# HyperJOIE: Two-View Hyperbolic Knowledge Graph Embedding with Entities and Concepts Jointly

Jing Dong[1], Binbin Gu[2], Jianfeng Qu[1](✉), An Liu[1], Lei Zhao[1], Zhigang Chen[3,4], and Zhixu Li[1]

[1] School of Computer Science and Technology, Soochow University, Suzhou, China
{jdong,jfqu,anliu,zhaol,zhixuli}@stu.suda.edu.cn

[2] University of California, Irvnie, USA
binbing@uci.edu

[3] iFLYTEK Research, Suzhou, China
zgchen@iflytek.com

[4] State Key Laboratory of Cognitive Intelligence, iFLYTEK, Hefei, China

**Abstract.** Knowledge graphs have two views: an entity graph in the instance view and a concept graph in the ontology view. Recent studies reveal that modeling the two graphs jointly can benefit the understanding to either one. However, the existing work has flaws on both modelling the hierarchical structures in the Euclidean space, and capturing the deep cross-view interaction between an entity and its corresponding concept. In this paper, we propose to explore hyperbolic space for two-view knowledge graph embedding, which provides more effective and efficient embedding learning mechanism, especially for hierarchical structured knowledge. We also propose to capture the deep cross-view interaction between an entity and its corresponding concept through modeling local structure information from intra-view neighbor nodes with hyperbolic attention mechanism. Finally, we propose to maintain the structural correspondence between the concept graph and the entity graph by first encoding two graphs with the same embedding model respectively and then aligning the two graphs with a hyperbolic transformation. Our empirical study conducted on two benchmark data collections proves that our model outperforms several state-of-the-art two-view knowledge graph embedding models.

**Keywords:** Knowledge representation learning · Hyperbolic attention mechanism · Two-view knowledge graph

## 1 Introduction

Knowledge graph embedding (KGE), which encodes knowledge graph (KG) structures into low-dimensional embedding spaces, has attracted a lot of attention in the past decade [2,15,23]. As an effective way to capture the latent


W. Zhang et al. (Eds.): WISE 2021, LNCS 13080, pp. 305–320, 2021.
https://doi.org/10.1007/978-3-030-90888-1_24

semantic relations of entities and concepts, KGE provides an efficient and systematic solution to various knowledge-driven machine learning tasks such as knowledge graph completion [25], entity typing [26], entity alignment [8] and question answering [3].

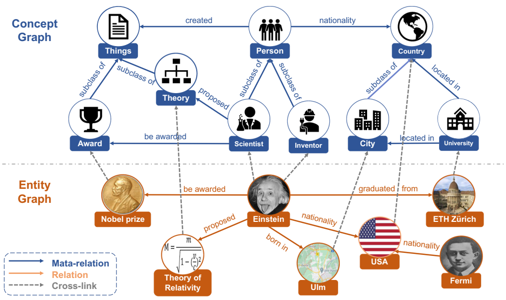



**Fig. 1.** An example knowledge graph in two views

Normally, a KG can be viewed in two levels as the example shown in Fig. 1, which includes an **entity graph** in the instance level and a **concept graph** in the ontology level. While most traditional KGE models [14,19] take concepts as special kind of entities, and learn the representations for both concepts and entities in the same way, some other efforts is limited to conduct KGE in a single view, that is, either on the instance-view graph [1,2] or on the ontology-view graph [7,9]. Some recent studies [10,14] reveal that representing a KG from both views provides more comprehensive insights, and thus improves the understanding to both views. A most representative work JOIE [10] proposes to jointly model the two graphs, i.e., entity graph and concept graph, and then characterize the cross-links between concepts and entities with a cross-view association model. While entity embeddings could provide detailed and rich information for their corresponding concepts, a concept embedding also provides a high-level summary of its corresponding entities, which is very helpful to long-tail instances [10]. Some other efforts [14,22] also works on leveraging concepts as supplementary information to assist entity representation learning.

However, the existing two-view KGE models are flawed in three aspects: (1) It is verified that the characterization of hierarchical structures in the concept meta-relations in the ontology view is crucial for KGE learning [10,14], where the hierarchical structures include hypernym-hyponym relations such as *(Person, sub_class_of, Scientist)*, and composition relations such as *(City, part_of, Country)*. Nevertheless, the existing work ignores the corresponding hierarchical structures in the instance graph, such as *(BeiJing, located_in, China)* and *(The Nobel Prize in Physics (1921), is_A, Nobel Prize)*. Thus, the learned entity embeddings are insufficient for the expression of hierarchical relationship. (2) So far, only conventional Euclidean space is employed for doing two-view KGE, but Euclidean space is not suitable for modeling hierarchical patterns, given that hierarchical patterns requires high-dimensional representation space to preserve different behaviors through different geometric patterns, but the space capacity in Euclidean space expands polynomially [24]. As a result, embedding hierarchical patterns in Euclidean space incurs huge memory costs. (3) The cross-view association model explored by the existing work only characterizes the connections between the entity and its corresponding concept, but ignores the local structure information explored from their intra-view neighbor nodes, which could provide deep cross-view interaction between the entity and its corresponding concept. For instance in Fig. 1, *"Einstein"*'s neighbor node *"Theory of Relativity"* in the entity graph, and *"Scientist"*'s neighbor node *"Theory"* could strengthen the connections between *"Einstein"* and *"Scientist"*.

