# Mixed-Curvature Multi-Relational Graph Neural Network for Knowledge Graph Completion

Shen Wang*
University of Illinois at Chicago
swang224@uic.edu

Xiaokai Wei
AWS AI
xiaokaiw@amazon.com

Cícero Nogueira dos Santos
AWS AI
cicnog@amazon.com

Zhiguo Wang
AWS AI
zhiguow@amazon.com

Ramesh Nallapati
AWS AI
rnallapa@amazon.com

Andrew Arnold
AWS AI
anarnld@amazon.com

Bing Xiang
AWS AI
bxiang@amazon.com

Philip S. Yu
University of Illinois at Chicago
psyu@uic.edu

Isabel F. Cruz
University of Illinois at Chicago
isabelcfcruz@gmail.com

## ABSTRACT

Knowledge graphs (KGs) have gradually become valuable assets for many AI applications. In a KG, a node denotes an entity, and an edge (or link) denotes a relationship between the entities represented by the nodes. Knowledge graph completion infers and predicts missing edges in a KG automatically. Knowledge graph embeddings have shed light on addressing this task. Recent research embeds KGs in hyperbolic (negatively curved) space instead of conventional Euclidean (zero curved) space and is effective in capturing hierarchical structures. However, as multi-relational graphs, KGs are not structured uniformly and display intrinsic heterogeneous structures. They usually contain rich types of structures, such as hierarchical and cyclic typed structures. Embedding KGs in single-curvature space, such as Euclidean or hyperbolic space, overlooks the intrinsic heterogeneous structures of KGs, and therefore cannot accurately capture their structures. To address this issue, we propose Mixed-Curvature Multi-Relational Graph Neural Network ($M^2$GNN), a generic approach that embeds multi-relational KGs in a mixed-curvature space for knowledge graph completion. Specifically, we define and construct a mixed-curvature space through a product manifold combining multiple single-curvature spaces (e.g., spherical, hyperbolic, or Euclidean) with the purpose of modeling a variety of structures. However, constructing a mixed-curvature space typically requires manually defining the fixed curvatures, which needs domain knowledge and additional data analysis. Improperly defined curvature space also cannot capture the structures of KGs accurately. To address this problem, we set mixed-curvatures as trainable parameters to better capture the underlying structures of the KGs. Furthermore, we propose a Graph Neural Updater by leveraging the heterogeneous relational context in mixed-curvature space to improve the quality of the embedding. Experiments on three KG datasets demonstrate that the proposed $M^2$GNN can outperform its single geometry counterpart as well as state-of-the-art embedding methods on the KG completion task.

## CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; **Natural language processing**.

## KEYWORDS

Knowledge Graph Completion, Knowledge Graph Embedding, Multi-relational Graph, Graph Neural Network, Non-Euclidean Space Embedding

**ACM Reference Format:**
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. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3442381.3450118

## 1 INTRODUCTION

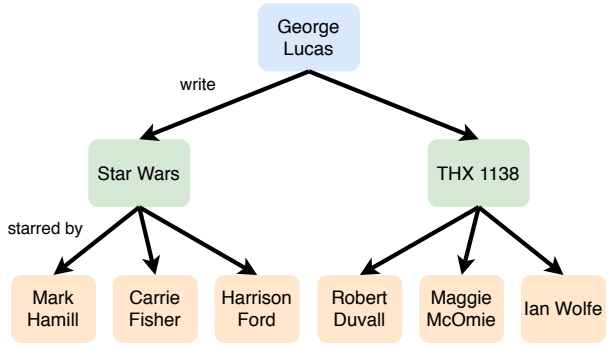
Knowledge graphs (KGs) play a key role in many semantic applications. In fact, reasoning with KGs is a popular research direction, with innovations that improve various downstream applications, such as semantic search (search information with meaning and not only with lexical matching) [3, 4], dialogue generation (generate responses for conversational agents) [22, 25], recommender systems (predict users' preferences) [16, 20, 51, 55], and question answering (answer questions posed in natural language automatically) [13, 25]. However, real-world KGs are often highly incomplete, thus making the task of link prediction between entities—*knowledge graph (KG) completion*—attract considerable attention.

Recent literature shows that embedding methods are powerful tools for KG completion [24, 52]. The basic idea is to map the entities/relations to a latent (and usually low-dimensional) vector space while preserving the semantics and inherent structures for link prediction in the KG. In fact, existing approaches, such as TransE [5] and STransE [35] map the KG data into a latent vector space and use Euclidean distance to measure similarity.

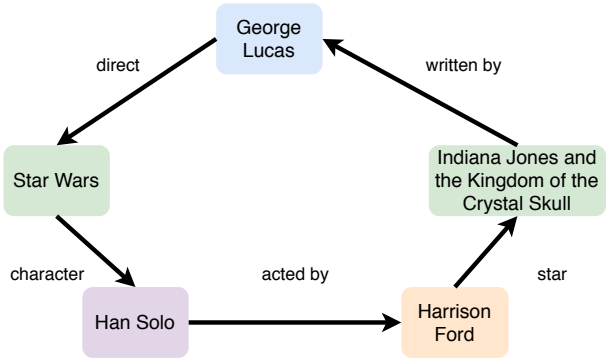
*Work done during an internship at the AWS AI.


*WWW '21, April 19–23, 2021, Ljubljana, Slovenia*

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450118




**(a) This KG example demonstrates a hierarchical (tree-like) structure. There are three types of KG entities (writer, movie, and actor) and two types of relations (write and starred by), where entities are connected in a hierarchic structure, e.g., *George Lucas* is the parent of *Star Wars*, therefore exhibiting a higher level than *Star Wars* in the hierarchy.**




**(b) This KG example demonstrates a cyclic structure. There are four types of KG entities (director, writer, movie, cast, and actor) and five types of relations (direct, character, acted by, star, and written by), where each entity connects with another entity in a cycle, in a way that *George Lucas* is at the same level as the rest of the entities.**

**Figure 1: Two KG examples demonstrate hierarchical and cyclic structures, respectively.**

KGs usually contain rich types of structures, such as hierarchical and cyclic structures. Figure 1a and Figure 1b demonstrate two KG examples showing hierarchical and cyclic structures, respectively. As shown in Figure 1a, in the triple *(George Lucas, write, Star Wars)*, *George Lucas* is a parent of *Star Wars*, therefore exhibiting a higher level than *Star Wars* in the hierarchy. However, in Figure 1b, each entity connects with another entity in a cycle, in a way that *George Lucas* is at the same level as *Star Wars/Han Solo/Harrison Ford/Indian Jones and the Kingdom of the Crystal Skull*. Existing Euclidean space approaches overlook the existence of these two particular structures and have limited representation ability to capture them.

More recently, KG embeddings in the Poincaré ball model of hyperbolic space, a non-Euclidean space, have been devised [1, 6]. They demonstrate boosted performance for KGs with rich hierarchical structures. However, because the focus of hyperbolic spaces is on capturing hierarchical structures and they overlook other structure types, the performance drops for KGs with limited hierarchical structures. KGs are usually not structured uniformly and demonstrate intrinsic heterogeneity, as in the previous examples. Therefore, a new approach is needed, which can capture different types of structures.

A number of approaches have been developed for embedding KGs in non-Euclidean space (constant non-zero curvature space), such as spherical (positively curved) space [31, 57] and hyperbolic (negatively curved) space [10, 36]. They exhibit improved representation ability to model specific types of structured data when compared with their Euclidean counterparts, such as the hierarchical or cyclic structures. To benefit from both spherical space and hyperbolic space, the embedding on a "non-constant" curved manifold is constructed by a product of constant curvature Riemannian manifolds [17, 41]. However, these methods focus mainly on unsupervised embeddings and they only consider homogeneous relation (with only one relation type), which cannot be applied directly to address the multi-relational KG embedding problem. Furthermore, constructing a constant non-zero curvature space typically requires manually defining the fixed curvatures using domain knowledge and additional data analysis. An improperly defined curvature cannot capture the structures of a KG accurately.

