# Dual-Geometric Space Embedding Model for Two-View Knowledge Graphs

Roshni G. Iyer
University of California, Los Angeles
Los Angeles, CA, USA
roshnigiyer@cs.ucla.edu

Yunsheng Bai
University of California, Los Angeles
Los Angeles, CA, USA
yba@ucla.edu

Wei Wang
University of California, Los Angeles
Los Angeles, CA, USA
weiwang@cs.ucla.edu

Yizhou Sun
University of California, Los Angeles
Los Angeles, CA, USA
yzsun@cs.ucla.edu

## ABSTRACT

Two-view knowledge graphs (KGs) jointly represent two components: an ontology view for abstract and commonsense concepts, and an instance view for specific entities that are instantiated from ontological concepts. As such, these KGs contain heterogeneous structures that are hierarchical, from the ontology-view, and cyclical, from the instance-view. Despite these various structures in KGs, most recent works on embedding KGs assume that the entire KG belongs to only one of the two views but not both simultaneously. For works that seek to put both views of the KG together, the instance and ontology views are assumed to belong to the same geometric space, such as all nodes embedded in the same Euclidean space or non-Euclidean product space, an assumption no longer reasonable for two-view KGs where different portions of the graph exhibit different structures. To address this issue, we define and construct a dual-geometric space embedding model (**DGS**) that models two-view KGs using a complex non-Euclidean geometric space, by embedding different portions of the KG in different geometric spaces. **DGS** utilizes the spherical space, hyperbolic space, and their intersecting space in a unified framework for learning embeddings. Furthermore, for the spherical space, we propose novel closed spherical space operators that directly operate in the spherical space without the need for mapping to an approximate tangent space. Experiments on public datasets show that **DGS** significantly outperforms previous state-of-the-art baseline models on KG completion tasks, demonstrating its ability to better model heterogeneous structures in KGs.

## CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; non-Euclidean geometric space embedding.

## KEYWORDS

Knowledge Graph Embeddings; Non-Euclidean Geometry



*KDD '22, August 14–18, 2022, Washington, DC, USA*

ACM ISBN 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539350

**ACM Reference Format:**
Roshni G. Iyer, Yunsheng Bai, Wei Wang, and Yizhou Sun. 2022. Dual-Geometric Space Embedding Model for Two-View Knowledge Graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539350

## 1 INTRODUCTION

Knowledge graphs (KGs) are essential data structures that have been shown to improve several semantic applications, including semantic search [4], question answering [13], and recommender systems [26]. Two-view knowledge graphs [14] consist of both (1) an instance-view component, containing entity-entity relations, and (2) an ontology-view component, containing concept-concept and entity-concept relations. In addition, there are nodes involved in both entity-concept relations and entity-entity relations, which we refer to as *bridge nodes*, e.g., nodes 5 and 6 in Figure 1(a). Bridge nodes provide the connections to bridge both KG views. The instance-view component contains data that form cyclical structures, such as nodes 8-10 in Figure 1(a). The ontology-view component contains data that form hierarchical structures. As such, the two-view KG contains intrinsic heterogeneous structures.

A key challenge with KGs is that they are often highly incomplete, making KG completion an important area of investigation. In recent years, several research works have focused on improving knowledge graph embeddings (KGE) to address this task [6, 28]. The idea behind embedding methods is to map entities and relations to a latent low-dimensional vector space while preserving the semantics and inherent structures in the KG. However, most embedding methods, including graph neural network (GNN)-based [8, 9, 22] and non-GNN-based methods [15, 25], are limited to operating on KGs belonging to either the instance-view or the ontology-view but not both. Specifically, they either model entities and relations in the zero-curvature Euclidean space [23], omitting both cyclic and hierarchical structures, or in the hyperbolic space [1, 30], omitting cyclic structures and treating relations as hierarchical.

For works that seek to put both views of the KG together, the instance and ontology views are assumed to belong to the same space, an assumption no longer reasonable for two-view KGs. Specifically, all KG relations are modeled using the Euclidean space [12], or are all modeled using a product space, being the Riemannian product manifold of the Euclidean, spherical and hyperbolic spaces [27]. However, our goal in modeling two-view KGs is to use a unified

Figure 1: Two-View KG visualization and embedding space. (a) Example two-view KG. (b) Corresp DGS representation.

framework to embed hierarchical structures of the KG in the hyperbolic space, cyclic structures of the KG in the spherical space, and nodes involved in both structures in a uni ed intersection space.

To address the above challenges, in this paper, we present the Dual-Geometric Space embedding model DGS for two-view KGs. To summarize, our work makes the following contributions:

- We formulate the problem of modeling a two-view KG in complex non-Euclidean geometric space for KG completion. To our knowledge, we are the rst to model two-view KGs in distinct non-Euclidean spaces using a unifying framework, e.g., di erent views belong to di erent geometric spaces.
- We propose to model instance-view entities in the spherical space for solely cyclical relations, and ontology-view concepts in the hyperbolic space for solely hierarchical relations, and bridge entities involved in both cyclic/hierarchical structures in a specially designed intersection bridge space.
- To the best of our knowledge we are also the rst to design closed spherical space operations, to directly operate in the spherical space without mapping to an external approximation space, e.g., the tangent Euclidean space.
- We investigate seven variant and ablation models of DGS and evaluate these models on two KG completion tasks. Extensive experiments demonstrate the e ectiveness of DGS in populating knowledge in two-view KGs. Further, our model signi cantly outperforms its single non-Euclidean and Euclidean geometric space counterparts including the product space, and existing state-of-the-art graph neural network (GNN) embedding methods.

## 2 PRELIMINARY AND RELATED WORK

Here, we formalize two-view KGs, which jointly model the entities and concepts in the instance and ontology views, and discuss the achievement of primary state-of-the-art models for two-view KGs.

### 2.1 Problem Formulation

A two-view KG, , consists of nodes and edges, such that nodes denote the set of entities, or set of concepts, . Edges denote relations where' are the set of entity-entity relations', $ are the set of concept-concept relations, and' $ are the set of entity-concept relations. The two-view KG consists of an instance-view component containing entities with relations of ' , and an ontology-view component containing concepts and bridge entities with relations of ' $ and ' $ . We denote a subset of entities in to be bridge entities, that communicate between both instance and ontology views. These bridge entities associate with both and ' $ relations. Relations in ' $ are entity-concept relations, which are similar to concept-concept relations such as is_a. and as well as ' , ' O and ' O are each disjoint sets. We denote 4 2 to be the 8-th entity, h48 to be entity 8s representation in the Cartesian coordinate system, and A: 2 ' to be the : -th relation in the instance-view component. Without loss of generality (WLOG), we also model the ontology-view component where 2 2 is the 8-th concept, h28 is concept 8s representation in the Cartesian coordinate system, and AO: 2 ' O is the : -th relation in the ontology-view component. Therefore, the instance-view can be considered as a set of triples in the form ¹4 8 • A: • 4 9 º 2 ' , which may be any real-world instance associations and are often cyclical in structure. Likewise, the ontology-view can be represented by a set of triples ¹2 8 • AO: • 2 9 º 2 ' O , which contain hierarchical associations such as is_a. Here we use Figure 1(a) to illustrate the aforementioned problem formulation. For example, (Soil, composed_of, Mineral) and (Fungus, is_a, Eukaryote) are instance-view and ontology-view triples respectively. Further, nodes Bacteria-Soil-Mineral is an example of a cycle between non-bridge entities.

A bridge entity may be involved with two types of triples: ¹4 8 • AO: • 2 9 º 2 ' O for (hierarchical) entity-concept relations and ¹4 8 • A: • 4 9 º 2 ' for (cyclical) entity-entity relations. For example, Rabbit is a bridge entity in Figure 1(a). The triple (Rabbit, emit, Carbon Dioxide) represents an entity-entity relation, and the triple (Rabbit, is_a, Herbivore) represents an entity-concept relation.

The objective of our research is to learn KG embeddings of nodes and relations in the KG, such that we seamlessly unify multiple curvature geometric spaces to better capture the contextual information and heterogeneous structures in the KGs. We evaluate the quality of the learned embeddings on the KG tasks of triple completion and entity typing, described in Section 4.

### 2.2 Non-Euclidean Geometric Spaces

In this section, we describe the various properties of non-Euclidean geometric spaces, which are curved spaces unlike the zero-curvature

Euclidean space. The textbook [9] provides more details. Geometric spaces of Euclidean ($\mathbb{E}^3$), spherical ($\mathbb{S}^3$), and hyperbolic ($\mathbb{H}^3$) spaces belong to Riemannian manifolds ($\mathcal{M}^3$), such that each point $a \in \mathcal{M}^3$ has a corresponding tangent space $\mathcal{T}_a\mathcal{M}^3$, that approximates $\mathcal{M}^3$ around $a$. Further, each Riemmanian manifold $\mathcal{M}^3$, is associated with a Riemanian metric $g(\cdot,\cdot)$ that defines the geodesic distance of two points on the manifold and the curvature of the space. In the spherical space, curvature $c > 0$, suitable for capturing cyclical structures [29], while in the hyperbolic space, curvature $c < 0$, suitable for capturing hierarchical structures [18]. Widely used models on the hyperbolic space include the Poincaré ball model [7], the Lorentz [3] model, and the Klein model [3]. As the these three models are both isometric and isomorphic to one another, WLOG we utilize the Poincaré ball model in our work.