To address the above flaws, we propose **HyperJOIE**, a novel two-view joint knowledge graph embedding model in a hyperbolic space. Compared to the Euclidean space, the hyperbolic space could offer greater capacity for embedding learning [18], thus is naturally more suitable for expressing hierarchical structures. Thus, the second flaw of the existing models is solved. Besides, modeling KGs in the hyperbolic space usually requires less dimensions than that in the Euclidean space, but can still reach competitive or even better results [5]. Although the hyperbolic space has been employed in single-view KGE [1,5], it is not applied to two-view KGE yet. To overcome the third flaw of the existing models, we would like to capture the local structure information from intra-view neighbor nodes with a recent proposed attention mechanism in the hyperbolic space [24]. Finally, while concept graph could be regarded as an abstract to the entity graph, the entity graph is an instantiation of the concept graph. In order to maintain the structural correspondence between the two graphs to the greatest extent, we propose to encode the two KGs with the same embedding model and then align the them with a hyperbolic transformation. In such a way, the first flaw of the existing work is handled.

To summarize, our contributions are as follows: (1) This work is the first attempt to explore two-view KGE in hyperbolic space, which provides more effective and efficient embedding learning mechanism, especially for hierarchical structured knowledge. (2) We propose to capture the deep cross-view interaction between the entity and its corresponding concept through modeling local structure information from intra-view neighbor nodes with an attention mechanisms in the hyperbolic space. (3) We propose to maintain the structural correspondence between the concept graph and the entity graph by first encoding the two KGs with the same embedding model respectively and then aligning the two KGs with a hyperbolic transformation.

Our experiments conducted on two datasets verifies the effectiveness of our model comparing to SOTA models.

## 2 Related Work

This section studies both single-view KGE models and two-view joint KGE models, and then covers some recent progress on hyperbolic KGE models.

### 2.1 Single-View KGE Models

Recent years have witnessed plenty of works contributing on the single-view KGE. A survey [11] categorizes KGE into four aspects, including representation space, scoring function, encoding models and auxiliary information. Given the triples $(h, r, t)$, where $r$ is the relation between head entity $h$ and tail entity $t$, the key of KGE is to choose a proper representation space, to devise an effective model encoding triples and to design a plausibility scoring function for optimizing objective. Most models are mainly based on Euclidean space [2,17,23], the others basically represent in complex vector space [20,21], whereas they are all linear embeddings that lack hierarchical expressiveness and spatial capacity. Although linear embedding methods are fairly simple, it is difficult to obtain enough capacity for tree structure data with the same dimensions as hyperbolic embedding. In addition, a few studies focus on embedding concept graph. On2Vec [7] proposes an augment method of the concept graph by replacing fine-grained concepts with coarse-grained concepts in the concept triples.

### 2.2 Two-View Joint KGE Models

Different from single-view KGE, two-view KGE not only trains two graphs respectively but also divides the single-view link prediction task into graph completion and type inference. Recent studies [10,14,22] show that the jointly modeling two-view graph can improve KG representation performance in both knowledge graph completion and entity typing. There have been several investigations [22] taking advantage of concepts as additional information to improve KG completion performance. Further, in order to model hierarchical relations, TransC [14] proposes to represent entities and concepts in the same embedding space with cross-view links association. Further, MTransE [8] aims at entity alignment and proposes to learn a transformation across two separate embedding spaces. However, TransC ignores differences in topology and data scale between entity graph and concept graph, and MTransE neglects the internal hierarchy of two graphs.

JOIE [10] is the first and the only joint two-view KGE model with the Euclidean space. This model introduces a hierarchical loss to model hierarchy and a transformation function to capture the correlation between entity graph and concept graph. Nevertheless, JOIE has weak hierarchy capacity since its representation space is the Euclidean space. Besides, owing to encoding two graphs with the uniform parameters, the problem of asynchronous convergence may occur between different modules, which limits the optimization process and is difficult to reach the global optimum.

### 2.3 Hyperbolic KGE Models

The recent progress in hyperbolic graph neural network challenges traditional graph neural network of Euclidean spaces, which fuels a lot of researches on introducing hyperbolic graph neural network into knowledge graph embedding.

**Table 1.** Characteristic properties of Euclidean, and hyperbolic geometries [13].

| Geometry | Property | |
|---|---|---|
| | Euclidean | Hyperbolic |
| Curvature $K$ | 0 | $< 0$ |
| Circle length | $2\pi r$ | $2\pi \sinh \sqrt{\lvert K \rvert} r$ |
| Disk area | $\pi r^2$ | $2\pi(\cosh \sqrt{\lvert K \rvert} r - 1)$ |

More recently, several studies on KG completion and alignment reveal that low-dimensional hyperbolic KG embeddings can represent knowledge graph effectively and efficiently. AttH [5], which achieves state-of-the-art results on single KG completion, utilizes hyperbolic geometrical operations (rotation and reflection) to learn KG's structure and introduce attention mechanism to balance two operations. Moreover, HyperKA [19] introduces hyperbolic translational embeddings within intra-view and train a hyperbolic transformation to capture the cross-link associations. Although these two works demonstrate that embedding knowledge into hyperbolic space can partly improve the performance of KG completion and entity typing, joint KGE learning in hyperbolic space is still an unexplored issue, which is exactly the focus of this paper.

## 3 Problem Formulation and Background

### 3.1 Problem Formulation

We define the entity graph as $\mathcal{G}_e = \{\mathcal{E}_e, \mathcal{R}_e, \mathcal{F}_e\}$ and the concept graph as $\mathcal{G}_c = \{\mathcal{E}_c, \mathcal{R}_c, \mathcal{F}_c\}$. The entity graph $\mathcal{G}_e$ and the concept graph $\mathcal{G}_c$ are two completely independent graphs, where $\mathcal{E}_e, \mathcal{R}_e$ are the sets of entities and relations and $\mathcal{E}_c, \mathcal{R}_c$ are the sets of concepts and meta-relations. Correspondingly, we represent the entity triples and concept triples as $(h_e, r_e, t_e) \in \mathcal{F}_e$ and $(h_c, r_c, t_c) \in \mathcal{F}_c$, and the cross-view triples as $(h_e, t_c) \in \mathcal{F}_{cross}$. Formally, the task of two-view KGE is to represent the embeddings of entities $e$, relations $r_e$, concepts $c$ and meta-relations $r_c$, which are represented by the boldfaced $\mathbf{e}$, $\mathbf{r_e}$, $\mathbf{c}$ and $\mathbf{r_c}$, respectively. In order to train and evaluate the model, three triples sets ($\mathcal{F}_e$, $\mathcal{F}_c$, $\mathcal{F}_{cross}$) are split into $\mathcal{F}^{Train}$, $\mathcal{F}^{Valid}$ and $\mathcal{F}^{Test}$, respectively. Embeddings are optimized by the scoring function $f(,,)$, which measures the plausibility of facts.