The Graph Neural Network (GNN) has received wide attention in the past few years [8, 11, 15, 21, 26, 29, 48, 50, 53, 54, 58, 61]. By leveraging rich context information, it may be possible to improve the quality of the learned embeddings significantly. Recently, a few attempts have been made to learn the KG embeddings via a graph neural network. Examples are the graph convolutional network [40, 47] and the graph attention network [2, 34]. These methods can learn the KG embedding automatically in a data-driven way. However, to the best of our knowledge, no attempt has been made to combine GNN with non-Euclidean space to address the multi-relational KG embedding problem.

To address the issues above, we propose the Mixed-Curvature Multi-Relational Graph Neural Network ($M^2$GNN), a generic graph neural network framework to embed multi-relational KG data in the mixed-curvature space for KG completion. In particular, we define and construct a mixed-curvature space through a Riemannian product manifold combining multiple single-curvature spaces (i.e., Euclidean with zero curvature, spherical with positive curvature, and hyperbolic with negative curvature) with a decomposable Riemannian distance function. By fusing multiple single-curvature spaces, a new space is constructed with a non-constant heterogeneous curvature. The constructed mixed-curvature space is more flexible and can better match the intrinsic heterogeneous data structures of the KGs and thus generate higher quality representations for a variety of KG data types with varying structures (e.g., hierarchical or cyclic). We note the already mentioned difficulties associated with a hand-engineered curvature space and introduce trainable heterogeneous curvatures. We also note the importance of the rich heterogeneous relational context and propose a GNN-based graph neural updater, which can adaptively integrate the relational context. Learned embeddings can better exploit the data heterogeneous structures by leveraging the multi-hop information in the mixed-curvature space. Experimental results show that the proposed $M^2$GNN can indeed recover non-uniform curvatures and outperform state-of-the-art methods on the benchmark datasets.

In summary, in this paper we make the following contributions:

- We formulate the problem of KG embedding in a mixed-curvature space for KG completion and develop a multi-relational graph neural network framework, which can benefit from the mixed-curvature geometry and graph neural network. To the best of our knowledge, we are the first to apply mixed-curvature geometry and graph neural network in tackling the KG completion problem.
- We propose the mixed-curvature space with trainable heterogeneous curvatures and space weights to embed the multi-relational KG data, which can better capture intrinsic heterogeneous structure in the KG.
- We generalize the graph neural network to the multi- relational and mixed-curvature settings, which can overcome the limitation of the translational distance model by more effectively leveraging the heterogeneous relational context.
- We conduct extensive experiments on three different KG datasets and demonstrate that the proposed method outperforms its single geometry counterpart and existing state-of-the-art embedding methods on the KG completion task.

The rest of the paper is organized as follows. We introduce preliminaries and the problem formulation in Section 2. Section 3 discusses in detail our proposed mixed-curvature multi-relational graph neural network for KG completion. Section 4 provides the experiment results and ablation study. Section 5 discusses related work. Finally, Section 6 concludes the paper.

## 2 PRELIMINARIES AND PROBLEM FORMULATION

In this section, we first present the preliminaries and notation required to define the mixed-curvature space (more details can be found in textbooks [28, 39]), including the Riemannian manifold, constant-curvature space. Then we define the problem of mixed-curvature multi-relational graph embedding for KG completion.

### 2.1 Riemannian Manifold

A manifold $\mathcal{M}$ of dimension $d$ is a generalization of higher dimensions of a surface. Let $\mathbf{p} \in \mathcal{M}$ be a point, $\mathcal{M}$ is associated with a tangent space $T_p\mathcal{M}$ — a $d$-dimensional vector space that approximates $\mathcal{M}$ around $\mathbf{p}$. A Riemannian metric assigns to each $\mathbf{p}$ a positive-definite inner product $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$, along with a norm $|\cdot|_p : T_p\mathcal{M} \to \mathbb{R}$ defined by $|v|_p = \sqrt{g_p(v, v)}$. A manifold $\mathcal{M}$ equipped with the metric $g$ is a *Riemannian Manifold*, denoted $(\mathcal{M}, g)$. In particular, $g$ is used to define the distance (geodesics) of two points on the manifold. A given $g$ also defines a *curvature* $K$ at each point, which determines how the space is curved.

### 2.2 Constant-curvature Spaces

A constant-curvature space $\mathcal{M}_K^d$ is a Riemannian manifold with curvature $K \in \mathbb{R}$ and dimension $d \geq 2$, such that $\mathcal{M}_K^d = \{\mathbf{x} \in \mathbb{R}^d : \langle \mathbf{x}, \mathbf{x} \rangle_I = 1/K\}$ with $I$ denoting the inner product type. Specifically, there exist three different types of constant-curvature space $\mathcal{M}$ with respect to the sign of the curvature: hypersphere $\mathbb{S}_K$ (positively curved space); Euclidean space $\mathbb{E}$ (flat space); and hyperboloid $\mathbb{H}_K$ (negatively curved space). Formally, these constant-curvature spaces are defined in Table 1. As the operations in some constant-curvature spaces (hyperbolic and spherical spaces) are different from those in Euclidean space, we also cover these operations and summarize them in Table 2.

**Table 1: Definitions of three types of constant-curvature spaces. $\langle \cdot, \cdot \rangle_2$ denotes the standard Euclidean inner product and $\langle \cdot, \cdot \rangle_{\mathcal{L}}$ denotes the Lorentz inner product, such that $\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} = -x_1 y_1 + \sum_{i=2}^{d+1} x_i y_i, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^{d+1}$.**

| Space | Curvature K | Riemannian Manifold $\mathcal{M}$ |
|---|---|---|
| $\mathbb{S}_K^d$ | $> 0$ | $\mathbf{x} \in \mathbb{R}^{d+1} : \langle \mathbf{x}, \mathbf{x} \rangle_2 = 1/K$ |
| $\mathbb{E}^d$ | 0 | $\mathbb{R}^d$ |
| $\mathbb{H}_K^d$ | $< 0$ | $\mathbf{x} \in \mathbb{R}^{d+1} : \langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = 1/K$ |

### 2.3 Problem Formulation

Let $\mathcal{G} = (\mathcal{E}, \mathcal{R})$ be a KG with multiple relations, where $\mathcal{E}$ and $\mathcal{R}$ represents the set of entities (nodes) and relations (edges), respectively. A triple $(e_h, r, e_t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is represented as an edge $r$ between head entity $e_h$ and tail entity $e_t$ in $\mathcal{G}$. The objective is to project entities $e \in \mathcal{E}$ onto entity embeddings $\mathbf{e} \in \mathcal{U}^d$ and relations $r$ onto relation embeddings $\mathbf{r} \in \mathcal{U}^d$, where the intrinsic heterogeneous structures of KG can be captured. Differently from previous work that defines the embedding space in constant-curvature space, such as Euclidean space $\mathbb{R}^d$ or hyperbolic space $\mathbb{H}_K^d$, we define the embedding space in a mixed-curvature space $\mathbb{P}_K^d$, such that $\mathcal{U}^d = \mathbb{P}_K^d$. In particular, the learned KG embeddings are used to predict the target entity of a given query of head entity and relation, $q := (e_h, r, ?)$ such that the predicted tuple does not exist in $\mathcal{G}$. We also learn a scoring function $\phi : \mathcal{E} \times \mathcal{R} \to \mathcal{R}$, which assigns a score $s = \phi(e_h, r, e_t)$ to each triple, indicating the probability with which the prediction is a true fact.

## 3 MIXED-CURVATURE MULTI-RELATIONAL GRAPH NEURAL NETWORK

In this section, we propose the mixed-curvature multi-relational graph neural network (M$^2$GNN). We start by introducing the design of the constant-curvature model. Then we describe how to construct the mixed-curvature model. Next we present the graph neural updater module. In the end, we provide the details of the training and optimization of the proposed method.