Non-Euclidean Space Optimization . DGS utilizes Riemannian optimization for updating entity and relational embeddings because Euclidean space optimization methods, such as SGD, provide the update direction of the Euclidean gradient to be in non-curvature space. This does not align with parameters in our model that must be updated in positive or negative curvature spaces. For parameter learning, DGS uses RSGD [5], whose update function is denoted below, where $\nabla_a \in \mathcal{T}_a\mathcal{M}^3$ denotes the tangent Euclidean space of $a \in \mathcal{M}^3$, $r_t \in \mathcal{T}_a\mathcal{M}^3$ denotes the Riemannian gradient of loss function $\mathcal{L}$, $R_{a_t}$ denotes retraction onto $\mathcal{M}^3$, or non-Euclidean space at $a_t$, and $\eta_t$ denotes learning rate at time $t$.

$$a_{t+1} = R_{a_t}(a_t - \eta_t r_t) \quad (0)$$

The retraction operator $R_{a}(v)$ involves mapping between spaces. For non-Euclidean spaces, the retraction is generally performed between the non-Euclidean space and approximate tangent Euclidean space using logarithmic and exponential mapping functions as follows, where $\log_0(h_{a_2})$ is a logarithmic map at center 0 from the hyperbolic space to Euclidean tangent space, and $\exp_0(h_{a_2})$ is an exponential map at center 0 from the Euclidean tangent space to hyperbolic space of $a$.

$$\log_0(a) = \tanh^{-1}(c\,\|a\|)\frac{a}{c\,\|a\|} \quad (1)$$

$$\exp_0(a) = \tanh(c\,\|a\|)\frac{a}{c\,\|a\|} \quad (2)$$

## 2.3 Two-View KG Models

In this section, we describe the models that are utilized for two-view KGs, which consider the problem setting of modeling ontological and instance views. To address the challenges of these models, we propose DGS in Section 3.

JOIE. The JOIE [12] model is an embedding model that considers the problem of two-view KGs. However, JOIE models all triples in the same zero-curvature Euclidean space, omitting the hierarchical and cyclical structures of the KG. Further, there is no special consideration of representation for bridge nodes.

Models leveraging product of spaces . Models including M2GNN [27] which extends [11] from social networks to the domain of KGs, [21], etc. help to address the limitation of JOIE by utilizing the hyperbolic and spherical spaces for representing triples. However, they treat all triples in the KG to be in the same non-Euclidean product space, formed by a Riemannian product of the Euclidean, spherical, and hyperbolic spaces. Thus, the embedding space of hierarchical triples is not distinguished from the embedding space of cyclic triples, and further there is also no special consideration of representation for bridge nodes.

# 3 DGS ARCHITECTURE

This section describes our proposed dual-geometric space embedding model (DGS), which jointly embeds entities and concepts in a knowledge graph. DGS embeds different link types of the two-view KG in different non-Euclidean manifold spaces to capture the inherent heterogeneous structures of the KG, as described below. We also design general procedures, which are detailed in the Supplements, for efficiently converting arbitrary 3-dimensional embeddings between the polar and Cartesian coordinate representations, which are utilized by sub-models of DGS. Source code is available at https://github.com/roshnigiyer/dgs .

## 3.1 Modeling

The key questions in the modeling process are in how to: (1) design or adopt an appropriate embedding space for nodes, (2) define the representation of entity/concept and relation in that embedding space, and (3) define the KG embedding model and loss function of DGS. Our framework enables each of entities and concepts to be influenced by bridge nodes, and simultaneously bridge nodes to be informed by entities and concepts. In this way, DGS exhibits a unified framework to jointly model two-views of KGs.

DGS models nodes in the instance view as points in a spherical space $\mathbb{S}^3$ with fixed norm space $F(\cdot)$, and nodes in the ontology view as points in a hyperbolic space $\mathbb{H}^3$ with learnable norm space $F_{h_{2_8}}$ per concept. The bridge nodes lie in the intersection of the two, which is a submanifold intersection space, called $\mathbb{B}^3$, shown as the dotted circle in Figure 1(b). $\mathbb{B}^3$ contains the same fixed norm space as the spherical space $F(\cdot)$. For modeling compatibility, we set the degrees of freedom of $\mathbb{S}^3$, $\mathbb{H}^3$, and $\mathbb{B}^3$ to $3-1, 3-1$, and $3-2$ respectively. $\mathbb{B}^3$ has one degree of freedom less than $\mathbb{H}^3$ and $\mathbb{S}^3$ because it is a submanifold intersection space of both the spherical and hyperbolic spaces. The norm of $\mathbb{B}^3$ is $F(\cdot)$ because that is the intersection norm space of $\mathbb{S}^3$ and $\mathbb{H}^3$. The concept-specific norm spaces $F_{h_{2_8}}$, are learnable in order for the hierarchy in the KG to be learned by the embedding. In practice, it can be seen that hierarchical root concepts move towards the center of the hyperbolic space e.g., towards norm 0, shown in Section 4.

Parameter optimization, detailed in Section 3.2, is performed using RSGD, described in Section 2.2, on hinge loss functions which utilize non-Euclidean space geodesic distances for the spherical and hyperbolic spaces respectively. We construct the hinge loss function such that positive triples are scored higher than negative triples within positive margin hyperparameters.

### 3.1.1 Modeling Instance View KG in Spherical Space.

Representation of entities. We propose to embed the entities from the instance-view on the surface of the spherical ball from the spherical space $\mathbb{S}^3$, in order to better capture the cyclic structures

present in this KG view. Entities are represented as points $h_{48}$ that belong to the surface of the spherical ball $S^3$ as shown in Figure 1(b). Formally, $S^3 = \{h_{48} \in R^3 \mid \|h_{48}\| = F(g) = F(2) > 0\}$, is the 3-dimensional $F(\cdot)$-norm ball, where $\|\cdot\|$ is the Euclidean norm.

For an entity $h_{48}$, we propose to model the relation applied to the entity as a rotation operation, and therefore represent the relation $A_{:}$ as a vector of angles $\theta_{A:} \in [0, 2\pi)^3$. Below, we show our proposed vector transformation procedure to model the relation as rotation of the head entity, $T_{rot}(h_{48}^{(\cdot=4F)}) = T_{rot}(h_{48}) \odot \theta_{A:}$, and prove that the operation is closed in the spherical space. $T_{rot}$ eliminates the need to project representations to the tangent Euclidean space in order to perform standard transformations. To the best of our knowledge, we are the first to propose a closed vector transformation procedure that operates directly in the spherical space.

Spherical Space Vector Transformation Procedure: $T_{rot}$. This section describes our proposed vector transformation procedure, $T_{rot}$, for the spherical space, which directly performs operations using properties, such as positive curvature, in the spherical space. This section also outlines the proof of the closedness property of the transformation procedure. For computational efficiency, we also extend the vector transformation procedure to the batch version, the batch vector transformation procedure, which is detailed in the Supplements. $T_{rot}$ takes as input an entity embedding and relation operator, which it uses to transform the entity embedding: $h_{48}^{(\cdot=4F)} = T_{rot}(h_{48}) \odot \theta_{A:}$:

(1) Given $h_{48}$, we convert $h_{48}$ to the corresponding representation $\phi_{48}$ in polar coordinates, with rad denoting the radius which has the value $F(\cdot)$. Refer to Section A.1 for details on the conversion procedure.

$$h_{48} : [G_{48\cdot1}, G_{48\cdot2}, \ldots, G_{48\cdot3}] \rightarrow \phi_{48} : [\phi_{48\cdot1}, \phi_{48\cdot2}, \ldots, \phi_{48\cdot3}, \text{rad} = F(\cdot)] \quad (3)$$

$$\theta_{A:} : [\theta_{A:\cdot1}, \theta_{A:\cdot2}, \ldots, \theta_{A:\cdot3}, \text{rad} = F(\cdot)] \quad (4)$$

(2) Denote $z_{\cdot i}$ to be the $i$-th entry of $z$ and apply the transformation:

$$\phi_{48,\cdot i} = \phi_{48,\cdot i} + \theta_{A:,\cdot i} \bmod 2\pi; \quad i = 1, 3 \quad (5)$$

$$\phi_{48}^{(\cdot=4F)} = [\phi_{48\cdot1}, \theta_{A:\cdot1} \bmod 2\pi, \ldots, \phi_{48\cdot3}, \theta_{A:\cdot3} \bmod 2\pi, \text{rad} = F(\cdot)] \quad (6)$$

(3) Convert from polar coordinates back to Cartesian coordinates. Refer to Section A.2 for details on the conversion procedure.

$$\phi_{48}^{(\cdot=4F)} \rightarrow h_{48}^{(\cdot=4F)} : [G_{48\cdot1}^{(\cdot=4F)}, G_{48\cdot2}^{(\cdot=4F)}, \ldots, G_{48\cdot3}^{(\cdot=4F)}] \quad (7)$$

Theorem. The vector transformation procedure $T_{rot}$, is closed in the spherical space.