### 3.2 Preliminaries on Hyperbolic Geometry

Hyperbolic space is a space with constant negative curvature. As shown in Table 1, non-zero curvature makes hyperbolic geometry different from traditional Euclidean geometry. It can be seen as the calculated metrics of the circle length and the disk area that the hyperbolic geometric space increases exponentially. This property allows hyperbolic space to provide greater spatial capacity under the same dimensional constraints, which is particularly suitable for forming hierarchies. Further, to facilitate the gradient descent operation, we choose the Poincaré ball model [4] as hyperbolic geometric equivalent model in this paper. Particularly, a n-dimensional Poincaré ball with negative curve $-c$ ($c > 0$) is defined as a mainfold $\mathbb{B}_c^n = x \in \mathbb{R}^n : c\|x\| < 1$, and we set $c = 1$ to simplify our work following [18]. Some hyperbolic geometry in this n-dimensional mainfold $\mathbb{B}_1^n$ are introduced below.

**Möbius Addition and Hyperbolic Distance.** Given two nodes $\mathbf{u}, \mathbf{v} \in \mathbb{B}_1^n$, the Möbius addition provides an analogue to Euclidean addition for hyperbolic space, which is:

$$\mathbf{u} \oplus \mathbf{v} = \frac{(1 + 2\langle \mathbf{u}, \mathbf{v}\rangle + \|\mathbf{v}\|^2)u + (1 - \|\mathbf{u}\|^2)\mathbf{v}}{1 + 2\langle \mathbf{u}, \mathbf{v}\rangle + \|\mathbf{u}\|^2 + \|\mathbf{v}\|^2} \tag{1}$$

Further, the hyperbolic distance on $\mathbb{B}_1^n$ between $\mathbf{u}$ and $\mathbf{v}$ is given by:

$$d_{\mathbb{B}_1^n}(\mathbf{u}, \mathbf{v}) = 2 \tanh^{-1}(\| - \mathbf{u} \oplus \mathbf{v}\|) \tag{2}$$

**Transformation.** The logarithmic map $\log_0()$ maps the mainfold $\mathbb{B}_c^n$ to the tangent space $\mathcal{T}_0\mathbb{B}_c^n$ at the origin 0, where $\mathcal{T}_0\mathbb{B}_c^n$ is a n-dimensional Euclidean space for $\mathbb{B}_c^n$, i.e. $\mathcal{T}_0\mathbb{B}_c^n = \mathbb{R}^n$ [1] . Conversely, the exponential map $\exp_0()$ projects $\mathcal{T}_0\mathbb{B}_c^n$ onto $\mathbb{B}_c^n$ at the origin. Specifically, for each point $\mathbf{u} \in \mathbb{B}_1^n, \mathbf{y} \in \mathcal{T}_0\mathbb{B}_c^n$, these maps are formulated as:

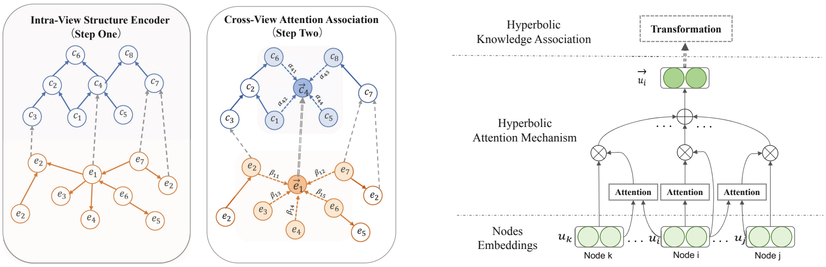
$$\begin{aligned} \log_0(\mathbf{u}) &= \tanh^{-1}(\|\mathbf{u}\|)\frac{\mathbf{u}}{\|\mathbf{u}\|} \\ \exp_0(\mathbf{y}) &= \tanh(\|\mathbf{y}\|)\frac{\mathbf{y}}{\|\mathbf{y}\|} \end{aligned} \tag{3}$$

With such two inverse maps, hyperbolic space can apply analogous Euclidean operations. Specifically, through mapping the hyperbolic vectors to the tangent space with logarithmic map, the Euclidean linear matrix-vector multiplication can be applied to the transformation matrix $\mathbf{M}$, and then the transformed vectors can be projected back on another mainfold with exponential map. This transformation that projects a vector $\mathbf{u} \in \mathbb{B}_1^n$ into $\mathbb{B}_1^m$ [19] is denoted as Möbius matrix-vector multiplication:

$$\mathbf{M} \otimes \mathbf{u} = \exp_0(\mathbf{M} \log_0(\mathbf{u})) \tag{4}$$

## 4 Methodology

In this section, we present our model **HyperJOIE**—a two-view hyperbolic KGE method with jointly learning entities and concepts. Figure 2(a) demonstrates the framework of **HyperJOIE** which consists of two components: intra-view structure encoder and cross-view attention association. In the following, we describe the two modules in details.




