \) have dimension \( F^{\ell-1} \) and its output node features have dimension \( F^{\ell} \). Then, \( W^{\ell} \in \mathbb{R}^{F^{\ell} \times F^{\ell-1}} \) denotes the weights and \( b^{\ell} \in \mathbb{R}^{F^{\ell}} \) denotes the bias. \( \sigma(\cdot) \) is a non-linear activation function. The message passing rule at layer \( \ell \) consists of 1) Feature Transformation:

\[
\mathbf{m}_{i}^{\ell,E} = W^{\ell} \mathbf{x}_{i}^{\ell-1,E} + b^{\ell}
\]

and 2) Neighbor Aggregation:

\[
\mathbf{x}_{i}^{\ell,E} = \sigma \left( \sum_{j \in \mathcal{N}(i)} e_{ij} \mathbf{m}_{j}^{\ell,E} \right).
\]

Aggregation weight \( e_{ij} \in \mathbb{R} \) can be computed with various mechanisms (Hamilton, Ying, and Leskovec 2017; Velickovic et al. 2017). Message passing is performed by multiple layers to learn node embeddings by propagating messages over neighbors.

The spatial-based GNNs have been favored due to their generalization and inductive learning capabilities. However, Euclidean space does not provide the most powerful geometrical representations when input data exhibits a highly complex non-Euclidean latent anatomy (Bronstein et al. 2017; Chamberlain, Clough, and Deisenroth 2017). Because the volume of nodes grows exponentially with distance from a central node, hyperbolic space is more suitable than the Euclidean space for embedding real-world graphs that are scale-free (Sala et al. 2018). This motivates our investigation of hyperbolic representations.

Hyperbolic Geometry
A hyperbolic space in \( n \) dimensions is the \( n \)-dimensional Riemannian manifold with constant negative sectional curvature. A Riemannian manifold is a pair \( (\mathcal{M}, g) \), where \( \mathcal{M} \) is a real and smooth manifold, and \( g = (g_x)_{x \in \mathcal{M}} \) is a Riemannian metric. Formally, \( g \) is a family of smoothly varying inner products on the tangent spaces, \( g_x : T_x\mathcal{M} \times T_x\mathcal{M} \to \mathbb{R} \), where the tangent space \( T_x\mathcal{M} \) is a \( n \)-dimensional vector space and can be seen as a first order local approximation of \( \mathcal{M} \) around point \( x \).

There exist several models of hyperbolic space such as the Poincaré ball or the Hyperboloid model (a.k.a. the Minkowski or the Lorentz model). We provide formulations of our method based on the Poincaré ball model for its practical popularity (Liu et al. 2020) (the most commonly used in gradient-based learning (Suris, Liu, and Vondr´ak 2021)), benefits in visualization (Chami et al. 2019), and relatively simple constraint on the representations (Zhu et al. 2020; Sala et al. 2018). Our method is compatible with the other models of hyperbolic space. The Poincaré ball model has closed-form expressions for basic objects (e.g., distance) and operations (e.g., addition, multiplication) derived by (Ganea, B´ecigneul, and Hofmann 2018).

Extended Discussion on Related Work

Graph Representation Learning (GRL).
GRL is an effective and efficient solution to address the high computation and space costs of most graph analytical tasks (Cai, Zheng, and Chang 2018; Dwivedi et al. 2020). As a common setting, graph data is converted into a low dimensional space, where each node is represented as a vector such that the graph structural information and semantic properties are maximally preserved (Cai, Zheng, and Chang 2018). As the field grows, GRL methods have evolved from matrix factorization (Jiang et al. 2016), graph kernels (Shervashidze et al. 2011), and random walk-based transductive models (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016), into GNNs (Zhou et al. 2018) which have surpassed these prior methods with strong generalization and inductive learning abilities. GNNs generalize convolutions to the graph domain. Categories of GNNs include 1) spectral-based approaches (Kipf and Welling 2016a; Defferrard, Bresson, and Vandergheynst 2016b) that stack spectral filters, and 2) spatial-based approaches (Velickovic et al. 2017; Monti et al. 2017; Hamilton, Ying, and Leskovec 2017; Xu et al. 2018) that achieve graph convolutions through neighbor feature aggregation (i.e., message passing). We refer readers to comprehensive reviews on GRL presented in (Zhou et al. 2018; Cai, Zheng, and Chang 2018; Kriege, Johansson, and Morris 2020; Wu et al. 2020).