### 3.1 Constant-curvature Models

As the the mixed-curvature model is constructed using constant-curvature models, we first present the design of our **constant-curvature models**. The translational distance model is a simple and effective embedding approach for KG data modeling [24, 52], which maps the entities and relations to the latent semantic space and measures the distance between the relation-translated head entity and the tail entity. In this work, we construct our constant-curvature model in the form of translational distance as [1]. In particular, given a triple $(e_h, r, e_t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, we define the scoring function as follows:

$$\phi(e_h, r, e_t) = -d^{(r)}(\mathbf{e}_h, \mathbf{e}_t)^2 + b_h + b_t \tag{1}$$

**Table 2: Summary of the operations in hyperbolic space $\mathbb{H}_K$ / spherical space $\mathbb{S}_K$.**

| Operations | Symbol | hyperbolic space $\mathbb{H}_K$ / spherical space $\mathbb{S}_K$ |
|---|---|---|
| Exponential map | $\exp_0^K(\cdot)$ | $\exp_0^K(\mathbf{v}) = \tanh(\sqrt{K}\|\|\mathbf{v}\|\|)\frac{\mathbf{v}}{\sqrt{K}\|\|\mathbf{v}\|\|}$ |
| Logarithmic map | $\log_0^K(\cdot)$ | $\log_0^K(\mathbf{x}) = \tanh^{-1}(\sqrt{K}\|\|\mathbf{x}\|\|)\frac{\mathbf{x}}{\sqrt{K}\|\|\mathbf{x}\|\|}$ |
| Addition | $\oplus_K$ | $\mathbf{x} \oplus_K \mathbf{y} = \frac{(1+2K\mathbf{x}^T\mathbf{y}+K\|\|\mathbf{y}\|\|^2)\mathbf{x}+(1-K\|\|\mathbf{x}\|\|^2)\mathbf{y}}{1+2K\mathbf{x}^T\mathbf{y}+K^2\|\|\mathbf{x}\|\|^2\|\|\mathbf{y}\|\|^2}$ |
| Scalar Multiplication | $\otimes_K$ | $r \otimes_K \mathbf{x} = \exp_0^K(r\log_0^K(\mathbf{x}))$ |
| Matrix Multiplication | $\otimes_K$ | $\mathbf{M} \otimes_K \mathbf{x} = \exp_0^K(\mathbf{M}\log_0^K(\mathbf{x}))$ |
| Applying Function<br>- linear transform<br>- activation function<br>- element-wise multiplication | $f_K(\cdot)$ | $f_K(\mathbf{x}) = \exp_0^K(f(\log_0^K(\mathbf{x})))$ |
| Distance function | $d_{(\mathcal{M}_K^d)}(\cdot,\cdot)$ | $d_{(\mathcal{M}_K^d)}(\mathbf{x},\mathbf{y}) = \frac{2}{\sqrt{K}}\tanh^{-1}(\sqrt{K}\|\|-\mathbf{x}\oplus_K\mathbf{y}\|\|)$ |

where $\mathbf{e}_h, \mathbf{e}_t \in \mathcal{U}^d$ are embeddings and $b_h, b_t \in \mathbb{R}$ are scalar biases of the head and tail entities $e_h$ and $e_t$, respectively. $d^{(r)}(\cdot,\cdot)$ denotes a distance function between a head entity and a entity over the relation $r$, which is the $\ell_2$ distance. First we define the constant-curvature model in Euclidean space with zero-curvature. The score function can be expressed as follows:

$$\begin{aligned}\phi_{\mathbb{E}^d}(e_h, r, e_t) &= -d^{(r)}_{\mathbb{E}^d}(\mathbf{o}_h, \mathbf{o}_t)^2 + b_h + b_t \\ &= -d^{(r)}_{\mathbb{E}^d}(\mathbf{R}\mathbf{o}_h, \mathbf{o}_t + \mathbf{r}_o)^2 + b_h + b_t\end{aligned} \tag{2}$$

where $\mathbf{o}_h, \mathbf{o}_t \in \mathbb{E}^d$ are Euclidean embeddings of the head and tail entities $e_h$ and $e_t$, respectively. $\mathbf{R} \in \mathbb{R}^{d\times d}$ represents a diagonal relation matrix and $\mathbf{r}_o \in \mathbb{E}^d$ denotes a relation embedding of relation $r$. After applying a stretch $\mathbf{R}$ and a translational relation $\mathbf{r}_o$, we can obtain the new representations of the head entity $\mathbf{o}_h = \mathbf{R}\mathbf{o}_h$ and tail entity $\mathbf{o}_t = \mathbf{o}_t + \mathbf{r}_o$. However, the above model is in Euclidean space, which has limited representation ability and cannot capture complex structures, such as hierarchical and cyclic structures.

To address this issue, we propose the second constant-curvature model, which maps the entities and relations onto a hyperbolic space. Taking the hyperbolic analogue of Equation 2, the scoring function can be rewritten as follows:

$$\begin{aligned}\phi_{\mathbb{H}^d_K}(e_h, r, e_t) &= -d^{(r)}_{\mathbb{H}^d_K}(\mathbf{h}_h, \mathbf{h}_t)^2 + b_h + b_t \\ &= -d^{(r)}_{\mathbb{H}^d_K}\Big(\exp_0^K\left(\mathbf{R}\log_0^K(\mathbf{h}_h)\right), \mathbf{h}_t \oplus_K \mathbf{r}_h\Big)^2 + b_h + b_t\end{aligned} \tag{3}$$

where $\mathbf{h}_h, \mathbf{h}_t \in \mathbb{H}^d_K$ are hyperbolic embeddings of the head and tail entities $e_h$ and $e_t$, respectively, and $\mathbf{r}_h \in \mathbb{H}^d_K$ represents a relation embedding of relation $r$ in hyperbolic space. The resultant head entity embedding $\mathbf{h}_h \in \mathbb{H}^d_K$ is computed by performing the matrix-vector multiplication defined in Table 2, such that the original entity $\mathbf{h}_h \in \mathbb{H}^d_K$ is projected to the tangent space of the Poincaré ball at $o$ with $\log_o^K(\cdot)$, transformed by the diagonal relational matrix $\mathbf{R} \in \mathbb{R}^{d\times d}$, and then mapped back to the Poincaré ball by $\exp_o^K(\cdot)$. The new tail entity embedding $\mathbf{h}_t \in \mathbb{H}^d_K$ is computed by adding the relation embedding $\mathbf{r}_h \in \mathbb{H}^d_K$ to the tail entity embedding $\mathbf{h}_t \in \mathbb{H}^d_K$. As the embedding space is defined in hyperbolic space, $K$ is a negative constant value.

With the help of the hyperbolic model, hierarchical structures can be captured. Because there are also rich cyclic structures in the KG, the KG embedding also needs to capture cyclic data. To fill this gap, we first attempt to design a constant-curvature model, which maps the entities and relations onto spherical space. Similar to the hyperbolic analogue of Equation 2, we define the scoring function in spherical space as follows:

$$\begin{aligned}\phi_{\mathbb{S}^d_K}(e_h, r, e_t) &= -d^{(r)}_{\mathbb{S}^d_K}(\mathbf{s}_h, \mathbf{s}_t)^2 + b_h + b_t \\ &= -d^{(r)}_{\mathbb{S}^d_K}\Big(\exp_0^K\left(\mathbf{R}\log_0^K(\mathbf{s}_h)\right), \mathbf{s}_t \oplus_K \mathbf{r}_s\Big)^2 + b_h + b_t\end{aligned} \tag{4}$$

where $\mathbf{s}_h, \mathbf{s}_t \in \mathbb{S}^d_K$ are spherical embeddings of the head and tail entities $e_h$ and $e_t$, respectively, and $\mathbf{r}_s \in \mathbb{S}^d_K$ is a spherical relation embedding of relation $r$. The new head entity embedding $\mathbf{s}_h \in \mathbb{S}^d_K$ is computed by performing the matrix-vector multiplication defined in Table 2. The new tail entity embedding $\mathbf{s}_t \in \mathbb{S}^d_K$ is computed by adding the relation embedding $\mathbf{r}_s \in \mathbb{S}^d_K$ to the tail entity embedding $\mathbf{s}_t \in \mathbb{S}^d_K$. As the embedding space is defined in spherical space, $K$ is a positive constant value.