Proof. The proof is outlined by examining all three steps for $T_{rot}$ from the transformation procedure, where $h_{48}$ and $\phi_{48}$ represent the same point $x_{48}$ in the embedding space by the isomorphic Cartesian and polar coordinate systems respectively.

(1) The Cartesian coordinate representation is equivalent to the polar coordinate representation of the point $x_{48}$, under MCP model, detailed in the Supplements. Further, radius = $F(\cdot)$ of the polar representation embeddings lies in the spherical norm space $F(\cdot)$.
(2) $\theta_{A:} \in [0, 2\pi)^3$; rad = $F(\cdot)$ defines polar representation embeddings in the spherical space, and $\phi_{48,\cdot i} = \phi_{48,\cdot i} + \theta_{A:,\cdot i} \bmod 2\pi \in [0, 2\pi)^3$; rad = $F(\cdot)$. Thus, both angular and radial coordinates are preserved.
(3) The polar coordinate representation is equivalent to the Cartesian coordinate representation of the point $x_{48}$, under MPC model, detailed in the Supplements.

Instance View Loss Function. The instance-view model uses a hinge loss function that is maximized for all triples in the instance space $\mathcal{T}$, with positive triples denoted $t^+ = (h_{48}, \theta_{A:}, h_{49}) \in \mathcal{T}$ with corresponding score $q_{obs}(t^+) \in \mathbb{R}$ and negative triples denoted $t^- = (h_{48}^0, \theta_{A:}, h_{49}^0) \notin \mathcal{T}$ with corresponding score $q_{corr}(t^-) \in \mathbb{R}$, and $\gamma > 0$ is a positive margin hyperparameter. Specifically, the instance loss function measures the distance between the predicted tail entity and the ground truth.

$$q_{obs}(t^+) = \text{dist}(T_{rot}(h_{48}) \odot \theta_{A:}, h_{49}) \quad (8)$$

$$q_{corr}(t^-) = \text{dist}(T_{rot}(h_{48}^0) \odot \theta_{A:}, h_{49}^0) \quad (9)$$

$$\mathcal{L}_{inst} = \frac{1}{|\mathcal{T}|} \sum_{t^+ \in \mathcal{T}, t^- \notin \mathcal{T}} \max(0, \gamma + q_{obs}(t^+) - q_{corr}(t^-)) \quad (10)$$

We calculate spherical geodesic distance [17] between points $x$ and $y$ on the manifold as follows:

$$\text{dist}(x, y) = \arccos(\langle x, y \rangle) \quad (11)$$

$\mathcal{T}$ includes links between all entities including both non-bridge entities and bridge-entities. In this way, the representation of non-bridge entities is influenced by the bridge entities.

### 3.1.2 Modeling Ontological View KG in Hyperbolic Space.

Representation of concepts. We propose to embed concepts from the ontology-view on the Poincaré disk from the hyperbolic space, $H^3$, in order to better capture the hierarchical structures present in this KG view. Concepts are represented as points $h_{28}$, that belong inside the Poincaré disk $H^3$ as shown in Figure 1(b). Formally, $H^3 = \{h_{28} \in R^3 \mid \|h_{28}\| = F_{28}(g) < F_{28}(2) > 0\}$, is the 3-dimensional $F_{28}$-norm ball, where $\|\cdot\|$ is the Euclidean norm. We assume the center of the disk is aligned with the center of the sphere, and for convenience set the last dimension to 0.

For concepts in the hyperbolic space $H^3$, any hyperbolic space model for KG can be applied in principle, which we denote as follows with $A_{O:} \in \mathcal{T}_O$ to denote a relation between two concepts:

$$T_{KGE}(t_\$) = T_{KGE}(h_{28}, A_{O:}, h_{29}) \quad (12)$$

We illustrate MuRP [2] as an example scoring function, which uses the hyperbolic geodesic distance and relies on Möbius addition to model the relation, where $\exp_0(\cdot)$ and $\log_0(\cdot)$ are defined in Section 2.2, $X \in R^{3\times3}$ is a learnable diagonal relation matrix

representing the stretch transformation by relation $r \in \mathcal{R}_O$ with representation $\mathbf{h}_{A_{O:}}$, and scalar biases $b_{h_{2_8}}, b_{h_{2_9}}$ of concepts $2_8$ and $2_9$:

$$\begin{aligned} d_{\mathrm{MuRP}}(pt_\$) &= d_{\mathrm{MuRP}}(\mathbf{h}_{2_8} \cdot A_{O:} \cdot \mathbf{h}_{2_9}) \\ &= d_{\mathrm{MuRP}}(\mathbf{h}_{2_8} \cdot \mathbf{h}_{A_{O:}} \cdot \mathbf{h}_{2_9}) \\ &= \mathrm{dist}\left(\exp_0(\log_0(\mathbf{h}_{2_8}) \cdot \mathbf{h}_{2_9}) \oplus \mathbf{h}_{A_{O:}}, b_{2_8}, b_{2_9}\right) \end{aligned} \tag{13}$$

$$x \oplus \tilde{} = \frac{(1 + 2\langle x, \tilde{} \rangle + \|\tilde{}\|^2)x + (1 - \|x\|^2)\tilde{}}{1 + 2\langle x, \tilde{} \rangle + \|x\|^2\|\tilde{}\|^2} \tag{14}$$

We calculate hyperbolic geodesic distance $\mathrm{dist}(\cdot,\cdot)$ between points $x$ and $\tilde{}$ on the manifold as follows:

$$\mathrm{dist}(x, \tilde{}) = \operatorname{arccosh}\left(1 + 2\frac{\|x \oplus \tilde{}\|^2}{(1 - \|x\|^2)(1 - \|\tilde{}\|^2)}\right) \tag{15}$$

Ontological View Loss Function. The ontology-view model uses a hinge loss function that is maximized for all links between concepts in the ontology space and all links between bridge nodes and concepts, i.e., $\mathcal{G}_O$. With positive triples denoted $pt_O = (\mathbf{h}_{2_8}, r) \cdot A_{O:} \cdot \mathbf{h}_{2_9}$ with $(\mathbf{h}_{4_8}, r) \cdot \mathbf{h}_{A_{O:}} \cdot \mathbf{h}_{2_9}$ with corresponding score $q_{>1B}(pt_O)$ and negative triples denoted $nt_O = (\mathbf{h}_{2_8}', r) \cdot A_{O:} \cdot \mathbf{h}_{2_9}'$ with $(\mathbf{h}_{4_8}', r) \cdot \mathbf{h}_{A_{O:}} \cdot \mathbf{h}_{2_9}'$ with corresponding score $q_{2>A\tilde{A}}(nt_O)$, and $\dot{W} \in \mathcal{G}_O$ is a positive margin hyperparameter:

$$q_{>1B}(pt_O) = d_{\mathrm{KGE}}(pt_O) \tag{16}$$

$$q_{2>A\tilde{A}}(nt_O) = d_{\mathrm{KGE}}(nt_O) \tag{17}$$

$$\mathcal{L}_{\mathrm{onto}}(\mathcal{G}_O) = \frac{1}{|\mathcal{G}_O|} \sum_{pt_O \in \mathcal{G}_O,\, nt_O \in \mathcal{G}_O} \max(0, \dot{W} - q_{>1B}(pt_O) + q_{2>A\tilde{A}}(nt_O)) \tag{18}$$

$\mathcal{G}_\$ \cup \mathcal{G}_O$ includes triples composed of concepts and bridge-entities. In this way, the representation of concepts is influenced by the bridge entities, which is described in Section 3.1.3.

### 3.1.3 Modeling the Intersection of the Two Spaces.

Representation of bridge entities. Bridge nodes are entity nodes that bridge the communication between the instance-view and ontology-view components. Bridge nodes are connected to concepts in the graph, formed by link ontology $\mathcal{G}_{A_{O:}}$ but may also be connected to other entities, formed by link instance $\mathcal{G}_{A:}$. As shown in Figure 1(a), nodes 5 and 6 are bridge nodes that are involved in both cyclic and hierarchical structures through their links to other entities as well as concepts. As such, we propose to embed bridge entities in the intersection space of the Poincare disk and surface of the spherical ball, in order to better capture the heterogeneous KG structures that are associated with these nodes. We refer to the intersection submanifold embedding space as the bridge space $\mathcal{B}^3$, where the representation of these nodes are informed by both $\mathcal{S}^3$ and $\mathcal{H}^3$ and has one lower degree of freedom. The bridge space can therefore be derived as a sphere in general. Formally, $\mathcal{B}^3 = \{\mathbf{h}_{4_8} \in \mathbb{R}^3 \mid \|\mathbf{h}_{4_8}\| = \mathcal{F}(g) = \mathcal{F}(\mathcal{L}_{\mathrm{onto}}) > 0\}$, is the 3-dimensional $\mathcal{F}(\cdot)$-norm ball, where $\|\cdot\|$ is the Euclidean norm, and the value of last dimension $\mathcal{B}^3 = 0$. Links associated with bridge nodes are $\mathcal{G}_{A:} \cdot \mathbf{h}_{A_{O:}} \in \{0, 2c\}^3 \setminus 1$, and operations on bridge nodes, such as geodesic distance and loss functions, happen in either the spherical space or hyperbolic space.