Directed Graph Embedding.
Despite the aforementioned remarkable advances in GRL, current popular GRL approaches (Beani et al. 2021; Kipf and Welling 2016a; Velickovic et al. 2017; Chami et al. 2019; Zhu et al. 2020; Zhang et al. 2021b; Beani et al. 2021) have primarily focused on undirected graph embedding, validated solely by experiments on undirected graphs (Zhang et al. 2021a). There are comparatively few works that have addressed digraph embedding. As one of the early attempts, DGE (Chen et al. 2007) uses the transition probability, together with the stationary distribution of Markov random walks, to preserve the locality property of nodes in digraphs. HOPE (Ou et al. 2016) captures asymmetric transitivity with high-order proximity measures like Katz (Katz 1953), but relies on the low rank assumption of input data. HOPE cannot be generalized to a variety of tasks such as node classification (Khosla et al. 2019). Similar to HOPE, APP (Zhou et al. 2017) also captures asymmetric and high-order similarities (Rooted PageRank) between node pairs, but it relies on random walks and Monte-Carlo End-Point sampling. ATP (Sun et al. 2019) is later proposed after finding that cycles in digraphs can undermine the performance of embedding strategies such as HOPE and APP, due to limitation of their high order proximity measures. To address the is-
sue brought by cycles in digraphs, ATP removes cycles in digraphs and leverages factorization to further incorporate graph hierarchy and reachability information.

Undirected GRL methods operate on a single embedding space, but distances in this space are symmetric. In order to capture the asymmetric connectivity of digraphs, HOPE, APP and ATP generate 2 embedding vectors for each node, namely a source vector and a target vector. Directed node-pair asymmetric likelihood can be captured through the inner product between the source node’s source vector and the target node’s target vector. NERD (Khosla et al. 2019) is another method of this type, but it extracts a source walk and a target walk, and employs a shallow neural model to maximize the likelihood of preserving source and target neighborhoods of nodes. Compared to deep alternatives (GNNs), methods with reliance on matrix decomposition or random walks with shallow embeddings, lack scalability, robustness, generalization and inductive capability (Veličković et al. 2017; Chami et al. 2019).

Gravity-inspired graph autoencoders (Salha et al. 2019) have been proposed, aiming to revitalize GNNs that utilize a single embedding space. They propose to learn an additional 1-dimensional mass for each node, which is used together with the node embedding vectors to compute the node-pair likelihood. Our method incorporates this idea. Furthermore, there emerge some GNN methods that extend spectral-based GCN to digraphs (Tong et al. 2020a; Ma et al. 2019b; Zhang et al. 2021a), but are tied to a graph’s Laplacian. DAGNN (Thost and Chen 2021) is proposed, specifically for directed acyclic graphs (DAGs), by injecting the inductive bias of DAG—partial ordering—into the GNN design.

Hyperbolic Embedding Learning.
Recent research has proven that Euclidean space does not provide the most powerful geometrical representations when input data exhibits a highly complex non-Euclidean latent anatomy (Bronstein et al. 2017; Chamberlain, Clough, and Deisenroth 2017). Hyperbolic neural networks (Ganea, Bécigneul, and Hofmann 2018b) and hyperbolic attention networks (Gulcehre et al. 2018) have been derived with the aim to extend the powerful deep learning methods to hyperbolic space. They have been used to address various machine learning problems (Sun et al. 2021a,b; Wang et al. 2020; Choudhary et al., 2021; Tifrea, Bécigneul, and Ganea 2018; Dhingra et al. 2018; Chami et al. 2020; Khrulkov et al. 2020; Liu et al. 2020; Long et al. 2020; Chen et al. 2020; Suris, Liu, and Vondrick 2021; Zeb et al. 2021; Kyriakakis, Fostioropoulos, and Bogdan 2020; Chami et al. 2021). Since in scale-free or hierarchical tree-like graphs, the volume of nodes grows exponentially with distance from a central node, hyperbolic space is suitable for embedding such graphs (Gulcehre et al. 2018; Chami et al. 2019; Blüsius, Friedrich, and Katzmann 2021). On the other hand, Euclidean space only supports a polynomial increase in volume, and thus highly distorts these graphs (Sala et al. 2018). Researchers have started to equip GRL with such geometrically appropriate inductive biases. However, most non-Euclidean embedding techniques, such as Poincaré Embedding (Nickel and Kiela 2017), the Lorentz Model of Hyperbolic Geometry (Nickel and Kiela 2018), Product Space Embedding (Gu et al. 2018), as well as Ultrahyperbolic Representation (Law and Stam 2020), only account for the graph structure and do not leverage rich node features. GRL requires capturing both structural and semantic graph properties to be robust and effective for real-word applications. With this regard, HGCN (Chami et al. 2019) and
To perform link prediction or graph reconstruction on a directed graph, there is a need to estimate non-reciprocal node connectivity probabilities for node pair \((u, v)\) and \((v, u)\). As mentioned earlier, HOPE, APP and ATP use 2 embedding vectors (source/target) for each node, and then use the classic symmetric node proximity measures (e.g., inner product or distance in the embedding space). On the other hand, in the domain of modeling relational data (e.g., knowledge graph embedding), an asymmetry-preserving score function is usually employed. A well-known practice is to design the score function by translating embedding vectors: \(\text{Bordes et al. 2013}\): we list a few typical ones here: 