(a) HyperJOIE Framework (b) Cross-View Attention Association

**Fig. 2.** (a) is the framework of HyperJOIE which first encodes intra-view KGs and then associates entities and concepts with aggregated hyperbolic attention representation. (b) is the framework of hyperbolic attention associations.

### 4.1 Intra-view Model

The goal of the intra-view model is to encode two independent KGs into two embedding spaces while preserving original structural information. Since hierarchical structures do not only appear in concept mapping, we use an advanced hyperbolic single-view model AttH [5] to represent the entity and concept graph respectively, so as to represent the hierarchical relations inside both KGs. In the following, we describe the AttH model.

AttH utilizes two kinds of hyperbolic isometries—rotations and reflections, to respectively model compositions and anti-symmetric relations. $\Theta_r := (\theta_{r,i})_{i\in\{1,\ldots\frac{n}{2}\}}$ are rotation parameters and $\Phi_r := (\phi_{r,i})_{i\in\{1,\ldots\frac{n}{2}\}}$ are reflection parameters, where $n$ is the embedded dimension. Through $2\times 2$ transformation matrices $G^{\pm}(\theta)$, AttH parameterizes rotation and reflection with the form of a block-diagonal matrices as $\mathrm{Rot}(\Theta_r)$ and $\mathrm{Ref}(\Phi_r)$. Then it applies relation-specific rotations and reflections to the head embedding (Eq. 6), using $H$ represent the embeddings in the Hyperbolic space.

$$\begin{aligned}\mathrm{Rot}(\Theta_r) &= diag(G^{+}(\theta_{r,1}),\ldots,G^{+}(\theta_{r,\frac{n}{2}})),\\ \mathrm{Ref}(\Phi_r) &= diag(G^{-}(\phi_{r,1}),\ldots,G^{-}(\phi_{r,\frac{n}{2}})),\\ \text{where}\quad G^{\pm}(\theta) &:= \begin{bmatrix}\cos(\theta) & \mp\sin(\theta)\\ \sin(\theta) & \pm\cos(\theta)\end{bmatrix}.\end{aligned} \tag{5}$$

$$\mathbf{q}^{H}_{\mathrm{Rot}} = \mathrm{Rot}(\Theta_r)\mathbf{u}^{H}_{h}, \quad \mathbf{q}^{H}_{\mathrm{Ref}} = \mathrm{Ref}(\Phi_r)\mathbf{u}^{H}_{h} \tag{6}$$

In order to balance the combination of rotations and reflections, AttH takes advantage of the convenience of attention calculation in tangent space. It projects hyperbolic representations to tangent space representations $\mathbf{u}^{E} = \log_0(\mathbf{u}^{H})$ to gets attention weights $\alpha_{\mathbf{u}}, \alpha_{\mathbf{v}}$, and then gets a weighted average combination as:

$$(\alpha_{\mathbf{u}}, \alpha_{\mathbf{v}}) = \mathrm{Softmax}(\mathbf{a}^{T}\mathbf{u}^{E}, \mathbf{a}^{T}\mathbf{v}^{E}) \tag{7}$$

$$\mathrm{Att}(\mathbf{u}^{H}, \mathbf{v}^{H}; \mathbf{a}) = \exp_0(\alpha_{\mathbf{u}}\mathbf{u}^{E} + \alpha_{\mathbf{v}}\mathbf{v}^{E}) \tag{8}$$

After combining two hyperbolic geometric representations ($\mathbf{q}_{\text{Rot}}^H$ and $\mathbf{q}_{\text{Ref}}^H$), AttH utilizes a hyperbolic translation to get the translated head embedding $Q(h, r)$. Finally, based on the hyperbolic distance, the score function compares the translated head embeddings to the target tail embeddings (Eq. 10).

$$Q(h,r) = \text{Att}(\mathbf{q}_{\text{Rot}}^H, \mathbf{q}_{\text{Ref}}^H; \mathbf{a}_r) \oplus \mathbf{r}^H \tag{9}$$

$$f(h,r,t) = -d_{\mathbb{B}}(Q(h,r), \mathbf{t}^H)^2 + b_h + b_t \tag{10}$$

where $(b_u)_{u\epsilon\mathcal{E}}$ are node biases which act as margins in the scoring function [1,5]. Then, the intra-view component is trained by minimizing the full cross-entropy loss, with negative samples:

$$\begin{aligned} \mathcal{L}_{intra} &= \sum_{t' \sim \mathcal{U}(\mathcal{E})} \log(1 + \exp(y'_t f(h,r,t'))) \\ \text{where} \qquad y'_t &= \begin{cases} -1 & \text{if } t' = t \\ 1 & \text{otherwise} \end{cases} \end{aligned} \tag{11}$$

### 4.2 Cross-View Model

The cross-view model component aims at modeling the associations between entity graph and concept graph. Figure 2(b) demonstrates the framework of hyperbolic cross-view attention associations. We take the concepts and entities embeddings encoded by the intra-view component as input, and then aggregate the latent information with hyperbolic attention mechanism for concepts and entities respectively. Finally, the hyperbolic knowledge association is adopted to associate the aggregated entities and aggregated concepts. Next, we introduce the hyperbolic attention mechanism and the hyperbolic knowledge association in details.