To ensure compatibility with the connected concept nodes, we map the bridge entities $\mathbf{h}_{4_8}$, to an embedding in the ontology space through a non-linear transformation function $\mathcal{G}_{\mathbf{h}_{A_{O:}}}(\mathbf{h}_{4_8})$, where $\mathrm{AGG}(\cdot)$ denotes an averaging over all relations in $\mathcal{G}_\$$. Logarithmic and exponential mapping functions $\log_0(\cdot)$ and $\exp_0(\cdot)$ are described in Section 2.2.

$$\mathcal{G}_{\mathbf{h}_{A_{O:}}}(\mathbf{h}_{4_8}) = \mathrm{AGG}\left(\mathrm{proj}\left(\tanh(\mathbf{h}_{A_{\$:}})\right)\left(\mathbf{h}_{4_8} \oplus b_{\mathbf{h}_{A_{\$:}}}\right)\right) \tag{19}$$

$$\mathrm{proj}(z) = \mathrm{proj}_{(}(z) \tag{20}$$

$$S(\mathbf{h}_{4_8}) = \exp_0(S \log_0(\mathbf{h}_{4_8})) \tag{21}$$

$$\mathbf{h}_{2_8} \oplus \mathbf{h}_{2_9} = \frac{(1 + 2\langle \mathbf{h}_{2_8}, \mathbf{h}_{2_9} \rangle + \|\mathbf{h}_{2_9}\|^2)\mathbf{h}_{2_8} + (1 - \|\mathbf{h}_{2_8}\|^2)\mathbf{h}_{2_9}}{1 + 2\langle \mathbf{h}_{2_8}, \mathbf{h}_{2_9} \rangle + \|\mathbf{h}_{2_8}\|^2\|\mathbf{h}_{2_9}\|^2} \tag{22}$$

where both the weight matrix $\mathcal{G}_{A_{O:}}$ and bias $b_{A_{O:}}$ are specific to each relation $r$ in $\mathcal{G}_\$$ and reserved for the ontology $\mathcal{G}_{A_{O:}}$.

Bridge Node Loss Function. The bridge-node model uses a hinge loss function as a combination of the entity's ontology-specific loss, $\mathrm{ontoLoss}_{\mathcal{G}_{A_{O:}}}$, and instance-specific loss, $\mathrm{instLoss}_{\mathcal{G}_{A:}}$, that is maximized for $\mathcal{G}_{[\mathcal{G}_\$}$ which contains all triples associated with the bridge nodes. Positive triples are denoted $pt_{\cdot} = (\mathbf{h}_{4_8}, r) \cdot A_{\cdot:} \cdot \mathbf{h}_{4_9}$ and $pt_{\cdot\$} = (\mathbf{h}_{4_8}, r) \cdot \mathbf{h}_{A_{\$:}} \cdot \mathbf{h}_{2_9}$, and negative triples are denoted $nt_{\cdot} = (\mathbf{h}_{4_8}', r) \cdot A_{\cdot:} \cdot \mathbf{h}_{4_9}'$ and $nt_{\cdot\$} = (\mathbf{h}_{4_8}', r) \cdot \mathbf{h}_{A_{\$:}} \cdot \mathbf{h}_{2_9}'$ with loss function defined as follows:

$$\begin{aligned} &\mathrm{ontoLoss}_{\mathcal{G}_{A_{O:}}}(pt_{B \cdot O}, nt_{B \cdot O}) \\ &\quad = \max(0, \dot{W} - q_{>1B}(pt_{\cdot O}) + q_{2>A\tilde{A}}(nt_{\cdot O})) \end{aligned} \tag{23}$$

$$\begin{aligned} &\mathrm{instLoss}_{\mathcal{G}_{A:}}(pt_{B \cdot I}, nt_{B \cdot I}) \\ &\quad = \max(0, \dot{W} - q_{\mathrm{obs}}(pt_{\cdot}) + q_{\mathrm{corr}}(nt_{\cdot})) \end{aligned} \tag{24}$$

$$\mathcal{L}_{\mathrm{bridge}}(\mathcal{G}_{[\mathcal{G}_\$}) = \frac{1}{|\mathcal{G}_{[\mathcal{G}_\$}|} \sum_{pt_{B \cdot I}, pt_{B \cdot O} \in \mathcal{G}_{[\mathcal{G}_\$},\, nt_{B \cdot I}, nt_{B \cdot O} \in \mathcal{G}_{[\mathcal{G}_\$}} \mathrm{ontoLoss}_{\mathcal{G}_{A_{O:}}}(pt_{B \cdot O}, nt_{B \cdot O}) + \mathrm{instLoss}_{\mathcal{G}_{A:}}(pt_{B \cdot I}, nt_{B \cdot I}) \tag{25}$$

The combination loss function above enables bridge nodes to learn from both spaces of intersection $\mathcal{S}^3$ and $\mathcal{H}^3$.

## 3.2 Training

This section details the training framework of DGS, for representing two-view KGs, described in Section 3.1. We describe, for each epoch, the training of each node in the two-view KG which includes (1) the forward propagation step, (2) the loss function, and (3) the backward propagation step to optimize parameters. Algorithm 1 provides a summary of the framework.

Embedding Initialization. We randomly initialize all embeddings of $\mathcal{H}^2$ and $\mathcal{A}$ in polar coordinates: $\theta_{4_8} \in \mathbb{R}^{3\ 1} \sim \text{Unif}(0, 2\pi)^{3\ 1}$ and $\theta_{A_{O:}} \in \mathbb{R}^{3\ 1} \sim \text{Unif}(0, 2\pi)^{3\ 1}$, then convert entity embeddings to their corresponding Cartesian representation: $h_{4_8} = \text{MPC}(\theta_{4_8})$. Link $h_{A_{O:}}$ is a randomly sampled position on the Poincare disk. Refer to Section A in Supplements for details about the conversion procedure, describing both polar-Cartesian coordinate conversion (MPC) and Cartesian-polar coordinate conversion (MCP). For the instance-view model, we also choose a value for norm, $r$, that is sampled from a uniform distribution: $r \in \mathbb{R}^{|E|} : r \in (0, 1) \sim \text{Unif}(0, 1)$ for all entities, and for the ontology-view model, we choose a value for norm, $r_{h_{2_8}}$, assigned uniformly at random per entity. We set the curvature values of the spherical and hyperbolic spaces as $\zeta = 1$ and $\kappa = -1$ respectively. We leave the non-trivial problem of learning optimal curvatures as future work.

Training Procedure for Instance View KG . Parameter optimization is performed using Riemannian stochastic gradient descent (RSGD) for the spherical space as follows for entity embedding and relational embedding updates respectively. To ensure that the updated entity embedding remains in the norm-$r$ space, we perform a rescaling operation, $\text{proj}_\zeta$, to project out-of-boundary embeddings back to the surface of the $r$-ball.

$$\text{proj}_\zeta(z) = \begin{cases} r \frac{z}{\|z\|} & \text{if } \|z\| > r \\ z & \text{otherwise} \end{cases} \tag{26}$$

$$\text{RiemannGrad}_{h_{4_{8,C}}}(\nabla_{\text{inst}}) = \left(\frac{1 - \|h_{4_{8,C}}\|^2}{2}\right)^2 \nabla_{\text{inst}} \tag{27}$$

$$h_{4_{8,C+1}} \leftarrow \text{proj}_\zeta\left(\exp_{h_{4_{8,C}}}\left(-\eta \nabla_{\text{inst}} \text{RiemannGrad}_{h_{4_{8,C}}}(\nabla_{\text{inst}})\right)\right) \tag{28}$$

$$\theta_{A_{:,C+1}} \leftarrow \exp_{\theta_{A_{:,C}}}\left(-\eta \nabla_{\text{inst}} \text{RiemannGrad}_{\theta_{A_{:,C}}}(\nabla_{\text{inst}})\right) \tag{29}$$

Training procedure for concepts. Parameter optimization is performed using RSGD for the hyperbolic space as follows for concept embedding and relational embedding updates respectively, where the corresponding concept norm space, $r_{h_{2_8}}$, is also learned through RSGD by updating embeddings of $h_{2_8}$. Diagonal relational matrix $X$ is also updated through RSGD, and we update scalar biases $\theta_{2_{8,C+1}}$ and $\theta_{2_{9,C+1}}$ through stochastic gradient descent.