1. \(\text{score}(u, v) = \|u - v + r\|^2\) where \(u\) and \(v\) are node embedding vectors, and \(r\) is a learned global translation vector (Nickel and Kiela 2017); 
2. \(\text{score}(u, v) = \|u - v + e\|^2\) where the local translation vector \(e\) is the feature of the directed relation/edge (Bordes et al. 2013); 
3. \(\text{score}(u, v) = (1 + \lambda(\|u\| - \|v\|))d(u, v)\) where the hyper-parameter \(\lambda\) is tuned and \(d(u, v)\) denotes the distance between embedding vector \(u\) and \(v\) in the embedding space (Ganea, Bécigneul, and Hofmann 2018a); and 
4. \(\text{score}(u, v) = \gamma(m_v - \lambda \log(\|u - v\|^2))\) where \(\gamma\) denotes the sigmoid function and \(m_v\) is a learned mass for node \(v\) (Salha et al. 2019). The score function can also be parameterized by neural networks that receive an ordered vertex feature pair as input (Battaglia et al. 2018; Klicpera, Groß, and Günnemann 2020). Notably, hyperbolic embedding methods are evaluated on their ability in learning hierarchical structures and modeling relational data since their advent (Nickel and Kiela 2017, 2018; Law and Stari 2020), and the asymmetric property of node connectivity probabilities are often considered by an asymmetry-preserving score function. However, their focus is not general GRL which shall capture both graph structural and semantic information for real-world applications with graphs that potentially contain noises. Another type of approaches constrain the embedding space with custom-designed energy penalty to capture the directed partial order of nodes (e.g., Order Embedding (Vendrov et al. 2015)). Hyperbolic Entailment Cones (Ganea, Bécigneul, and Hofmann 2018a) and Hyperbolic Disk Embedding (Suzuki, Takahama, and Onoda 2019) pass along this idea to hyperbolic space. However, they are specifically designed for DAGs, and cannot generalize to the general digraphs that may contain cycles. In this work, we consider the asymmetric node connectivity probabilities and propose a hyperbolic digraph embedding technique that is suitable for real-world applications with graphs that contain cycles and noises (we set no constraints on the graph structure unlike previous related efforts).

### Details of Experimental Setups

#### Baselines

A taxonomy of baselines is provided below:
• **Digraph embedding methods**: APP ([Zhou et al. 2017]), ATP ([Sun et al. 2019]), NERD ([Khosla et al. 2019]), Gravity GCN and Gravity VGAE ([Salha et al. 2019]), DGCN ([Tong et al. 2020b]), DiGCN ([Tong et al. 2020a]);

• **Spectral-based GNNs**: GCN ([Kipf and Welling 2016a]), VGAE ([Kipf and Welling 2016b]), ChebNet ([Defferrard, Bresson, and Vandergheynst 2016a]), APPNP ([Klicpera, Bojchevski, and G"unnemann 2018]), InfoMax ([Veličković et al. 2018]), SGC ([Wu et al. 2019]), SIGN ([Rossi et al. 2020]);

• **Spatial-based GNNs**: GAT ([Veličković et al. 2017]), and GraphSage ([Hamilton, Ying, and Leskovec 2017]);

• **Hyperbolic embedding methods**: HNN ([Ganea, Bécigneul, and Hofmann 2018b]), and HGCN ([Chami et al. 2019]).