## 3.2 Mixed-curvature Model

Constant-curvature models can benefit from their specific bias to better fit certain structure types. However, the real-world KG is usually not structured uniformly and demonstrates intrinsic heterogeneous structures. To address this problem, we propose the **mixed-curvature model**. In particular, we design and construct the mixed-curvature space and project the KG entities and relations onto it, in order to provide a space of heterogeneous curvature that can capture the intrinsic heterogeneous structures in KGs.

We construct the mixed-curvature spaces by leveraging the product of constant-curvature spaces. Specifically, our mixed-curvature space is constructed by performing the Cartesian product of several component constant-curvature spaces, such that $\mathbb{P} = \times_{i=1}^k \mathcal{M}_i^{d_i}$, where $\times$ denotes the Cartesian product and $k$ denotes the number of components. Here, each component space $\mathcal{M}_i^{d_i} \in \{\mathbb{E}, \mathbb{H}, \mathbb{S}\}$ is a $d_i$−dimensional constant-curvature space, with curvature $K_i$. By

fusing all the components, the curvature of the new space demonstrates the non-constant curved property. The distance function in the constructed mixed-curvature space can be obtained by performing decomposition as follows:

$$d^{\ell}_{\mathbb{P}}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{k} d^{\ell}_{\mathcal{M}_i^{d_i}}(\mathbf{x}^{(i)},\mathbf{y}^{(i)}) \tag{5}$$

where $\mathbf{x}^{(i)}$ denotes a embedding in $\mathcal{M}_i^{d_i}$ and $\ell$ represents the norm indicator.

This distance function enables us to introduce combinatorial constructions, which provide simple and interpretable embedding spaces. Since all operations defined on our manifolds are element-wise, we can decompose it into parts $\mathbf{x}^{(i)}$ and apply the operation $f_i^{d_i}(\mathbf{x}^{(i)})$. Then the result of each part is $\tilde{x}^{(i)}$. The final result is the combination of the resulting parts $\tilde{x} = \odot_{i=1}^{k}\tilde{x}^{(i)}$, where $\odot$ denotes the combining operation for the individual parts.

Our mixed-curvature space is described by the signature, a parameterization, including several degrees of freedom per component: the space type $\mathcal{M}_i$, the dimensionality $d_i$, and the curvature $K_i$. It is required to select all three parameters for every component in the product space. For $d_1, ..., d_k \in \mathbb{Z}$, such that $\sum_{i=1}^{k} d_i = d \in \mathbb{Z}$, the Cartesian product of Euclidean space is $\mathbb{E}^d = \times_{i=1}^{k}\mathbb{E}^{d_i}$. Since the product space can be decomposed into three types of spaces (spherical, hyperbolic and Euclidean), all vector-like operations can be conducted in the corresponding space component.

After we define and construct the mixed-curvature space, we design the scoring function for KG entities and relations in the mixed-curvature product space as follows:

$$\begin{aligned}
\phi_{\mathbb{P}^{\{d_o,d_h,d_s\}}_{\{K_o,K_h,K_s\}}}(e_h,r,e_t) &= -d^{(r)}_{\mathbb{P}^{\{d_o,d_h,d_s\}}_{\{K_o,K_h,K_s\}}}(\mathbf{p}_h,\mathbf{p}_t)^2 + b_h + b_t \\
&= -\big(d^{(r)}_{\mathbb{H}^{d_h}_{K_h}}(\mathbf{h}_h,\mathbf{h}_t)^2 + d^{(r)}_{\mathbb{S}^{d_s}_{K_s}}(\mathbf{s}_h,\mathbf{s}_t)^2 \\
&\quad + d^{(r)}_{\mathbb{E}^{d_o}}(\mathbf{o}_h,\mathbf{o}_t)^2\big) + b_h + b_t \\
&= -\Bigg(d^{(r)}_{\mathbb{H}^{d_h}_{K_h}}\Big(\exp_0^{K_h}\big(\mathbf{R}\log_0^{K_h}(\mathbf{h}_h)\big), \mathbf{h}_t \oplus_{K_h} \mathbf{r}_h\Big)^2 \\
&\quad + d^{(r)}_{\mathbb{S}^{d_s}_{K_s}}\Big(\exp_0^{K_s}\big(\mathbf{R}\log_0^{K_s}(\mathbf{s}_h)\big), \mathbf{s}_t \oplus_{K_s} \mathbf{r}_s\Big)^2 \\
&\quad + d^{(r)}_{\mathbb{E}^{d_o}}(\mathbf{R}\mathbf{o}_h, \mathbf{o}_t + \mathbf{r}_o)^2\Bigg) + b_h + b_t
\end{aligned} \tag{6}$$

where $\mathbf{p}_h, \mathbf{p}_t \in \mathbb{P}^{\{d_o,d_h,d_s\}}_{\{K_o,K_h,K_s\}}$ are mixed-curvature space embeddings of the head and tail entities $e_h$ and $e_t$, respectively. $d_o, d_h, d_s$ are the dimensions of the component spaces of Euclidean, hyperbolic, and spherical. $K_h, K_s$ are the curvatures of the component spaces of the hyperbolic and spherical spaces. As the curvature of Euclidean is 0, $K_o = 0$. Though performing optimization in the mixed-curvature space is challenging, our objective is a distance function. Based on the decomposable property of the mixed-curvature space, we can follow Equation 5 and decompose the distance function $d_{\mathbb{P}^{\{d_o,d_h,d_s\}}_{\{K_o,K_h,K_s\}}}$ to be the sum of the distance functions of each component space. Then, we can just optimize the scoring function as in constant-curvature space.

## 3.3 Graph Neural Updater

The translational distance model learns embeddings with simple operations and limited parameters. Recent work [2, 34, 40] applies the graph neural network (GNN) to KG completion and demonstrates the improved performance by leveraging the local neighborhood of a triple. Non-GNN models cannot capture the complex context information and are restricted on the quality of the learned embedding [24, 52]. As the developed mixed-curvature model is designed in the translational distance manner, it suffers from the same limitation as other translational distance models. To overcome this problem, we propose the **graph neural updater**, a graph-neural-network-based embedding updating module to aggregate the entity and relation embedding features in the neighborhood of each entity and update the embedding representation of each entity and relation. Formally, we define our graph neural updater as follows:[1]

$$\mathbf{e}'_i = \mathrm{GNN}(\mathbf{e}_j, \mathbf{r}_k | j \in \mathcal{N}_i, k \in \mathcal{R}_{ij}) \tag{7}$$

where $\mathbf{e}'_i$ denotes the updated embedding of the target entity $e_i$, $\mathbf{e}_j$ denotes the embedding of the tail entity $e_j$ connecting to the target entity $e_i$ with the embedding $\mathbf{e}_i$ corresponding to the relation $r_k$, $\mathcal{N}_i$ denotes the neighborhood of entity $e_i$ and $\mathcal{R}_{ij}$ denotes the relations connecting entities $e_i$ and $e_j$. As KG is a multi-relational graph, both the entity and relation embeddings lie in the same embedding space. The relational context of an entity should include both entity neighbor and relation neighbor. We define a message embedding $\mathbf{m}_{ijk}$ to capture the relational context. Following [34], it is computed by concatenating entity and relation embeddings for a specific triple $(e_i, r_k, e_j)$ followed by a linear transformation. Formally, it is defined as follows:

$$\mathbf{m}_{ijk} = \mathbf{W}_1 \otimes_K [\mathbf{e}_i || \mathbf{e}_j || \mathbf{r}_k] \tag{8}$$

where $\mathbf{W}_1$ denotes the linear transformation matrix and $\otimes_c$ indicates the matrix multiplication in the constant-curvature space. As the importance of the neighborhood is usually quite different [49], treating each relational context equally is not appropriate. Therefore, we introduce the importance of each relational context and learn the importance of each message embedding. In particular, we first apply the linear transformation with a matrix $\mathbf{W}_2$ and then employ a LeakyReLU activation function. The message importance can be computed as follows:

$$b_{ijk} = \exp_0^K(\mathrm{LeakyReLU}(\log_0^K(\mathbf{W}_2 \otimes_K \mathbf{m}_{ijk})) \tag{9}$$