$$h_{2_{8,C+1}} \leftarrow h_{2_{8,C}} \left(\exp_{h_{2_{8,C}}}\left(-\eta \left(\frac{1 - \|h_{2_{8,C}}\|^2}{2}\right)^2 \nabla_{\text{onto}}\right)\right) \tag{30}$$

$$h_{A_{O:,C+1}} \leftarrow h_{A_{O:,C}} \left(\exp_{h_{A_{O:,C}}}\left(-\eta \left(\frac{1 - \|h_{A_{O:,C}}\|^2}{2}\right)^2 \nabla_{\text{onto}}\right)\right) \tag{31}$$

$$X_{C+1} \leftarrow X_C \left(\exp_{X_C}\left(-\eta \left(\frac{1 - \|X_C\|^2}{2}\right)^2 \nabla_{\text{onto}}\right)\right) \tag{32}$$

$$\theta_{2_{8,C+1}} \leftarrow \theta_{2_{8,C}} \left(\exp_{\theta_{2_{8,C}}}\left(-\eta \nabla_{\text{onto}}\right)\right) \tag{33}$$

$$\theta_{2_{9,C+1}} \leftarrow \theta_{2_{9,C}} \left(\exp_{\theta_{2_{9,C}}}\left(-\eta \nabla_{\text{onto}}\right)\right) \tag{34}$$

After the epoch's update of concept embeddings, we once again reset the value of the last dimension of $\mathcal{H}^2$ to 0 to satisfy the original framework constraint of the Poincaré disk. We also enforce that the angular dimensions of relational embeddings are in $(0, 2\pi)$.

Training Procedure for Bridge Nodes . Parameter optimization is performed using RSGD for the bridge space as follows for bridge entity embedding and relational embedding updates respectively. Ontology optimization, ontoOpt$^{-1}$, and instance optimization, instOpt$^{-1}$, are performed alternatively in batches for each of the two embedding types according to the type of link that the embedding is associated with, e.g., $\in$ or $\ni$. This enables the representation of bridge nodes to be informed by both $\mathcal{S}^2$ and $\mathcal{H}^2$.

$$\text{ontoOpt}^{-1}(z) = z \left(\exp_z\left(-\eta \left(\frac{1 - \|z\|^2}{2}\right)^2 \nabla_{\text{bridge}}\right)\right) \tag{35}$$

$$\text{instOpt}^{-1}(z) = \left(\exp_{A^{-1}(z)}\left(-\eta \nabla_{\text{bridge}}\right)\right) \tag{36}$$

$$h_{4_{8,C+1}} \leftarrow \text{proj}\left(\text{ontoOpt}^{-1}\left(h_{4_{8,C}}\right)\right) \| \text{proj}\left(\text{instOpt}^{-1}\left(h_{4_{8,C}}\right)\right) \tag{37}$$

$$h_{A_{O:,C+1}} \leftarrow \text{ontoOpt}^{-1}\left(h_{A_{O:,C}}\right) \tag{38}$$

$$\theta_{A_{:,C+1}} \leftarrow \text{instOpt}^{-1}\left(\theta_{A_{:,C}}\right) \tag{39}$$

After each epoch's update of the instance optimization for $h_{4_8}$, the value of the last dimension of $\mathcal{H}^3$ is reset to 0 and rescaled with $\text{proj}^{-1}(z)$ de ned to be the same as $\text{proj}_\zeta(z)$ to ensure the intersection space constraint of the bridge entity model. Angular dimensions are also enforced to be in $[0, 2\pi)$.

**Algorithm 1:** Overall training procedure of DGS. * indicates that the three steps can be performed in any order.

Input : set of entities $\mathcal{E}$; set of relations $\mathcal{A}$
instance-view entity to entity triples with links, $\theta_{\mathcal{A}_{:}}$
ontology-view concept to concept triples with links, $h_{\mathcal{A}_{O:}}$
bridge entity to concept triples with links, ontology $h_{\mathcal{A}_{O:}}$
bridge entity to entity triples with links, instance $\theta_{\mathcal{A}_{:}}$
Output: Updated embeddings $\mathcal{S}$: $\theta_{\mathcal{A}_{:,EP}}$, $h_{\mathcal{A}_{O:,EP}}$, ontology $h_{\mathcal{A}_{O:,EP}}$, and instance $\theta_{\mathcal{A}_{:,EP}}$ at nal epoch EP

```
for epoch ∈ (1, 2, ..., EP) do
    Step 1*:
    Sample links from θ_{A_:}
    Perform spherical update of entities: Section 3.1.1
    Step 2*:
    Sample links from h_{A_O:}
    Perform hyperbolic update of concepts: Section 3.1.2
    Step 3*:
    Sample links from ontology h_{A_O:} and instance θ_{A_:}
    Alternatively perform spherical and hyperbolic updates of
      bridge nodes: Section 3.1.3
end
return θ_{A_:,EP}, h_{A_O:,EP}, ontology h_{A_O:,EP}, and instance θ_{A_:,EP}
```

## 4 EXPERIMENTS

In this section, we evaluate DGS on two KG tasks: the triple completion task on each of the instance and ontology views of the KG and the entity typing task to test quality of the learned bridge space in communicating between each view of the KG. We also provide a case study on entity typing for different variants of DGS by embedding on other combinations of geometric spaces. Further, we provide a visualization of embeddings before and after the learning process projected onto the 3-D geometric space of DGS.

Table 1: Dataset statistics for entities, concepts and their relations. denotes entity-entity links, denotes concept-concept links, and - denotes entity-concept links.

| Dataset | Nodes | | Relations | | |
|---|---|---|---|---|---|
| | # | # | # - :' | # - :' $ | # - :' $ |
| YAGO26K-906 | 26,078 | 906 | 390,738 | 8,962 | 9,962 |
| DB111K-174 | 111,762 | 174 | 863,643 | 763 | 99,748 |

Table 2: Data splits for triple completion and entity typing. We provide splits for all KG triples in' •' $ •' $ for train(tr), validation(v), and test(ts).

| | YAGO26K-906 | | |
|---|---|---|---|
| Task | Tr(' •' $ •' $ ) | V(' •' $ •' $ ) | Ts(' •' $ •' $ ) |
| Triple Completion | 332,128/7,618/8,691 | 19,536/448/485 | 39,074/896/1,019 |
| Entity Typing | 211,346/4,876/5,379 | 23,549/543/598 | 156,311/3,592/3,985 |
| | DB111K-174 | | |
| Task | Tr(' •' $ •' $ ) | V(' •' $ •' $ ) | Ts(' •' $ •' $ ) |
| Triple Completion | 734,096/648/84,864 | 43,182/38/5,018 | 86,365/77/10,131 |
| Entity Typing | 466,538/462/53,863 | 51,828/46/5,985 | 345,504/337/39,900 |

## 4.1 Datasets

We utilize the datasets of YAGO26K-906 and DB111K-174 since they have the two-view KG setting unlike other datasets for KG embeddings that consider solely an instance-view [24] or ontology-view [10]. YAGO26K-906 and DB111K-174 are prepared from [12], which are extracted from YAGO [20] and DBpedia [16] respectively. Refer to [12] for the detailed construction process. Table 1 provides dataset statistics and Table 2 provides data splits for both datasets on both KG tasks. It can be observed that the instance-view contains many more triples than the ontology-view and that DB111K-174 contains a larger proportion of entity-concept triples (10.35%) compared to YAGO26K-906 (2.43%).

## 4.2 Models

4.2.1 Baselines. We compare DGS to state-of-the-art neural network embedding models, which include Euclidean, non-Euclidean, and product space KGE models, as well as GNN-based models for KG completion and entity typing.

TransE [6], one of the first KGE models, which simply captures the relationship between entities as a translation.
DistMult [28], a matrix factorization KGE model, modeling the relationship between entities via multiplication.
ComplEx [25], a KGE model that extends DistMult into the complex number field.
RotatE [23], a recent KGE model, based on the rotation assumption where a relation is a rotation from the subject to the object in the complex vector space.
JOIE [12] and M2GNN [27]: Refer to Section 2.3 where this is discussed.
HyperKG [15], a KGE model extending translational KGE methods to the Poincaré-ball model of hyperbolic geometry.
HAKE [30], which extends RotatE by having relations combining modulus scaling with rotation.
ConE [1], a KGE model embedding entities into hyperbolic cones and relations as transformations between cones.
RefH /RotH /AttH [8], which are hyperbolic KGE models that combine hyperbolic spaces using hyperbolic attention, where RefH and RotH are variants of AttH using only reflections and rotations respectively.
HGCN [9], a hyperbolic GCN model utilizing Riemannian geometry and the hyperboloid model.
HyperKA [22], which extends GCNs from the Euclidean to hyperbolic space using the Poincaré ball model.