**Hyperbolic Attention Mechanism.** The attention mechanism is helpful to capture the high-order proximity of nodes based on local neighborhood information. Based on this, we utilize hyperbolic attention mechanism to aggregate the latent representations. Inspired by the recent proposal of HAT [24], we perform the self-attention mechanism on the nodes to capture the deep cross-view interactions. The attention value $\alpha_{ij}$ indicates the importance of node $j$ to node $i$, which is measured by the hyperbolic distance from node $i$ to node $j$. Given two nodes $\mathbf{u}_i, \mathbf{u}_j \in \mathbb{B}_1^n$, the attention weight $\alpha_{ij}$ is computed as:

$$\begin{aligned} \alpha_{ij} &= -d_{\mathbb{B}}(\mathbf{u}_i, \mathbf{u}_j) \\ &= -\tanh^{-1}(| -\mathbf{u}_i \oplus \mathbf{u}_j|) \end{aligned} \tag{12}$$

According to HAT [24] using hyperbolic distance to measure attention weights can not only preserve the structure transitivity among nodes but also maintain the original characteristics of the nodes. This is because the attention weight of a node to itself is $\alpha_{ii} = -d(\mathbf{u}_i, \mathbf{u}_i) = 0$ meaning that a node is always the nearest point to itself among its neighbors. Moreover, the relative attention weights $w_{ij}$ is calculated by $softmax$ function over all the neighbors of node $i$ (including itself) to get the normalized values:

$$w_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{k \epsilon N_i} \exp(\alpha_{ik})} \tag{13}$$

The aggregated representation $\overrightarrow{\mathbf{u}_i}$ is a linear combination of the normalized weights $w_{ij}$ and the latet representations of all the nodes $j \epsilon N_i$, which is denoted as follows:

$$\overrightarrow{\mathbf{u}_i} = \sigma(\sum\nolimits_{j \epsilon N_i}^{\oplus} w_{ij} \otimes \mathbf{u}_j) \tag{14}$$

where the $\sum\nolimits_{j \epsilon N_i}^{\oplus}$ is the accumulation of Möbius addition. Different from HAT, HyperJOIE adopts TanH as the nonlinearity function $\sigma$ and calculates the weighted sum by the Möbius scalar multiplication. Formally, given the weight $w_{ij} \in \mathbb{R}$ of the node $u_i \in \mathbb{B}_1^n$, the weighted sum is denoted as:

$$w_{ij} \otimes \mathbf{u}_i = \tanh(w_{ij} \tanh^{-1}(\|\mathbf{u}_i\|)) \frac{\mathbf{u}_i}{\|\mathbf{u}_i|} \tag{15}$$

In addition, with the help of the logarithmic map, we transform the weighted representation $w_{ij} \otimes \mathbf{u}_i$ into the tangent space, where we can utilize the linear combination as in the Euclidean spaces, i.e. $\sum_{j \in N_i} \log_0(w_{ij} \otimes \mathbf{u}_j)$ [6]. After that, we use the exponential map to project the representation back to the hyperbolic space, simplifying the final representation as follows:

$$\overrightarrow{\mathbf{u}_i} = \sigma(\exp_0(\sum\nolimits_{j \epsilon N_i} \log_0(w_{ij} \otimes \mathbf{u}_j))) \tag{16}$$

**Hyperbolic Knowledge Association.** Given a pair of cross-link triples $(e, c) \in \mathcal{F}_{cross}$, we firstly utilize the Möbius transformation to project the aggregated entity $\overrightarrow{\mathbf{e}}$ to the concept embedding space. For each cross-link triple, we hope the transformed aggregated entity vector close to the aggregated concept vector $\overrightarrow{\mathbf{c}}$ in concept embedding space, so we choose the hyperbolic distance to design the scoring function. Inspired by [1], we add the biases of entity $b_e \in \mathbb{R}$ and the biases of concept $b_c \in \mathbb{R}$ to manifest the margins among instantiated entities of the same concept. Finally, we define the basis score function for the cross-view attention association as follows:

$$f(e, c) = -d_{\mathbb{B}}(\mathbf{M} \otimes \overrightarrow{\mathbf{e}}, \overrightarrow{\mathbf{c}}) + b_e + b_c \tag{17}$$

where $\mathbf{M} \in \mathcal{R}_{n \times m}$ transforms the hyperbolic vectors from $\mathbb{B}_1^n$ to $\mathbb{B}_1^m$.

We choose the full cross-entropy loss with uniform negative sampling to train cross-view component, where the negative triples are sampled uniformly from all possible triples by perturbing the tail concept. The cross-view attention association loss is given as:

$$\mathcal{L}_{cross} = \sum_{c' \sim \mathcal{U}(\mathcal{E}_c)} \log(1 + \exp(y_{c'} f(e, c'))),$$
$$\text{where} \qquad y_c' = \begin{cases} -1 & \text{if } c' = c \\ 1 & \text{otherwise} \end{cases} \tag{18}$$

### 4.3 Loss Function and Training

The total loss is the combination of the intra-view component and the cross-view component, which is defined as follows:

$$\mathcal{L} = \mathcal{L}_{intra}^{\mathcal{G}_e} + \mathcal{L}_{intra}^{\mathcal{G}_c} + \mathcal{L}_{cross} \tag{19}$$

In order to avoid the challenging optimization in hyperbolic space, we define all parameters in the tangent space at the origin and optimize parameters with the Adam optimizer [12]. The parameters can be recovered to the hyperbolic space with the exponential map [5]. We follow a two-step training procedure: For one epoch, (1) first train our intra-view component $\mathcal{L}_{intra}^{\mathcal{G}_e}$ and $\mathcal{L}_{intra}^{\mathcal{G}_c}$ respectively and then (2) optimize the cross-view loss $\mathcal{L}_{cross}$ in the successive step.

## 5 Experiments

### 5.1 Experimental Setup

**Datasets.** We use two benchmark datasets for evaluation: YAGO26K-906 and DB111K-174 [10], which are extracted from YAGO and DBpedia, respectively. Each dataset includes entity graph, concept graph and cross-view links. Table 2 provides statistics of all datasets used. We use the data splits provided by [10] in order to compare with previous work.