Datasets
We use open access datasets, including:

- Citation Network: Cora,[1] Cora-ML,[2] CiteSee,[3] and DBLP[4];
- Transportation Network: Air[5];
- User Network: Survey[6] and Wiki[7];
- Web Network: Blog[8]

The statistics of the datasets are available in the main paper. Recent surveys ([Dwivedi et al. 2020]) find the split of dataset will greatly affect the method performance. A single split will not only cause overfitting, but also produce misleading results. Therefore, when a dataset is used for a task, we create numerous splits with different random seeds for the dataset.

Tasks
• **Link Prediction (LP).** Given a digraph $\mathcal{G}$, we train models on its incomplete version $\mathcal{G}'$ where 40% of edges are randomly removed. Half of the removed edges form the positive samples in the val set, and the other half forms the positive samples in the test set. The negative samples are randomly sampled from unconnected node pairs in $\mathcal{G}$ with the same number of positive samples. LP is a binary classification task of discriminating the actual removed edges from the fake ones. For each dataset, 10 random splits are created for this task.

• **Semi-supervised Node Classification (NC).** Each dataset contains 20 labeled nodes for each class, forming the training set. The val set consists of 500 random, unlabeled nodes. Unlabeled nodes that are not in the val set make up the test set. 20 random dataset splits are used for this task. We strictly follow the experimental protocol established by the previous work ([Tong et al. 2020a]), and use their reported results on baselines when applicable. We conduct additional experiments by varying the fraction of nodes labeled in training, ranging from 1% to 10%, and then examine the performance of the methods.

• **Link Sign Prediction (SP).** In the Wiki dataset ([Kim et al. 2018]), the edge’s sign is from {oppose, neutral, support}. Given a digraph $\mathcal{G}$, 5% of edges are labeled for training, 5% for validation, and 90% for testing. In addition, 10 random dataset splits are created for this task.

• **Embedding Visualization (EV).** We provide visualizations of the t-SNE ([Van der Maaten and Hinton 2008]) and PCA ([Wold, Esbensen, and Geladi 1987]) transformed node representations in the projected 2D space. For hyperbolic embedding methods including D-HYPR, the final hyperbolic node embedding is mapped to the Euclidean space before using t-SNE and PCA.

Reproducibility

Training Objectives

**Link Prediction (LP).** We follow prior works ([Zhu et al. 2020], [Chami et al. 2019]) to minimize the binary cross-entropy loss with negative sampling. We use the hyperbolic Fermi-Dirac and Gravity regularizers to estimate the likelihood of each node pair (for both positive samples and negative samples). The total loss $\mathcal{L}_{LP}$ for this task is a linear combination of the binary cross-entropy loss computed using the Fermi-Dirac regularizer $\mathcal{L}_f$ (with coefficient 1), and the binary cross-entropy loss computed using the Gravity regularizer $\mathcal{L}_g$: $\mathcal{L}_{LP} = \mathcal{L}_f + w_g \mathcal{L}_g$, where $w_g \in \mathbb{R}$ weights the relative importance of the Gravity regularizer to the Fermi-Dirac regularizer.

**Node Classification (NC).** We use the multiclass logistic regression model to perform node classification with inputs of node embeddings from the models. The graph embedding model and the logistic regression model are trained end to end. For D-HYPR, we map the hyperbolic node embeddings to the tangent space of the origin with the logarithmic map (i.e., $z_{i,E}^j = \log(z_{i,H})$), then perform Euclidean logistic regression. The total loss $\mathcal{L}_{NC}$ is a linear combination of the negative log likelihood loss $\mathcal{L}_{class}$, the Fermi-Dirac loss $\mathcal{L}_f$ and the Gravity loss $\mathcal{L}_g$: $\mathcal{L}_{NC} = \mathcal{L}_{class} + w_f (\mathcal{L}_f + w_g \mathcal{L}_g)$, where $w_f, w_g \in \mathbb{R}$ weights the relative importance of regularization.

**Link Sign Prediction (SP).** Similar to the node classification task, we use the multiclass logistic regression model to perform link sign classification, but, with concatenated node embeddings from nodes of every node pair as input. This applies to all methods in our SP experiments. For D-HYPR, after mapping the hyperbolic node embeddings to the tangent space using the logarithmic map, in order to obtain the class likelihood of edge $(i, j)$, the concatenated vector of embeddings from node $i$ and node $j$ serves as the input to the Euclidean logistic regression model. The total loss is computed as $\mathcal{L}_{SP} = \mathcal{L}_{class} + w_f (\mathcal{L}_f + w_g \mathcal{L}_g)$. 