Next we compute the attention value of each message embedding by employing the softmax function over $b_{ijk}$ as follows:

$$\alpha_{ijk} = \mathrm{softmax}_{jk}(b_{ijk}) = \frac{\exp(b_{ijk})}{\sum_{v\in\mathcal{N}_i}\sum_{q\in\mathcal{R}_{iv}}\exp(b_{ivq})} \tag{10}$$

Then we compute the updated entity embedding. As suggested in [49], using a multi-head mechanism. Thus the updated entity embedding with $N$-head attention is computed as follows:

$$\mathbf{e}_i = \|_{n=1}^{N}\ \sigma_K\left(\sum_{j\in\mathcal{N}_i}\sum_{k\in\mathcal{R}_{ij}} \alpha^n_{ijk} \otimes_K \mathbf{m}^n_{ijk}\right) \tag{11}$$

[1] In this subsection, we describe the graph neural updater in the constant-curvature space and remove the subscripts of space indicator for all embedding related symbols for simplicity.

To obtain the updated embedding of the relation, a linear transformation is employed:

$$\mathbf{r}'_k = \mathbf{W}_{rel'} \otimes_K \mathbf{r}_k \tag{12}$$

where $\mathbf{W}_{rel'}$ denotes the parameterized linear transformation matrix for relation embedding mapping. In particular, we perform two iterations of embedding update to capture two-hop relational context. In the second iteration, we obtain the updated entity embedding $\mathbf{e}''_i$ by averaging (instead of concatenating) the embeddings from multiple heads by:

$$\mathbf{e}''_i = \sigma_K\left(\frac{1}{N}\sum_{n=1}^{N}\sum_{j\in\mathcal{N}_i}\sum_{k\in\mathcal{R}_{ij}} \alpha^n_{ijk} \otimes_K \mathbf{m}'^n_{ijk}\right) \tag{13}$$

To get the final updated embedding $\mathbf{e}_i^{final}$ while keeping the initial embedding information, we further employ a linear transformation layer to fuse the initial embedding with the new embedding:

$$\mathbf{e}_i^{final} = (\mathbf{W}_3 \otimes_K \mathbf{e}_i) \oplus_K \mathbf{e}''_i \tag{14}$$

where $\mathbf{W}_3$ is a weight matrix for fusion of original and new entity embeddings. With the help of the graph neural updater, we can selectively gather the multi-hop context information to capture the heterogeneous structures in the mixed-curvature space.

## 3.4 Training and Optimization

To train our proposed M$^2$GNN model, *Ne* negative samples are constructed for each triple $(e_h, r, e_t)$ by performing corruption on either the head $(e_h, r, e'_t)$ or the tail $(e_t, r^{-1}, e'_h)$ entity with a randomly chosen new entity. Our objective is to minimize the Bernoulli negative log-likelihood loss as follows:

$$\mathcal{L}(y,p) = -\frac{1}{N}\sum_{i=1}^{N}(y^{(i)})\log(p^{(i)}) + (1-y^{(i)})log(1-p^{(i)}) \tag{15}$$

where $p$ denotes the predicted probability, $y$ denotes the binary label indicating whether a sample fact is available or not and $N$ is the number of training samples. To optimize the proposed model, we define all parameters in the tangent space at the origin [7].

As discussed in the previous subsection, the mixed-curvature space is parameterized by the signature (space type, curvature, and dimensionality). For the space type, we use the combination of three constant-curvature space components (hyperbolic space, spherical space and Euclidean space) to cover the intrinsic heterogeneous structures of the KGs. For the curvatures of each constant-curvature space component ($K_h$ for hyperbolic space and $K_s$ for spherical space), they are typically fixed and need to be manually defined through domain knowledge and data analysis. Improperly defined curvatures cannot capture the intrinsic heterogeneous structures of KGs accurately. To address this problem, we propose a trainable curvature for each constant-curvature space component. As our loss function is differentiable with respect to these curvatures, we treat these curvatures as parameters of the model and learn them using gradient based optimization. For the dimensionality, we set the dimension of each constant-curvature space component to be the same and propose space weights to balance each constant-curvature space component. In particular, if the hyperbolic space weight is $\lambda \in \{0,1\}$, the spherical space weight is $\mu \in \{0,1\}$, and the Euclidean space weight is $(1-\lambda-\mu) \in \{0,1\}$, that is, $\mu+\lambda+(1-\lambda-\mu)=1$, then a trade-off is established: as we increase (resp. decrease) one of the three weights, the other two will decrease (resp. increase). However, the space weight setting also requires prior knowledge or human-engineering. We address this problem by proposing the trainable space weights and search the proper space weight in a data-driven way. Then Equation 6 can be rewritten as follows:

$$\begin{aligned}
\phi_{\mathbb{P}^{\{d_o,d_h,d_s\}}_{\{K_o,K_h,K_s\}}}(e_h,r,e_t) &= -\big(\lambda d^{(r)}_{\mathbb{H}^{d_h}_{K_h}}(\mathbf{h}_h,\mathbf{h}_t)^2 + \mu d^{(r)}_{\mathbb{S}^{d_s}_{K_s}}(\mathbf{s}_h,\mathbf{s}_t)^2 \\
&\quad + (1-\lambda-\mu) d^{(r)}_{\mathbb{E}^{d_o}}(\mathbf{o}_h,\mathbf{o}_t)^2\big) + b_h + b_t \\
&= -\Bigg(\lambda d^{(r)}_{\mathbb{H}^{d_h}_{K_h}}\Big(\exp_0^{K_h}\big(\mathbf{R}\log_0^{K_h}(\mathbf{h}_h)\big), \mathbf{h}_t \oplus_{K_h} \mathbf{r}_h\Big)^2 \\
&\quad + \mu d^{(r)}_{\mathbb{S}^{d_s}_{K_s}}\Big(\exp_0^{K_s}\big(\mathbf{R}\log_0^{K_s}(\mathbf{s}_h)\big), \mathbf{s}_t \oplus_{K_s} \mathbf{r}_s\Big)^2 \\
&\quad + (1-\lambda-\mu) d^{(r)}_{\mathbb{E}^{d_o}}(\mathbf{R}\mathbf{o}_h, \mathbf{o}_t + \mathbf{r}_o)^2\Bigg) + b_h + b_t
\end{aligned} \tag{16}$$

# 4 EXPERIMENTS AND RESULTS

In this section, we evaluate the proposed M$^2$GNN model, and present its performance on three KG datasets. We first introduce the experimental setup. Next, we show the effectiveness of the proposed M$^2$GNN model. We further conduct several ablation studies to demonstrate the effectiveness of each proposed module.

## 4.1 Experimental Setup

*4.1.1 Datasets.* We evaluate our proposed models on the KG completion task using WN18RR [45], FB15k-237 [12] and YAGO3-10 [32], in order to cover different structures and scales. We summarize the data statistics in Table 3.