**Evaluation Metrics.** During the test, we use different scoring function to test different tasks. Specifically, we use Eq. 10 for intra-view task and Eq. 17 for cross-view task. Similar to previous work, we compute two ranking-based metrics: (1) mean reciprocal rank (MRR) and (2) hits at $K$ ($H@K, K \in 1, 3, 10$). Following previous standard evaluation protocol [2], we filter out all true triples from the dataset to put penalty on the true triples which are predicted with low rankings.

**Hyper-Parameter Settings.** As for the hyper-parameters, we conduct a grid search to detect the learning rate and batch size, using the validation set to select the best hyper-parameters. We search the learning rate among $\{0.00005, 0.0001, 0.0002, 0.0005\}$, and batch size among $\{200, 1000, 2000, 5000\}$. The optimal configurations are as follows: On YAGO26K-906, the learning rate $\alpha = 0.0001$ and batch size $B = 1000$; while on DB111K-174, the learning rate $\alpha = 0.0001$ and batch size $B = 5000$.

**Table 2.** Statistics of datasets.

| Datasets | | # Nodes | # Rel | # Triples | # Cross-links |
|---|---|---|---|---|---|
| YAGO26K-906 | Ent | 26,078 | 34 | 390,738 | 9,962 |
| | Cpt | 906 | 30 | 8,962 | |
| DB111K-174 | Ent | 111,762 | 305 | 863,643 | 99,748 |
| | Cpt | 174 | 20 | 763 | |

**Table 3.** KG completion results

| Dataset | YAGO26K-906 | | | | | | DB111K-174 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Entity | | | Concept | | | Entity | | | Concept | | |
| Metrics | MRR | H@1 | H@10 | MRR | H@1 | H@10 | MRR | H@1 | H@10 | MRR | H@1 | H@10 |
| TransE(base) | 0.195 | 0.141 | 0.345 | 0.145 | 0.123 | 0.206 | 0.327 | 0.223 | 0.490 | 0.313 | 0.232 | 0.469 |
| TransE(all) | 0.189 | 0.137 | 0.350 | 0.189 | 0.147 | 0.244 | 0.318 | 0.227 | 0.481 | 0.539 | 0.479 | 0.618 |
| DistMult(base) | 0.253 | 0.229 | 0.288 | 0.197 | 0.177 | 0.251 | 0.265 | 0.256 | 0.276 | 0.235 | 0.152 | 0.291 |
| DistMult(all) | 0.288 | 0.240 | 0.312 | 0.156 | 0.143 | 0.165 | 0.280 | 0.272 | 0.297 | 0.501 | 0.455 | 0.647 |
| HolE(base) | 0.265 | 0.259 | 0.283 | 0.192 | 0.187 | 0.203 | 0.301 | 0.292 | 0.315 | 0.227 | 0.189 | 0.328 |
| HolE(all) | 0.252 | 0.242 | 0.266 | 0.138 | 0.113 | 0.114 | 0.290 | 0.287 | 0.303 | 0.432 | 0.388 | 0.560 |
| RotatE(base) | 0.652 | 0.563 | 0.806 | 0.252 | 0.185 | 0.376 | 0.435 | 0.372 | 0.550 | 0.506 | 0.422 | 0.669 |
| RotatE(all) | 0.567 | 0.465 | 0.753 | 0.304 | 0.234 | 0.436 | 0.337 | 0.287 | 0.427 | 0.445 | 0.377 | 0.552 |
| AttH(base) | 0.667 | 0.589 | 0.804 | 0.250 | 0.199 | 0.344 | 0.425 | 0.348 | 0.566 | 0.520 | 0.429 | 0.701 |
| AttH(all) | 0.610 | 0.519 | 0.549 | 0.262 | 0.203 | 0.380 | 0.372 | 0.309 | 0.595 | 0.475 | 0.403 | 0.675 |
| TransC | 0.252 | 0.157 | 0.378 | – | – | – | 0.359 | 0.248 | 0.493 | – | – | – |
| JOIE | 0.327 | 0.224 | 0.524 | 0.263 | 0.209 | 0.385 | 0.473 | 0.338 | 0.714 | 0.622 | 0.581 | 0.797 |
| HyperKA | 0.314 | 0.161 | 0.613 | 0.269 | 0.169 | 0.429 | 0.302 | 0.147 | 0.387 | 0.241 | 0.200 | 0.447 |
| HyperJOIE(no_att) | 0.735 | 0.683 | 0.826 | 0.523 | 0.413 | 0.739 | 0.635 | 0.584 | 0.729 | 0.725 | 0.645 | 0.870 |
| HyperJOIE(with_att) | **0.745** | **0.697** | **0.830** | **0.587** | **0.497** | **0.761** | **0.708** | **0.656** | **0.804** | **0.768** | **0.667** | **0.906** |

## 5.2 KG Completion

**Baselines.** Although lots of methods contribute on single-view KG completion, only a few methods have been applied to the two-view KG completion task with jointly learning entities and concepts. In this work, we compare HyperJOIE to SotA single-view KGE model AttH [5], SotA two-view KGE method JOIE [10], SotA two-view KG alignment method HyperKA [19] and five other baseline methods TransE [2], DistMult [23], HolE [16], TransC [14] and RotatE [20]. For JOIE, we take its best performance for fair comparison. For the other baselines, we set the entity dimension $n$ to 300 and the concept dimension $m$ to 50. We also tried to set higher dimensions, but we did not observe further improvements.

**Ablations.** To analyze the benefits of hyperbolic joint framework, we follow the JOIE which deploys two variants of AttH: (1) We train each independent knowledge graph separately without cross-view associations, denoted as *(base)*; (2) We train all triples in $\mathcal{F}_e^{Train}$, $\mathcal{F}_c^{Train}$ and $\mathcal{F}_{cross}^{Train}$, denoted as *(all)*. Moreover, to evaluate the role of hyperbolic attention in the intra-view component, we report scores for variants of HyperJOIE training cross-view associations only with hyperbolic transformation, denoted as *(no_att)*. Furthermore, we re-evaluate the intra-view KG completion performance of HyperKA [19] to estimate the intra-view embedding ability of the SotA entity typing method.