---

[1] https://linqs.soe.ucsc.edu/data
[2] https://github.com/flyingtango/DiGCN
[3] https://github.com/flyingtango/DiGCN
[4] http://konect.uni-koblenz.de/networks/dblp-cite
[5] http://konect.uni-koblenz.de/networks/maayan-faa
[6] http://konect.uni-koblenz.de/networks/moreno_health
[7] https://snap.stanford.edu/data/wiki-Vote.html
[8] http://konect.uni-koblenz.de/networks/moreno_blogs
Implementation Details

We implement D-HYPR and the majority of the baseline models, i.e., MLP, GCN, GAT, HNN, HGCN using the PyTorch Python library. Our implementations of HAT, Gravity VGAE, Gravity GCN, VGAE, and GCN for the link prediction experiment are based on the provided code from the authors with the TensorFlow Python library. We use the public available Python implementation of ATP and C++ implementation of NERD provided from the authors. The machine used for running most of the experiments has 256 GB memory, 48 CPU cores at 2.30GHz, an NVIDIA Quadro K6000 graphics card, and in Linux operating system.

We perform hyper-parameter search for each model with experimentation on the first fold (i.e., split) of each dataset and task based on the validation performance, over initial learning rates \{0.001, 0.01, 0.1\}, momentums \{0.9, 0.999\}, weight decays \{0.0, 0.001\}, and dropout rates \{0, 0.05, 0.1\}. For attention-based models (e.g., GAT), the number of attention heads is searched from \{4, 8\}, and alpha from \{0.1, 0.2\}. We tune \lambda\) from \{0.01, 0.05, 1.5\} for the Gravity regularizer (except the node classification experiments on Cora-ML and CiteSeer where we perform more fine-grained parameter sensitivity study as demonstrated in the main paper). We use \(r=2\) and \(t=1\), the default values, for the Fermi-Dirac regularizer. \(w_f\) is searched from \{0.1, 0.5, 1, 5\} and \(w_r = 0.5\). In addition, we set \(K = 2\) to obtain 8 sets of neighborhoods per node for our model D-HYPR (except the dblp dataset with \(K = 1\)). Moreover, averaging that obtains \(x_{\text{fuse}}\) is achieved through Möbius scalar multiplication in Equa. 5 and Möbius addition in Equa. 4 in the main paper. For a rigorous and fair comparison, we control the number of dimensions of the node embeddings produced by each method to be the same for all methods (32-dimensional embeddings), and use 2 layers with the first layer producing 64-dimensional outputs. We optimize the models with Adam (Kingma and Ba 2014), and use early stopping based on the validation performance. Random seed selection is automatic and the selected seeds are recorded. In addition, results of ChebNet, GCN, SGC, APPNP, InfoMax, GraphSage, SIGN, DGCN, DiGCN on Semi-supervised Node Classification are obtained from (Tong et al. 2020a) since we strictly follow their established experimental protocols and their code.

Additional Experimental Results

Link Prediction. In Table 7, we provide the standard deviation of the link prediction experiment with 32-dimensional node embeddings. In addition, we provide results on a large-scale dataset, DBLP. The DBLP digraph is a citation network with 12,591 nodes, and 49,743 edges. As demonstrated in the table, overall, methods with higher mean metric values have more stable performance (smaller standard deviations). D-HYPR produces stable results across different dataset splits and random seeds.

The strong performance of D-HYPR attributes to its use of hyperbolic space, information collected from the multi-ordered diverse neighborhoods and accounting for directionality. D-HYPR models the distinctive node neighborhoods, captures the local directed structural characteristics, and encodes critical semantic information from a given digraph with a hyperbolic GNN-based design.

Embedding Visualization. In Figure 4, we visualize 2D projections of embeddings produced by MLP, GCN, GAT, HNN, HGCN and D-HYPR, using both PCA and t-SNE. Each dot represents a node, and color denotes the ground truth class label. As shown in the figure, the visualizations from D-HYPR exhibit discernible clustering with clusters correspond to the class labels of the dataset. D-HYPR leads to a better class separation, which indicates a embedding space that better captures the semantics of the dataset.

Additional Experimental Results

Link Prediction. In Table 7, we provide the standard deviation of the link prediction experiment with 32-dimensional node embeddings. In addition, we provide results on a large-scale dataset, DBLP. The DBLP digraph is a citation network with 12,591 nodes, and 49,743 edges. As demonstrated in the table, overall, methods with higher mean metric values have more stable performance (smaller standard deviations). D-HYPR produces stable results across different dataset splits and random seeds.