*4.1.2 Baselines.* We compare the proposed method to state-of-the-art single-curvature embedding based KG completion methods defined in different spaces, as follows:

- **RESCAL** [38]: Euclidean embedding models with each relation as a full rank matrix.
- **TransE** [5]: First translational distance Euclidean embedding.
- **DisMult** [5]: Euclidean embedding models with a diagonal relational matrix.
- **MuRE** [1]: Translational distance Euclidean embedding with a diagonal relational matrix.
- **ComplEx** [46]: Extension of DisMult in a complex space.
- **RotatE** [43]: Extension of TransE in a complex space with modulus part and phase part.
- **Conve** [12]: NN-based method with score function defined by a convolutional neural network.
- **CompGCN** [47]: NN-based method with score function defined by a graph convolutional network.
- **A2N** [2]: NN-based method with score function defined by a graph attentional network.
- **MuRP** [1]: Translational distance hyperbolic embedding with a diagonal relational matrix.

**Table 3: Dataset statistics.**

| Dataset | #Entity | #Relation | #Train | #Valid | #Test | Structure Heterogeneity | Scale |
|---|---|---|---|---|---|---|---|
| WN18RR | 40,943 | 11 | 86,835 | 3,034 | 3,134 | Low | Small |
| FB15-237 | 14,541 | 237 | 272,115 | 17,535 | 20,466 | High | Medium |
| YAGO3-10 | 123,182 | 37 | 1,079,040 | 5000 | 5000 | Low | Large |

Baseline results in Subsection 4.2 are taken from the original papers. The rest of the baseline results are obtained by open-source implementations of each model, where we perform hyper-parameter search over the same parameters suggested by the original papers.

*4.1.3 Ablations.* To analyze the advantage of mixed-curvature geometry, we consider several variants of the proposed method without the GNN module, including:

- **MuRS** – multi-relational spherical embedding.
- **MuRMP** – multi-relational mixed-curvature space embedding with fixed curvature [1,-1].
- **MuRMP-autoK** – multi-relational mixed-curvature space embedding with learnable curvature.
- **MuRMP-autoT** – multi-relational mixed-curvature space embedding with learnable space weight.
- **MuRMP-autoKT** – multi-relational mixed-curvature space embedding with learnable curvature and space weight.

Also, we consider the following variants to analyze the advantage of GNN:

- **M$^2$noGNN** – mixed-curvature space embedding without the GNN updater.
- **H$^2$GCN** – mixed-curvature space embedding with the GCN updater.
- **H$^2$-khead** – mixed-curvature space embedding with a k-head attentional GNN updater.

*4.1.4 Implementation Details.* Following previous KG completion work [5], we use MRR and Hits@K as evaluation metrics. We perform the optimization in tangent space [7] and use standard Euclidean optimizers. We implement the proposed method in PyTorch and conduct the experiments on NVIDIA Tesla V100 GPU. For our proposed model, we conduct a hyperparameter search on dimensionality, learning rate, optimizer, negative sample size, batch size and number of attention heads. We report the best hyperparameters (dimensionality, learning rate, optimizer, negative sample size, batch size and number of attention heads) for each dataset as follows: {WN18RR: 200, 0.001, Adam, 500, 500, 4}, {FB15k-237: 200, 0.05, Adagrad, 500, 500, 4}, {YAGO3-10: 500, 0.005, Adam, 250, 500, 4}.

## 4.2 Overall Results

In this subsection, we compare the proposed method with existing state-of-the-art methods and some ablations with different geometry space. The experimental results are shown in Table 4. As we can see, the proposed M$^2$GNN outperforms all the baselines on all three datasets with various typed structures, which verifies the effectiveness of the proposed methods in capturing intrinsic heterogeneous structures in KG. We also observe that the mixed-curvature methods outperform the single-curvature methods, while the performance of single-curvature methods varies depending on the datasets. Compared with mixed-curvature method with fixed curvatures or fixed space weights, learnable curvatures and space weights improve the performance significantly. Another interesting fact is that GNN-based methods achieve best performance on all the datasets, which shows the benefit of relational context.

## 4.3 Ablations on Graph Neural Network

In this subsection, we study the effectiveness of the graph neural updater module. The ablation results on the WN18RR and FB15k-237 datasets are shown in Table 5. M$^2$noGNN denotes the results without the graph neural updater. M$^2$GCN denotes the results with the graph convolutional updater. M$^2$GAT-*k*head denotes the results with the proposed graph neural updater with $k$ heads. We can observe that the GNN model outperforms the non-GNN model. The proposed graph neural updater with one head has similar performance to the graph convolutional variants. When the number of heads increases, the performance also improves.

## 4.4 Ablations on Dimensionality

In this subsection, we investigate the role of dimensionality. We conduct experiments on WN18RR and report the MRR of mixed-curvature model (MuRMP-autoKT, MuRMP) against single constant-curvature space methods (MuRS, MuRP, and MuRE) and dimension $d \in \{10, 15, 20, 40, 100, 200, 400\}$. Fig. 2 shows the results, which are obtained by averaging over 10 runs. As expected, MuRMP-autoKT achieves the best performance across a broad range of dimensions. Its variant MuRMP is less stable, since it uses fixed curvature values and space weights and cannot capture the intrinsic heterogeneous structures very well. As WNRR18 has rich hierarchical structures, both the proposed mixed-curvature models and MuRP achieve good performance when the dimensionality is low.

## 4.5 Ablations on Relation Type

In this subsection, we investigate how the performance of the proposed method is affected by relation types on WN18RR. We report a number of metrics to describe each relation, including global graph curvature ($\xi_G$) [17] and Krackhardt hierarchy score (Khs) [27]. These two metrics are used to justify if the given data have a rich hierarchical structure. Specifically, we compare averaged hits@10 over 10 runs for each relations of MuRMP-autoKT, MuRMP, MuRS, MuRP and MuRE for entity embeddings of low dimensionality (d = 20). From Table 6 we can see that the proposed mixed-curvature model MuRMP-autoKT outperforms its variant with fixed-curvature and fixed space weights. Besides, it also outperforms all the single constant-curvature methods. The experiments verify the effectiveness of the proposed method in dealing with heterogeneous types of data.

**Table 4: KG completion results for embeddings on WN18RR, FB15k-237, and YAGO3-10. The best results are in bold and the second best results are underlined.**

| $\mathcal{M}$ | Model | WN18RR | | | | FB15k-237 | | | | YAGO3-10 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@1 | H@10 |
| $\mathbb{R}$ | RESCAL | .420 | - | - | .447 | .270 | - | - | | - | - | - | - |
| $\mathbb{R}$ | TransE | .226 | - | - | .501 | .294 | - | - | .465 | - | - | - | - |
| $\mathbb{R}$ | DisMult | .430 | .390 | .440 | .490 | .241 | .155 | .263 | .419 | .340 | .240 | .380 | .540 |
| $\mathbb{R}$ | ConvE | .430 | .400 | .440 | .520 | .325 | .237 | .356 | .501 | .440 | .350 | .490 | .620 |
| $\mathbb{R}$ | MuRE | .465 | .436 | .487 | .554 | .336 | .245 | .370 | .521 | .532 | .444 | .584 | .694 |
| $\mathbb{R}$ | CompGCN | .479 | .443 | .494 | .546 | .355 | .264 | .390 | .535 | .489 | .395 | .500 | .582 |
| $\mathbb{R}$ | A2N | .430 | .410 | .440 | .510 | .317 | .232 | .348 | .486 | .445 | .349 | .482 | .501 |
| $\mathbb{C}$ | ComplEx | .440 | .410 | .460 | .510 | .247 | .158 | .275 | .428 | .360 | .260 | .400 | .550 |
| $\mathbb{C}$ | RotatE | .476 | .428 | .492 | .571 | .338 | .241 | .375 | .533 | .495 | .402 | .550 | .670 |
| $\mathbb{H}$ | MuRP | .481 | .440 | .495 | .566 | .335 | .243 | .367 | .518 | .354 | .249 | .400 | .567 |
| $\mathbb{S}$ | MuRS | .454 | .432 | .482 | .550 | .338 | .249 | .373 | .525 | .351 | .244 | .382 | .562 |
| $\mathbb{P}$ | MuRMP | .473 | .435 | .485 | .552 | .345 | .258 | .385 | .542 | .358 | .248 | .389 | .566 |
| $\mathbb{P}$ | MuRMP-autoK | .476 | .435 | .488 | .556 | .352 | .261 | .388 | .544 | .361 | .252 | .394 | .569 |
| $\mathbb{P}$ | MuRMP-autoT | .479 | .438 | .489 | .559 | .357 | .267 | .393 | .550 | .471 | .385 | .503 | .609 |
| $\mathbb{P}$ | MuRMP-autoKT | .481 | .441 | .496 | .569 | .358 | .273 | .394 | .561 | .495 | .448 | .591 | .698 |
| $\mathbb{P}$ | $M^2$GNN | **.485** | **.444** | **.498** | **.572** | **.362** | **.275** | **.398** | **.565** | **.543** | **.478** | **.605** | **.702** |