4.2.2 DGS Variants. We describe variant models of DGS below.

DGS-RO-FC, which is DGS with the Riemannian operator (RO) used for vector transformation instead of our proposed closed spherical space vector transformation procedure in Section 3.1.1, and with fixed center (FC) of spherical ball at 0, which is the same center as the Poincaré disk. For single geometric spaces of $S^3$ and $H^3$ for non-bridge nodes and concepts, the Riemannian operator is performed as a retraction operation to the tangent Euclidean spaces. However, we extend the Riemannian operator when performing retraction for the intersection ring of the bridge space, which is formed by intersecting the spherical ball's surface and Poincaré disk. This is described in the Supplements.
DGS-SO-FC, which is DGS with the proposed closed spherical space operator (SO), and with FC of spherical ball at 0, which is the same center as the Poincaré disk.
DGS (ours), which is DGS with the proposed closed SO, and with learnable center (LC) of spherical ball, which for simplicity of constructing the intersection space for bridge nodes, is set to the last dimension to only allow for vertical shift. Note that we do not need to also make the center of the Poincaré disk learnable as this shift is already introduced with making one of the centers learnable. For learning the spherical center $l$, , in the model, we follow the same training procedure for Section 3.2 but for the non-bridge entities and bridge entities, we perform the operations by temporarily shifting to the center 0 (e.g., $l$ shift), then shift back to the new center $l$ (e.g., $l$ shift) after the updates are performed.

4.2.3 DGS Ablation Models. We study different ablations models of DGS, which are formed by utilizing different combinations of manifold spaces for each type of link of DGS(' •' $ ) in the two-view KG including the spherical space $S^3$, , hyperbolic space using the Poincaré model $H^3$, or Euclidean space $E^3$. These include DGS ($S^3$•$S^3$), MGS ($H^3$•$H^3$), DGS ($E^3$•$H^3$), and DGS ($S^3$•$E^3$). Since ' $ is always at the intersection of the two spaces and ' $ , we do not need to specify the geometric space separately.

## 4.3 Evaluation

In this section, we detail our evaluation on the tasks of KG triple completion and entity typing. The goal of triple completion is to construct the missing relation facts in a KG structure. Specifically, we test constructing a missing target node, from each of the ontology or instance views: or queries $^{14}_{8}$• A •?$^{49}_{9}$ and $^{12}_{8}$• A •?$^{29}_{9}$, such that each model evaluated is trained on the entire two-view KG. The goal of entity typing is to predict the concepts that correspond to the entities, or queries $^{14}_{8}$• A •?$^{29}_{9}$.

Using plausibility scores to rank each test candidate set, for each task, we report results for the evaluation metrics of mean reciprocal rank ("'' ) and Hits@, e.g., 8C8@1, 8C8@3, 8C8@10 Table 2 reports our data splits for each task. For triple completion, this is chosen to be embedding distance of the source node to the missing target node modeled under the relation, and for entity typing, this is chosen to be the embedding distance from the entity's representation in the concept space to the concept [12] provides more details

Table 3: Results of KG triple completion. $\mathbb{S}$ denotes the 3-dimensional manifold space, ' are entity links, '$ are concept links. For each group of models, the best results are bold-faced. The overall best results on each dataset are underscored.

| Type | $\mathbb{S}$ | Datasets / Graphs / Metrics | YAGO26K-906 ' KG Completion MRR | H@1 | H@10 | YAGO26K-906 '$ KG Completion MRR | H@1 | H@10 | DB111K-174 ' KG Completion MRR | H@1 | H@10 | DB111K-174 '$ KG Completion MRR | H@1 | H@10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| non-GNN based KGE models | $E^3$ | TransE | 0.187 | 13.73 | 35.05 | 0.189 | 14.72 | 24.36 | 0.318 | 22.70 | 48.12 | 0.539 | 47.90 | 61.84 |
| | $E^3$ | DistMult | 0.288 | 24.06 | 31.24 | 0.156 | 14.32 | 16.54 | 0.280 | 27.24 | 29.70 | 0.501 | 45.52 | 64.73 |
| | $C^3$ | ComplEx | 0.291 | 24.85 | 37.28 | 0.180 | 14.83 | 22.97 | 0.321 | 27.39 | 46.63 | 0.549 | 47.80 | 62.23 |
| | $C^3$ | RotatE | 0.302 | 25.31 | 42.17 | 0.228 | 16.35 | 27.28 | 0.356 | 29.31 | 54.60 | 0.557 | 49.16 | 68.19 |
| | $E^3$ | JOIE | 0.316 | 24.62 | 51.85 | 0.289 | 18.66 | 39.13 | 0.479 | 35.21 | 72.38 | 0.602 | 52.48 | 79.71 |
| | $H^3$ | HyperKG | 0.215 | 18.35 | 36.02 | 0.174 | 14.50 | 23.26 | 0.302 | 23.31 | 46.72 | 0.542 | 47.59 | 62.11 |
| | $H^3$ | HAKE | 0.293 | 23.04 | 40.19 | 0.301 | 19.27 | 41.09 | 0.391 | 31.10 | 60.46 | 0.638 | 55.69 | 81.07 |
| | $H^3$ | ConE | 0.299 | 23.56 | 41.23 | 0.313 | 20.05 | 41.80 | 0.422 | 33.69 | 68.12 | 0.639 | 55.89 | 81.45 |
| GNN-based models | $H^3$ | RefH | 0.282 | 23.19 | 40.52 | 0.298 | 19.70 | 41.26 | 0.407 | 30.06 | 66.93 | 0.622 | 55.35 | 81.09 |
| | $H^3$ | RotH | 0.295 | 23.50 | 41.03 | 0.308 | 19.97 | 41.78 | 0.418 | 30.18 | 67.05 | 0.639 | 55.82 | 81.44 |
| | $H^3$ | AttH | 0.298 | 23.43 | 41.20 | 0.310 | 19.99 | 41.53 | 0.419 | 30.10 | 66.58 | 0.629 | 55.37 | 81.39 |
| | $H^3$ | HGCN | 0.307 | 23.04 | 40.25 | 0.302 | 19.38 | 40.49 | 0.396 | 31.54 | 61.78 | 0.638 | 55.81 | 81.60 |
| | $H^3$ | HyperKA | 0.320 | 26.71 | 52.09 | 0.305 | 18.83 | 40.28 | 0.486 | 35.79 | 72.33 | 0.613 | 53.36 | 80.59 |
| | $P^3$ | M2GNN | 0.347 | 29.63 | 54.28 | 0.341 | 23.70 | 42.19 | 0.506 | 36.52 | 73.11 | 0.644 | 56.82 | 83.01 |
| Ablation variants: DGS(' •'$) | | DGS($S^3$•$S^3$) | 0.338 | 27.15 | 53.20 | 0.318 | 20.36 | 41.02 | 0.491 | 34.58 | 71.40 | 0.606 | 53.29 | 80.17 |
| | | DGS($H^3$•$H^3$) | 0.314 | 25.11 | 52.02 | 0.358 | 24.61 | 43.28 | 0.502 | 35.79 | 73.61 | 0.663 | 57.59 | 84.16 |
| | | DGS($E^3$•$H^3$) | 0.327 | 25.32 | 52.89 | 0.343 | 23.95 | 41.62 | 0.498 | 35.11 | 72.37 | 0.640 | 56.17 | 82.68 |
| | | DGS($S^3$•$E^3$) | 0.322 | 24.91 | 52.36 | 0.297 | 19.43 | 40.61 | 0.484 | 33.29 | 73.54 | 0.619 | 53.72 | 80.51 |
| DGS variants | | DGS-RO-FC | 0.352 | 29.79 | 55.21 | 0.364 | 25.04 | 43.27 | 0.518 | 37.65 | 73.97 | 0.681 | 59.23 | 84.16 |
| | | DGS-SO-FC | 0.364 | 30.15 | 55.93 | 0.369 | 25.81 | 44.18 | 0.536 | 38.29 | 74.28 | 0.687 | 59.26 | 84.82 |
| | | DGS(ours) | 0.366 | 30.15 | 56.06 | 0.372 | 25.88 | 44.38 | 0.536 | 38.31 | 74.85 | 0.690 | 59.88 | 84.82 |

on the evaluation procedure. For evaluation consistency, for both tasks, model training hyperparameters are chosen for dimensionality 3 2 {50, 100, 200, 300} for all triples, learning rate 2 {5e-4, 1e-3, 1e-2, 1e-1} and margin 2 {0.5, 1}. Further, different batch sizes and epochs are used according to the type and size of the graphs.

KG Triple Completion Results. Results are reported in Table 3. DGS outperforms all of the baseline models on both datasets. DGS achieves an average performance gain over all baselines by 32.36% on "" , 27.59% on @1, and 29.17% on @10 for the instance-view completion across both YAGO26K-906 and DB111K-174. DGS achieves an average performance gain over all baselines by 23.43% on "" , 28.41% on @1, and 18.11% on @10 for the ontology-view completion across both YAGO26K-906 and DB111K-174.