**Results and Discussion.** We report the KG completion results in Table 3. As can be seen from Table 3, HyperJOIE outperforms all the other methods. Compared with the linear embedding methods (TransE, DistMult, HolE, TransC and JOIE ) in Euclidean space and RotatE [20] in the complex space, HyperJOIE is extremely dominant regarding the performance. For example, the H@1 score of HyperJOIE on entity graph of YAGO26K-906 reaches 0.697, surpassing JOIE by 0.473 and RotatE by 0.134. Also, compared with hyperbolic single-view KGE model, HyperJOIE also achieves better performance on both entity graph and concept graph. Specifically, HyperJOIE improves AttH by 0.108 and 0.238 in H@1 on entity and concept graph completion of YAGO26K-906 respectively. This illustrates that hyperbolic projections for cross-view associations can enhance the embedding representation of the intra-view graphs, especially in concept graph. In addition, we evaluate the performance of HyperJOIE without hyperbolic attention association HyperJOIE(no_att). We find that the version with hyperbolic attention associations HyperJOIE(with_att) shows slight improvement. This indicates that the hyperbolic attention mechanism also contributes to the representation of the intra-view KGs. Moreover, by comparing the performance of AttH and HyperKA on KG completion task, we find that even the SotA entity alignment approach is still lacking in terms of intra-view KGs, which demonstrates the importance of intra-view structure encoder.

### 5.3 Entity Typing

**Baselines.** We compare our method with the SotA entity typing methods (JOIE and HyperKA), entity alignment method MTransE [8], and five other single-view KGE methods, i.e. TransE [2], DistMult [23], HolE [16], TransC [14] and RotatE [20]. The dimension settings are same as Sect. 5.2.

**Ablations.** We deploys two variants of intra-view structure encoder to detect the impact of single-view KGE methods with different learning abilities on cross-view association: (1) We randomly initialize the embedding of entities and concepts without any intra-view structure modeling, denoted as (*basic*); (2) We transpose TransE [2] into hyperbolic space as encoder, denoted as (*simple*). Additionally, to verify the effect of hyperbolic association on cross-view links, we consider an alternative of our method without hyperbolic attention mechanism, denoted as (*no_att*).

**Results and Discussion.** As shown in Table 4, we observe that HyperJOIE outperforms both HyperKA and JOIE on MRR and H@1 on YAGO26K-906 in entity typing task. However, on the DB111K-174, our method is only 0.007

**Table 4.** Entity typing results

| Dataset | YAGO26K-906 | | | DB111K-174 | | |
|---|---|---|---|---|---|---|
| Metrics | MRR | H@1 | H@3 | MRR | H@1 | H@3 |
| TransE | 0.144 | 0.073 | 0.353 | 0.503 | 0.437 | 0.608 |
| DistMult | 0.411 | 0.361 | 0.553 | 0.551 | 0.680 | 0.551 |
| HolE | 0.395 | 0.348 | 0.548 | 0.504 | 0.654 | 0.504 |
| RotatE | 0.479 | 0.429 | 0.507 | 0.382 | 0.309 | 0.434 |
| AttH | 0.476 | 0.430 | 0.502 | 0.498 | 0.447 | 0.535 |
| MTransE | 0.689 | 0.609 | 0.776 | 0.672 | 0.599 | 0.813 |
| JOIE | 0.897 | 0.856 | **0.959** | <u>0.857</u> | 0.756 | **0.959** |
| HyperKA | <u>0.913</u> | <u>0.871</u> | <u>0.948</u> | **0.863** | **0.789** | <u>0.927</u> |
| HyperJOIE(*basic_no_att*) | 0.534 | 0.398 | 0.594 | 0.470 | 0.336 | 0.558 |
| HyperJOIE(*basic_with_att*) | 0.804 | 0.733 | 0.853 | 0.546 | 0.407 | 0.637 |
| HyperJOIE(*simple_no_att*) | 0.598 | 0.450 | 0.666 | 0.646 | 0.507 | 0.749 |
| HyperJOIE(*simple_with_att*) | 0.836 | 0.765 | 0.890 | 0.651 | 0.523 | 0.745 |
| HyperJOIE(*no_att*) | 0.870 | 0.806 | 0.922 | 0.735 | 0.642 | 0.887 |
| HyperJOIE(*with_att*) | **0.916** | **0.880** | 0.942 | 0.846 | 0.782 | 0.907 |

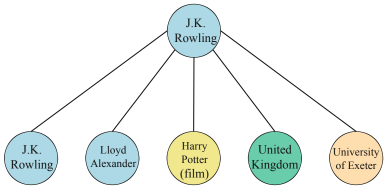
less than the SotA HyperKA on the MRR metric. To estimate the impact of different intra-view structure encoder, we compare HyperJOIE (*basic_no_att*), (*simple_no_att*) and (*no_att*), and find that HyperJOIE(*no_att*) achieves the best performance. This illustrates that high-quality intra-view embedding is beneficial for cross-view associations.

To further evaluate the role of hyperbolic attention and hyperbolic association, we compare HyperJOIE(*with_att*) with AttH and HyperKA on entity typing task, and find that HyperJOIE is superior to both hyperbolic KGE methods on YAGO26K-906, i.e. the H@1 metric of our methods surpasses HyperKA by 0.009 and surpasses AttH by 0.450 in YAGO26K-906. As for the comparison between (*no_att*) and (*with_att*), we can see that hyperbolic attention can enhance entity typing performance.