**Table 5: Ablation results on the WN18RR and FB15k-237 datasets. We compare variants of the proposed model with various graph neural network settings.**

| Model | WN18RR | | | | FB15k-237 | | | |
|---|---|---|---|---|---|---|---|---|
| | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 |
| CompGCN | .479 | .443 | .494 | .546 | .355 | .264 | .390 | .535 |
| A2N | .430 | .410 | .440 | .510 | .317 | .232 | .348 | .486 |
| $M^2$noGNN | .481 | .441 | .496 | .571 | .358 | .273 | .394 | .561 |
| $M^2$GCN | .482 | .442 | .496 | .570 | .360 | .274 | .396 | .563 |
| $M^2$GAT-1head | .479 | .442 | .494 | .565 | .359 | .273 | .396 | .562 |
| $M^2$GAT-4head | **.485** | **.444** | **.498** | **.572** | **.362** | **.275** | **.398** | **.565** |

**Table 6: Comparison of hits@10 for WN18RR with $d = 20$. Average computed over 10 runs.**

| WN relation name | $\xi_G$ | Khs | MuRE | MuRP | MuRS | MuRMP | MuRMP-autoKT |
|---|---|---|---|---|---|---|---|
| also_see | -2.09 | .24 | .634 | .705 | .483 | .692 | **.725** |
| hypernym | -2.46 | .99 | .161 | .228 | .126 | .222 | **.232** |
| has_part | -1.43 | 1 | .215 | .282 | .301 | .134 | **.316** |
| member_meronym | -2.90 | 1 | .272 | .346 | .138 | .343 | **.350** |
| synset_domain_topic_of | -0.69 | .99 | .316 | .430 | .163 | .421 | **.445** |
| instance_hypernum | -0.82 | 1 | .488 | .471 | .258 | .345 | **.491** |
| member_of_domain_region | -0.78 | 1 | .308 | .347 | .201 | .344 | **.349** |
| member_of_domain_usage | -0.74 | 1 | .396 | .417 | .228 | .416 | **.420** |
| derivationally_related _form | -3.84 | .04 | .954 | .967 | .965 | .967 | **.970** |
| similar_to | -1.00 | 0 | **1** | **1** | **1** | **1** | **1** |
| verb_group | -0.5 | 0 | .974 | .974 | .976 | .976 | **.981** |

## 4.6 Ablations on Curvatures and Space Weights

It is important to set the curvature of the constant-curvature space and the space weights correctly. These parameter sets provide flexibility for the model to capture the intrinsic heterogeneous structures in KG data. Specifically, we report the learned curvatures and space weights for each dataset in Table 7. The results show that for datasets with rich hierarchical structures (WN18RR, YAGO3-10), the hyperbolic curvature values as compared with those of the other set (FB15k-237) are about 30% smaller and the hyperbolic space weights are approximately double. The spherical space weight value for set FB15k-23 is larger than those of the other two sets.

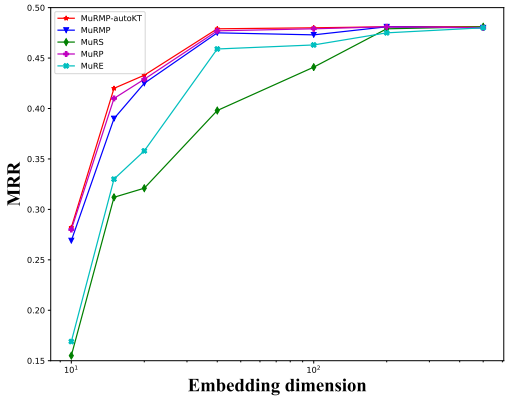



**Figure 2: MRR as a function of the embedding dimension with $d \in \{10, 15, 20, 40, 100, 200, 500\}$ on the WN18RR dataset. Average computed over 10 runs.**

**Table 7: Comparison of parameters—curvature value and space weight—for hyperbolic and spherical space—in three datasets.**

| Parameters | WN18RR | FB15k-237 | YAGO3-10 |
|---|---|---|---|
| Hyperbolic Curvature | -1.263 | -0.351 | -1.180 |
| Spherical Curvature | 0.184 | 1.112 | 0.215 |
| Hyperbolic space weight | 0.623 | 0.324 | 0.587 |
| Spherical space weight | 0.198 | 0.338 | 0.275 |

# 5 RELATED WORK

Our work is related to knowledge graph completion and non-Euclidean embedding. We briefly discuss them in this section.

## 5.1 Knowledge Graph Completion

*5.1.1 Euclidean Embedding Models.* Euclidean embedding models for KG completion have been extensively studied. They typically assign an embedding vector to each entity and relation in the Euclidean embedding space and train the embeddings based on observed facts. Examples include the following Euclidean translational models: TransE [5], TransD [23], TransH [56], and TransR [30]. There are also Euclidean bilinear models, such as RESCAL [38] and DistMult [60]. More recently, extensions of Euclidean embeddings in complex space have been proposed, such as ComplEx [46], RotatE [43], HAKGE [18], and HAKE [62]. These methods only consider Euclidean distance between the translated head entity embedding and tail entity embedding, which cannot capture the intrinsic heterogeneity of KG structures.

*5.1.2 Neural Network Models.* Neural network models have attracted considerable research interest in recent years. They map entities and relations into the embedding space using a neural network in an end-to-end manner. Based on the type of neural network, we can categorize these methods in three groups: NN-based model [14, 42], which uses fully connected neural networks, CNN-based model [12, 12], which uses convolutional neural networks, and GNN-based model, which uses graph neural networks [2, 34, 40]. These methods can learn the KG embedding automatically in a data driven way, but they still assume the embedding space is Euclidean. Thus, they still fail to capture the intrinsic heterogeneity of KG structures.

*5.1.3 Hyperbolic Embedding Models.* Hyperbolic embedding models include MuRP [1], which develops the hyperbolic analogy of the translational distance model for KG embeddings for the completion task and AttH [6], which builds a rotation based model in hyperbolic space for KG embeddings [6]. Because these two methods only consider a specific negative-curvature space, they are not flexible to other type of structures and therefore fail to capture the intrinsic heterogeneity of KG structures.