It can be observed that in both the instance and ontology views on both datasets, the hyperbolic-based KGE models outperform their Euclidean and complex space counterparts. Further, hyperbolic KGE models perform better on ontology view than instance view likely due to there being prevalence of hierarchy in the ontology. It is also seen that using multiple geometric spaces is more effective than using a single geometric space. For GNN-based models, M2GNN, which uses a product space, $P^3$, combining Euclidean, spherical, and hyperbolic spaces, outperforms the models using only one of the spaces. Compared to M2GNN, the most competitive baseline model in most cases, DGS shows significant improvement of 3.25% on "" , 5.15% on @1, and 2.73% on @10 on average for both YAGO26K-906 and DB111K-174. This is likely because hierarchy in the KG is better modeled through the hyperbolic space than with in uence from the spherical or Euclidean space, and cyclic links are better modeled through the space than with in uence from the hyperbolic or Euclidean space. For bridge entities involved in both hierarchical and cyclic links, we see it is beneficial to model them in an intersection space to better model both of these properties.

Entity Typing Results. Results are reported in Table 4. Consistent with the KG completion results, it can be seen that DGS

Table 4: Results of entity typing. For each group of models, the best results are bold-faced. The overall best results on each dataset are underscored.

| Datasets / Metrics | YAGO26K-906 MRR | Acc. | Hit@3 | DB111K-174 MRR | Acc. | Hit@3 |
|---|---|---|---|---|---|---|
| TransE | 0.144 | 7.32 | 35.26 | 0.503 | 43.67 | 60.78 |
| DistMult | 0.411 | 36.07 | 55.32 | 0.551 | 49.83 | 68.01 |
| ComplEx | 0.519 | 43.08 | 59.28 | 0.583 | 55.07 | 70.17 |
| RotatE | 0.673 | 58.24 | 73.95 | 0.721 | 61.48 | 75.67 |
| JOIE | 0.899 | 85.72 | 96.02 | 0.859 | 75.58 | 96.02 |
| HyperKG | 0.627 | 55.39 | 65.64 | 0.736 | 61.20 | 74.29 |
| HAKE | 0.905 | 87.42 | 96.37 | 0.866 | 76.04 | 96.22 |
| ConE | 0.912 | 87.51 | 96.39 | 0.869 | 76.85 | 96.38 |
| RefH | 0.907 | 87.49 | 96.37 | 0.867 | 76.26 | 96.31 |
| RotH | 0.909 | 87.49 | 96.39 | 0.868 | 76.55 | 96.36 |
| AttH | 0.910 | 87.50 | 96.38 | 0.868 | 76.50 | 96.33 |
| HGCN | 0.905 | 87.44 | 96.37 | 0.867 | 76.11 | 96.27 |
| HyperKA | 0.918 | 87.76 | 96.45 | 0.871 | 76.65 | 96.50 |
| M2GNN | 0.922 | 88.16 | 97.01 | 0.880 | 77.58 | 96.94 |
| DGS($S^3$•$S^3$) | 0.919 | 87.94 | 96.81 | 0.875 | 77.03 | 96.78 |
| DGS($H^3$•$H^3$) | 0.924 | 88.12 | 97.01 | 0.887 | 77.53 | 96.95 |
| DGS($E^3$•$H^3$) | 0.922 | 88.07 | 96.91 | 0.883 | 77.37 | 96.58 |
| DGS($S^3$•$E^3$) | 0.917 | 87.68 | 96.41 | 0.866 | 76.33 | 96.48 |
| DGS-RO-FC | 0.930 | 88.47 | 97.32 | 0.891 | 77.81 | 97.04 |
| DGS-SO-FC | 0.938 | 89.02 | 98.11 | 0.895 | 77.92 | 97.40 |
| DGS(ours) | 0.939 | 89.07 | 98.18 | 0.895 | 77.94 | 97.47 |

variants outperform the baseline models on both datasets for entity typing. DGS achieves an average performance gain over all baselines of 24.49% on " , 26.39% on 2 , and 18.78% on @3 for YAGO26K-906. DGS achieves an average performance gain over all baselines of 14.86% on " , 13.74% on 2 , and 12.15% on @3 for DB111K-174. Compared to the most competitive GNN-based model, M2GNN, DGS has nearly a 2% gain in " for each YAGO26K-906 and DB111K-174.

Ablation Studies. Tables 3 and 4 report our results for ablation studies in models belonging to the category Ablation variants. It can be seen that using the spherical space for links learns a better representation than its hyperbolic or Euclidean space counterparts. This indicates the prevalence of cyclic relations between entities. Utilizing the hyperbolic space for '$ and '$ links learns a better

representation than its spherical or Euclidean space counterparts. This indicates the prevalence of hierarchical relations between concepts and entities-concepts. Interestingly, a fully hyperbolic-space model in general achieves better performance than its fully spherical counterpart, indicating that YAGO26K-906 and DB111K-174 may contain relatively more hierarchical links than cyclic links. Further, learning a better representation in one view bene ts the other view more if there are more $\dot{e}_{\$}$ bridge links. For example, since DB111K-174 contains signi cantly more $\dot{e}_{\$}$ links than YAGO26K-906 shown in Table 1, it can be seen that DGS $(H^3 \bullet H^3 \bullet H^3)$ has better performance against baselines on the instance view of DB111K-174 compared to the instance view of YAGO26K-906.

Visualizations. We project the learned embeddings into 3-D space, and plot seven nodes with their relations in Figure 2. Concepts that are higher in the ontological hierarchy such as *animal* and *person* are closer to the origin compared to lower-level concepts such as *leader*, and entities tend to be farther away from the origin.

Figure 2: Example of embeddings learned by DGS. Solid lines connect entities and concepts as in the original dataset. Dashed lines connect every node to the origin to indicate closeness to origin.

## 5 CONCLUSIONS

We are among the rst to explore utilization of di erent embedding spaces for di erent views of a knowledge graph. Entities in instance view often have cyclical connections and are hereby modeled via spherical embeddings; whereas concepts and bridge entities in ontology view tend to form a hierarchy and are hereby modeled in hyperbolic space. We propose the notion of bridge space to seamlessly model intersection of the two spaces in which bridge entities reside. In addition, we propose a set of closed spherical space operations to eliminate the need of projecting embeddings to the tangent Euclidean space. The resulting model, DGS, is a uni ed framework that signi cantly outperforms all baselines, showing the e ectiveness of capturing di erent structures in the knowledge graph using embedding spaces of di erent curvatures and properties.

Future directions include supporting geometric spaces of learnable curvature (to better capture the knowledge graph structure) and allowing a learnable o set between the origins of the two geometric spaces (to better accommodate uneven entity distribution and/or unbalanced ontology structures) We also aim to extend our dual-space model to a multi-space geometric embedding model.

## 6 ACKNOWLEDGEMENTS

This work was partially supported by NSF III-1705169, 1829071, 1937599, 2106859, 2119643; NIH R35-HL135772; NIBIB R01-EB027650, Amazon Research Awards; Cisco Research Grant USA000EP280889; Picsart Gifts; and Snapchat Gifts. The authors would also like to thank anonymous reviewers for their constructive feedback.

## REFERENCES

[1] Yushi Bai, Rex Ying, Hongyu Ren, and Jure Leskovec. 2021. Modeling Heterogeneous Hierarchies with Relation-speci c Hyperbolic Cones. NeurIPS (2021).
[2] Ivana Bala evi , Carl Allen, and Timothy Hospedales. 2019. Multi-relational Poincaré Graph Embeddings. NeurIPS (2019).
[3] Eugenio Beltrami. 1897. Teoria fondamentale degli spazii di curvatura costante.
[4] Jonathan Berant and Percy Liang. 2014. Semantic Parsing via Paraphrasing. ACL (2014).
[5] Silvère Bonnabel. 2013. Stochastic gradient descent on Riemannian manifolds. IEEE (2013).
[6] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. NeurIPS (2013).
[7] James W. Cannon, William J. Floyd, Richard Kenyon, and Walter R. Parry. 1997. Hyperbolic Geometry
[8] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré . 2020. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. ACL (2020).
[9] Ines Chami, Rex Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic Graph Convolutional Neural Networks. NeurIPS (2019).
[10] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. AAAI.
[11] Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. 2018. Learning mixed-curvature representations in product spaces. ICLR
[12] Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. 2019. Universal Representation Learning of Knowledge Bases by Jointly Embedding Instances and Ontological Concepts. KDD (2019).
[13] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge Graph Embedding Based Question Answering. WSDM (2019).
[14] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge Graph Embedding Based Question Answering. WSDM (2019).
[15] Prodromos Kolyvakis, Alexandros Kalousis, and Dimitris Kiritsis. 2019. HyperKG: Hyperbolic Knowledge Graph Embeddings for Knowledge Base Completion. arXiv preprint arXiv:1908.04895 (2019).
[16] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes , Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Soren Auer, and Christian Bizer. 2015. DBpedia: A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web Journal 6 (2015), 167 195.
[17] Yu Meng, Jiaxin Huang, Guangyuan Wang, Chao Zhang, Honglei Zhuang, Lance Kaplan, and Jiawei Han. 2019. Spherical Text Embedding. NeurIPS (2019).
[18] Maximilian Nickel and Douwe Kiela. 2017. Poincaré Embeddings for Learning Hierarchical Representations. NeurIPS (2017).
[19] Peter Petersen, S. Axler, and K.A. Ribet. 2006. Riemannian Geometry
[20] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. WWW.
[21] Li Sun, Zhongbao Zhang, Junda Ye, Hao Peng, Jiawei Zhang, Sen Su, and Philip S. Yu. 2022. A Self-supervised Mixed-curvature Graph Neural Network. AAAI (2022).
[22] Zequn Sun, Muhao Chen, Wei Hu, Chengming Wang, Jian Dai, and Wei Zhang. 2020. Knowledge Association with Hyperbolic Knowledge Graph Embeddings. ACL (2020).
[23] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. ICLR (2019).
[24] Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality
[25] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. ICML (2016).
[26] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. WSDM (2019).
[27] Shen Wang, Xiaokai Wei, Cícero Nogueira dos Santos, Zhiguo Wang, Ramesh Nallapati, Andrew Arnold, Bing Xiang, Philip S. Yu, and Isabel F Cruz. 2021. Mixed-Curvature Multi-Relational Graph Neural Network for Knowledge Graph Completion. WWW (2021).
[28] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. ICLR (2015).
[29] Shuai Zhang, Yi Tay, Wenqi Jiang, Da-cheng Juan, and Ce Zhang. 2021. Switch spaces: Learning product spaces with sparse gating. arXiv preprint arXiv:2102.08648 (2021).
[30] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. AAAI (2020).