The strong performance of D-HYPR attributes to its use of hyperbolic space, information collected from the multi-ordered diverse neighborhoods and accounting for directionality. D-HYPR models the distinctive node neighborhoods, captures the local directed structural characteristics, and encodes critical semantic information from a given digraph with a hyperbolic GNN-based design.

Embedding Visualization. In Figure 4, we visualize 2D projections of embeddings produced by MLP, GCN, GAT, HNN, HGCN and D-HYPR, using both PCA and t-SNE. Each dot represents a node, and color denotes the ground truth class label. As shown in the figure, the visualizations from D-HYPR exhibit discernible clustering with clusters correspond to the class labels of the dataset. D-HYPR leads to a better class separation, which indicates a embedding space that better captures the semantics of the dataset.

Additional Experimental Results

Link Prediction. In Table 7, we provide the standard deviation of the link prediction experiment with 32-dimensional node embeddings. In addition, we provide results on a large-scale dataset, DBLP. The DBLP digraph is a citation network with 12,591 nodes, and 49,743 edges. As demonstrated in the table, overall, methods with higher mean metric values have more stable performance (smaller standard deviations). D-HYPR produces stable results across different dataset splits and random seeds.

The strong performance of D-HYPR attributes to its use of hyperbolic space, information collected from the multi-ordered diverse neighborhoods and accounting for directionality. D-HYPR models the distinctive node neighborhoods, captures the local directed structural characteristics, and encodes critical semantic information from a given digraph with a hyperbolic GNN-based design.

Embedding Visualization. In Figure 4, we visualize 2D projections of embeddings produced by MLP, GCN, GAT, HNN, HGCN and D-HYPR, using both PCA and t-SNE. Each dot represents a node, and color denotes the ground truth class label. As shown in the figure, the visualizations from D-HYPR exhibit discernible clustering with clusters correspond to the class labels of the dataset. D-HYPR leads to a better class separation, which indicates a embedding space that better captures the semantics of the dataset.

Additional Experimental Results

Link Prediction. In Table 7, we provide the standard deviation of the link prediction experiment with 32-dimensional node embeddings. In addition, we provide results on a large-scale dataset, DBLP. The DBLP digraph is a citation network with 12,591 nodes, and 49,743 edges. As demonstrated in the table, overall, methods with higher mean metric values have more stable performance (smaller standard deviations). D-HYPR produces stable results across different dataset splits and random seeds.

The strong performance of D-HYPR attributes to its use of hyperbolic space, information collected from the multi-ordered diverse neighborhoods and accounting for directionality. D-HYPR models the distinctive node neighborhoods, captures the local directed structural characteristics, and encodes critical semantic information from a given digraph with a hyperbolic GNN-based design.

Embedding Visualization. In Figure 4, we visualize 2D projections of embeddings produced by MLP, GCN, GAT, HNN, HGCN and D-HYPR, using both PCA and t-SNE. Each dot represents a node, and color denotes the ground truth class label. As shown in the figure, the visualizations from D-HYPR exhibit discernible clustering with clusters correspond to the class labels of the dataset. D-HYPR leads to a better class separation, which indicates a embedding space that better captures the semantics of the dataset.

Additional Experimental Results

Link Prediction. In Table 7, we provide the standard deviation of the link prediction experiment with 32-dimensional node embeddings. In addition, we provide results on a large-scale dataset, DBLP. The DBLP digraph is a citation network with 12,591 nodes, and 49,743 edges. As demonstrated in the table, overall, methods with higher mean metric values have more stable performance (smaller standard deviations). D-HYPR produces stable results across different dataset splits and random seeds.

The strong performance of D-HYPR attributes to its use of hyperbolic space, information collected from the multi-ordered diverse neighborhoods and accounting for directionality. D-HYPR models the distinctive node neighborhoods, captures the local directed structural characteristics, and encodes critical semantic information from a given digraph with a hyperbolic GNN-based design.

Embedding Visualization. In Figure 4, we visualize 2D projections of embeddings produced by MLP, GCN, GAT, HNN, HGCN and D-HYPR, using both PCA and t-SNE. Each dot represents a node, and color denotes the ground truth class label. As shown in the figure, the visualizations from D-HYPR exhibit discernible clustering with clusters correspond to the class labels of the dataset. D-HYPR leads to a better class separation, which indicates a embedding space that better captures the semantics of the dataset.