## 5.2 Non-Euclidean Embedding

*5.2.1 Single-curvature Models.* Embedding data in non-Euclidean space (constant non-zero curvature space) has attracted considerable attention. [36] defined the WordNet non-embedding space in the Poincaré ball and shows the significant gain over Euclidean embeddings in a low-dimensional setting. [37] embeds hierarchical data in the Lorentz hyperbolic space. Some later work redefines and develops the existing algorithms in hyperbolic space, such as Poincaré GloVe [44], Hyperbolic Attention Networks [19] and Hyperbolic Graph Convolutional Neural Networks [7]. These works study the embedding in hyperbolic space with negative curvature. On the other hand, some researchers study the embedding in spherical space with positive curvature and show that a spherical embedding can better capture a cyclic structure, such as directional data. [9, 59] develop the Spherical Variational Autoencoders and apply them in language and document modeling. [33] proposes a spherical generative model and learns word and paragraph embeddings jointly. Among these approaches, only negative-curvature models are considered for the multi-relational KG completion task.

*5.2.2 Mixed-curvature Models.* Noticing that data can be non-uniformly structured, researchers have studied embeddings in mixed-curvature space. [17, 41] construct a mixed-curvature space by multiplying manifolds with different curvatures and show the flexibility in dealing with data with rich structures. Though they study non-Euclidean geometry for intrinsic heterogeneous structures, their focus is on unsupervised embeddings and they only consider homogeneous relation (with only one relation type), which cannot be applied directly to address the multi-relational KG embedding problem. Furthermore, they required manually defining the fixed curvatures using domain knowledge and additional data analysis, which is difficult to obtain.

# 6 CONCLUSION

We develop a novel Mixed-curvature Multi-relational Graph Neural Network (M$^2$GNN) for knowledge graph completion. The mixed-curvature space is constructed by a tractable Riemannian product manifold, which combines Euclidean, spherical, and hyperbolic spaces. Benefiting from mixed-curvature space modeling, our method improves multi-relational graph representations by better capturing the intrinsic heterogeneous structures in KGs. Furthermore, the proposed method can adaptively aggregate the relational

context information in the mixed-curvature space and improve the embedding quality.

## ACKNOWLEDGMENTS

Shen Wang and Philip S. Yu are supported in part by NSF grants III-1763325, III-1909323, and SaTC-1930941. Isabel F. Cruz is funded in part by NSF grant III-1618126 and by the Office of Naval Research. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research.

## REFERENCES

[1] Ivana Balažević, Carl Allen, and Timothy Hospedales. 2019. Multi-relational Poincaré graph embeddings. In *Advances in Neural Information Processing Systems*. 4465–4475.

[2] Trapit Bansal, Da-Cheng Juan, Sujith Ravi, and Andrew McCallum. 2019. A2N: Attending to Neighbors for Knowledge Graph Inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 4387–4392.

[3] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural language Processing*. 1533–1544.

[4] Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1415–1425.

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*. 2787–2795.

[6] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 6901–6914.

[7] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems*. 4868–4879.

[8] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247* (2018).

[9] Tim R Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M Tomczak. 2018. Hyperspherical variational auto-encoders. *arXiv preprint arXiv:1804.00891* (2018).

[10] Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. 2018. Representation tradeoffs for hyperbolic embeddings. *Proceedings of machine learning research* 80 (2018), 4460.

[11] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advancesin Neural Information Processing Systems*. 3844–3852.

[12] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[13] Dennis Diefenbach, Kamal Singh, and Pierre Maret. 2018. WDAqua-core1: A question answering service for RDF knowledge bases. In *Companion Proceedings of The Web Conference 2018*. 1087–1091.

[14] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge Vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 601–610.

[15] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212* (2017).

[16] Jibing Gong, Shen Wang, Jinlong Wang, Wenzheng Feng, Hao Peng, Jie Tang, and Philip S Yu. 2020. Attentional Graph Convolutional Networks for Knowledge Concept Recommendation in MOOCs in a Heterogeneous View. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 79–88.

[17] Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. 2018. Learning mixed-curvature representations in product spaces. In *International Conference on Learning Representations*.

[18] Yulong Gu, Yu Guan, and Paolo Missier. 2020. Efficient Rule Learning with Template Saturation for Knowledge Graph Completion. *arXiv preprint arXiv:2003.06071* (2020).

[19] Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. 2018. Hyperbolic attention networks. *arXiv preprint arXiv:1805.09786* (2018).

[20] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[21] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advancesin Neural Information Processing Systems*. 1024–1034.

[22] He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. *arXiv preprint arXiv:1704.07130* (2017).

[23] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 687–696.

[24] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. 2020. A survey on knowledge graphs: Representation, acquisition and applications. *arXiv preprint arXiv:2002.00388* (2020).

[25] Simon Keizer, Markus Guhe, Heriberto Cuayáhuitl, Ioannis Efstathiou, Klaus-Peter Engelbrecht, Mihai Dobre, Alex Lascarides, and Oliver Lemon. 2017. Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Volume 2: Short Papers*. Association for Computational Linguistics, 480–484.

[26] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[27] David Krackhardt. 2014. Graph theoretical dimensions of informal organizations. In *Computational Organization Theory*. Psychology Press, 107–130.

[28] John M Lee. 2013. Smooth manifolds. In *Introduction to Smooth Manifolds*. Springer, 1–31.

[29] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493* (2015).

[30] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *29th AAAI Conference on Artificial Intelligence*.

[31] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 212–220.

[32] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. 2015. YAGO3: A knowledge base from multilingual wikipedias. In *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Online Proceedings*. www.cidrdb.org.

[33] Yu Meng, Jiaxin Huang, Guangyuan Wang, Chao Zhang, Honglei Zhuang, Lance Kaplan, and Jiawei Han. 2019. Spherical text embedding. In *Advances in Neural Information Processing Systems*. 8208–8217.

[34] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 4710–4723.

[35] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. 2016. Stranse: a novel embedding model of entities and relationships in knowledge bases. *arXiv preprint arXiv:1606.08140* (2016).

[36] Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*. 6338–6347.

[37] Maximillian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*. PMLR, 3779–3788.

[38] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data.. In *Icml*, Vol. 11. 809–816.

[39] Peter Petersen, S Axler, and KA Ribet. 2006. *Riemannian geometry*. Vol. 171. Springer.

[40] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*. Springer, 593–607.

[41] Ondrej Skopek, Octavian-Eugen Ganea, and Gary Bécigneul. 2020. Mixed-curvature Variational Autoencoders. In *International Conference on Learning Representations*.

[42] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*. 926–934.

[43] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

[44] Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. 2018. Poincare Glove: Hyperbolic Word Embeddings. In *International Conference on Learning Representations*.

[45] 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*. 57–66.

[46] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. International Conference on Machine Learning (ICML).

[47] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*.

[48] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[49] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[50] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2018. Deep Graph Infomax. *arXiv preprint arXiv:1809.10341* (2018).

[51] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 417–426.

[52] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[53] Shen Wang, Zhengzhang Chen, Ding Li, Zhichun Li, Lu-An Tang, Jingchao Ni, Junghwan Rhee, Haifeng Chen, and Philip S Yu. 2019. Attentional heterogeneous graph neural network: Application to program reidentification. In *Proceedings of the 2019 SIAM International Conference on Data Mining*. SIAM, 693–701.

[54] Shen Wang, Zhengzhang Chen, Xiao Yu, Ding Li, Jingchao Ni, Lu-An Tang, Jiaping Gui, Zhichun Li, Haifeng Chen, and Philip S Yu. 2019. Heterogeneous graph matching networks for unknown malware detection. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 3762–3770.

[55] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 950–958.

[56] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence*.

[57] Richard C Wilson, Edwin R Hancock, Elżbieta Pekalska, and Robert PW Duin. 2014. Spherical and hyperbolic embeddings of data. *IEEE transactions on pattern analysis and machine intelligence* 36, 11 (2014), 2255–2269.

[58] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* (2020).

[59] Jiacheng Xu and Greg Durrett. 2018. Spherical latent spaces for stable variational autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

[60] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).

[61] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. *arXiv preprint arXiv:1806.08804* (2018).

[62] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. In *Thirty-Fourth AAAI conference on artificial intelligence*.