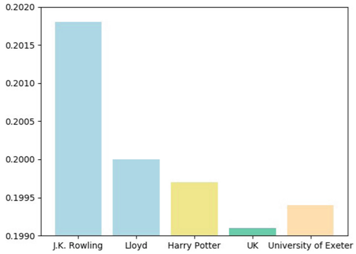
Further, to figure out the reasons of the poor performance of the hyperbolic attention mechanism on the DB111K-174, we compare the two datasets and find that the concept graph of DB111K-174 is too small and spare to provide enough latent neighbor information. The concept graph only has 763 triples (8,962 in YAGO26K-906) and an average of 4.38 triples per concept (9.89 in YAGO26K-906). But for a relatively complete concept graph, hyperbolic attention can significantly empower knowledge associations of the two-view KG.

### 5.4 Case Study

For case study, we visualize the attention weights of "J.K. Rowling"'s neighbors in DB111K-174 dataset as an illustrative example of the hyperbolic attention.




(a) "J.K. Rowling"'s neighbors




(b) Neighbors' weights

**Fig. 3.** Neighbors of "J.K. Rowling" and its attention weights

As shown in Fig. 3(a), "J.K. Rowling" has 4 neighbors, and the types of nodes are identified by its colors. From Fig. 3(b), we observe that "J.K. Rowling" gets highest weight and "Lloyd" gets the second attention weight. This means the hyperbolic attention mechanism can capture the type information.

## 6 Conclusion and Future Work

We propose a novel hyperbolic two-view knowledge graph embedding method with hyperbolic attention associations, which is the first attempt to explore two-view KGE in hyperbolic space. Our method almost surpasses SotA baselines on both KG completion and entity typing. Our future work will explore the application of a hyperbolic joint learning framework to multi-modal graph representation learning.

**Acknowledgments.** This work was supported by the National Key R&D Program of China (No. 2018AAA0101900), the Priority Academic Program Development of Jiangsu Higher Education Institutions, National Natural Science Foundation of China (Grant No. 62072323, 61632016, 62102276), Natural Science Foundation of Jiangsu Province (No. BK20191420).

## References

1. Balazevic, I., Allen, C., Hospedales, T.: Multi-relational poincaré graph embeddings. In: Advances in Neural Information Processing Systems, pp. 4463–4473 (2019)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, pp. 2787–2795 (2013)
3. Bordes, A., Weston, J., Usunier, N.: Open question answering with weakly supervised embedding models. In: Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) ECML PKDD 2014. LNCS (LNAI), vol. 8724, pp. 165–180. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44848-9_11

4. Cannon, J.W., Floyd, W.J., Kenyon, R., Parry, W.R., et al.: Hyperbolic geometry. Flavors Geom. **31**, 59–115 (1997)
5. Chami, I., Wolf, A., Juan, D.C., Sala, F., Ravi, S., Ré, C.: Low-dimensional hyperbolic knowledge graph embeddings. arXiv preprint arXiv:2005.00545 (2020)
6. Chami, I., Ying, R., Ré, C., Leskovec, J.: Hyperbolic graph convolutional neural networks. Adv. Neural. Inf. Process. Syst. **32**, 4869 (2019)
7. Chen, M., Tian, Y., Chen, X., Xue, Z., Zaniolo, C.: On2vec: embedding-based relation prediction for ontology population. In: Proceedings of the 2018 SIAM International Conference on Data Mining, pp. 315–323. SIAM (2018)
8. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. arXiv preprint arXiv:1611.03954 (2016)
9. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 192–202 (2016)
10. Hao, J., Chen, M., Yu, W., Sun, Y., Wang, W.: Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1709–1719 (2019)
11. Ji, S., Pan, S., Cambria, E., Marttinen, P., Yu, P.S.: A survey on knowledge graphs: representation, acquisition and applications. arXiv preprint arXiv:2002.00388 (2020)
12. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
13. Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., Boguná, M.: Hyperbolic geometry of complex networks. Phys. Rev. E **82**(3), 036106 (2010)
14. Lv, X., Hou, L., Li, J., Liu, Z.: Differentiating concepts and instances for knowledge graph embedding. arXiv preprint arXiv:1811.04588 (2018)
15. Nathani, D., Chauhan, J., Sharma, C., Kaul, M.: Learning attention-based embeddings for relation prediction in knowledge graphs. arXiv preprint arXiv:1906.01195 (2019)
16. Nickel, M., Rosasco, L., Poggio, T.: Holographic embeddings of knowledge graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)
17. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: ICML, vol. 11, pp. 809–816 (2011)
18. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: Advances in Neural Information Processing Systems, pp. 6338–6347 (2017)
19. Sun, Z., Chen, M., Hu, W., Wang, C., Dai, J., Zhang, W.: Knowledge association with hyperbolic knowledge graph embeddings. arXiv preprint arXiv:2010.02162 (2020)
20. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)
21. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International Conference on Machine Learning (ICML) (2016)
22. Xie, R., Liu, Z., Sun, M.: Representation learning of knowledge graphs with hierarchical types. In: IJCAI, pp. 2965–2971 (2016)
23. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2014)

24. Zhang, Y., Wang, X., Jiang, X., Shi, C., Ye, Y.: Hyperbolic graph attention network. arXiv preprint arXiv:1912.03046 (2019)
25. Zhang, Z., Cai, J., Zhang, Y., Wang, J.: Learning hierarchy-aware knowledge graph embeddings for link prediction. In: AAAI, pp. 3065–3072 (2020)
26. Zhao, Y., Zhang, A., Xie, R., Liu, K., Wang, X.: Connecting embeddings for knowledge graph entity typing. arXiv preprint arXiv:2007.10873 (2020)