# A GENERAL POLAR-CARTESIAN CONVERSION FRAMEWORK

In this section, we detail our designed general procedures for e - ciently converting arbitrary3-dimensional embeddings between the polar and Cartesian coordinate representations, which are utilized by sub-models inDGS. These procedures include: (1) the Cartesian to polar coordinate transformation (MCP) and (2) the polar to Cartesian coordinate transformation (MPC).

Figure 3: Illustration of a vectorz in the spherical space on a 3-dimensional surface. Angular dimensions ofz are\ 1, which is the distance from theG1-axis toz projected onto theG1-G2 plane, and \ 2, which is the distance from theG3-axis toz projected onto the G2-G3 plane.

## A.1 MCP Transformation Procedure.

Using Figure 3, we derive transformations for the 3-dimensional spherical surface, and extend this to3adimensional spherical surface as follows:

$$\tan^{-1}(\cdot)^{o} = \frac{G_{\cdot 1}}{G_{\cdot 2}}; \text{rad} = F$$

$$\cot^{-1}(\cdot)^{o} = \frac{G_{\cdot 3}}{\sqrt{G^2_{\cdot 1} + G^2_{\cdot 2}}}; \text{rad} = F$$

$$\cot^{-1}(\cdot)^{o} = \frac{G_{\cdot 4}}{\sqrt{G^2_{\cdot 1} + G^2_{\cdot 2} + G^2_{\cdot 3}}}; \text{rad} = F$$

$$\vdots$$

$$\cot^{-1}(\cdot)^{o} = \frac{G_{\cdot 3}}{\sqrt{G^2_{\cdot 1} + G^2_{\cdot 2} + \cdots + G^2_{\cdot 3-1}}}; \text{rad} = F$$

$$\Rightarrow$$

$$(\cdot)_1 = \tan^{-1}\frac{G_{\cdot 2}}{G_{\cdot 1}}; \text{rad} = F$$

$$(\cdot)_8 = \cot^{-1}\frac{G_{\cdot 8,1}}{\sqrt{\sum_{8'=1}^{8} G^2_{\cdot 8'}}} \cdot 82 \gg 2 \cdot 3 \quad 1¼\text{rad} = F$$

## A.2 MPC Transformation Procedure.

Using Figure 3, we derive transformations for the3dimensional spherical surface, as follows with:

$$G_{\cdot 1} = \cos^{-1}(\cdot)^{o}; \text{rad} = F$$

$$G_{\cdot 3-8} = \sin^{-1}(\cdot)_{8'}^{o} \quad \sin^{-1}(\cdot)_{3-8-1}^{o}; \text{rad} = F$$

$$G_{\cdot 3-1} = \cos^{-1}(\cdot)_{3-1}^{o}; \text{rad} = F$$

Theorem. The MCP procedure and MPC procedure are closed in the spherical space.

Proof. The above equations for the MCP and MPC procedures directly use properties of trigonometry for right triangles for each embedding dimension$_8 \cdot 82 \gg 1 \cdot 3 \quad 1¼$ ensuring that the transformation is closed.

# B BATCH VECTOR TRANSFORMATION PROCEDURE

We illustrate the batch version of the rotation operation presented in Section 3.1.1. Consider for entity$_{4_8}$, there are relations connected to$4_8$. Instead of applying$_{\text{rot}}$ to each one of the relations, we can perform the following to compute the result e ciently:

$$\mathcal{R}_{\text{rot}}(\cdot h_{4_8}) = \begin{bmatrix} \mathcal{R}_{\text{rot}}(\cdot h_{4_8})(\cdot A_1 \cdot 1^{o}, \cdots, \cdot A_1 \cdot 3^{o}) \\ \vdots \\ \mathcal{R}_{\text{rot}}(\cdot h_{4_8})(\cdot A \cdot 1^{o}, \cdots, \cdot A \cdot 3^{o}) \end{bmatrix} \bmod 2\pi$$

$$= \begin{bmatrix} \mathcal{R}_{\text{rot}}(\cdot h_{4_8})(\cdot A_1)^{o} \\ \vdots \\ \mathcal{R}_{\text{rot}}(\cdot h_{4_8})(\cdot A)^{o} \end{bmatrix}$$

Theorem. The batch vector transformation procedure is closed.

Proof. Each relation has$_{\text{rot}}{}^{1}{}^{o}$ applied to the entity, and$_{\text{rot}}{}^{1}{}^{o}$ has been shown to be closed because every dimension of the transformed head is within the angular constraints e.g., by$2\pi$, and norm space constraints because$_{\text{rot}}{}^{1}{}^{o}$ preserves the norm space of the spherical space.

# C RIEMANNIAN RETRACTION OPERATION FOR THE BRIDGE SPACE OFDGS

Figure 4 describes the Riemannian retraction operation$_R$ for the bridge space to the tangent Euclidean space. The bridge space is formed by intersecting the surface of the spherical ball with the Poincaré disk in the hyperbolic space. As shown in the gure, we separately map the Poincaré disk and the spherical ball to di erent tangent Euclidean spaces, and enable for joint communication between the two spaces to represent the bridge entities. On the left, we perform isomorphic mapping from the Poincaré disk$_{\mathbb{H}^3}$, to the hyperboloid model$_{\mathbb{B}^{3,1}}$ as in [9] using:

$$\mathcal{R}_{\mathbb{H}^3 \to \mathbb{B}^{3,1}}(G_{\cdot 1} \cdots G_{\cdot 3}) = \frac{2G_{\cdot 1} \cdots 2G_{\cdot 3-1}, k \times k_2^2}{1 \quad k \times k_2^2} \tag{40}$$

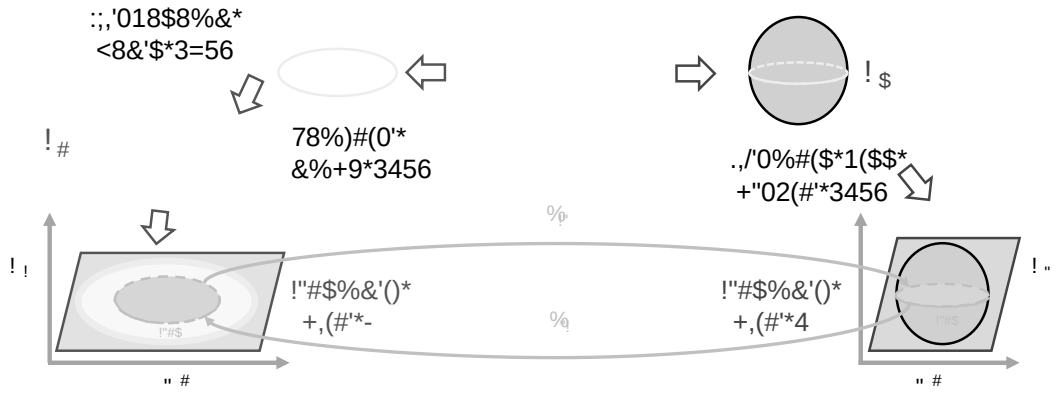


**Figure 4:** Riemannian retraction operation for the bridge space of **DGS**

$$\mathcal{R}_{\mathcal{B}^{3+1}\to\mathcal{H}^{3}}(G_1,\ldots,G_{3+1}) = \frac{(G_1,\ldots,G_3)}{G_{3+1}+1} \tag{41}$$

, then using logarithmic mapping map to the tangent Euclidean space 1. On the right, we use logarithmic mapping to map the spherical ball to the tangent Euclidean space. Then we allow for transformation between the two Euclidean spaces through transformation mappings $W_{UV}$ and $W_{VU}$ such that

$$W_{UV} = W_V \circ W_U^{-1} \tag{42}$$

$$W_{VU} = W_U \circ W_V^{-1} \tag{43}$$