Low-Dimensional Hyperbolic Knowledge Graph Embeddings

Ines Chami[1]*, **Adva Wolf**[1], **Da-Cheng Juan**[2], **Frederic Sala**[1], **Sujith Ravi**[3]† and **Christopher Ré**[1]
[1]Stanford University
[2]Google Research
[3]Amazon Alexa
{chami,advaw,fredsala,chrismre}@cs.stanford.edu
dacheng@google.com
sravi@sravi.org

Abstract

Knowledge graph (KG) embeddings learn low-dimensional representations of entities and relations to predict missing facts. KGs often exhibit hierarchical and logical patterns which must be preserved in the embedding space. For hierarchical data, hyperbolic embedding methods have shown promise for high-fidelity and parsimonious representations. However, existing hyperbolic embedding methods do not account for the rich logical patterns in KGs. In this work, we introduce a class of hyperbolic KG embedding models that simultaneously capture hierarchical and logical patterns. Our approach combines hyperbolic reflections and rotations with attention to model complex relational patterns. Experimental results on standard KG benchmarks show that our method improves over previous Euclidean- and hyperbolic-based efforts by up to 6.1% in mean reciprocal rank (MRR) in low dimensions. Furthermore, we observe that different geometric transformations capture different types of relations while attention-based transformations generalize to multiple relations. In high dimensions, our approach yields new state-of-the-art MRRs of 49.6% on WN18RR and 57.7% on YAGO3-10.

1 Introduction

Knowledge graphs (KGs), consisting of (*head entity*, *relationship*, *tail entity*) triples, are popular data structures for representing factual knowledge to be queried and used in downstream applications such as word sense disambiguation, question answering, and information extraction. Real-world KGs such as Yago (Suchanek et al., 2007) or Wordnet (Miller, 1995) are usually incomplete, so a common approach to predicting missing links in KGs is via embedding into vector spaces. Embedding methods learn representations of entities and relationships that preserve the information found in the graph, and have achieved promising results for many tasks.

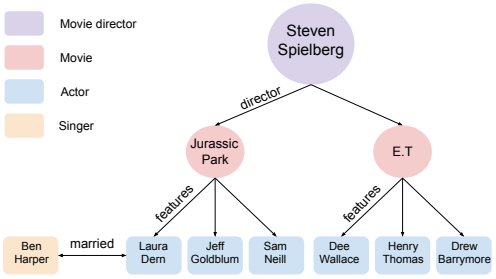


Figure 1: A toy example showing how KGs can simultaneously exhibit hierarchies and logical patterns.

Relations found in KGs have differing properties: for example, (*Michelle Obama*, *married to*, *Barack Obama*) is symmetric, whereas hypernym relations like (*cat*, *specific type of*, *feline*), are not (Figure 1). These distinctions present a challenge to embedding methods: preserving each type of behavior requires producing a different geometric pattern in the embedding space. One popular approach is to use extremely high-dimensional embeddings, which offer more flexibility for such patterns. However, given the large number of entities found in KGs, doing so yields very high memory costs.

For hierarchical data, hyperbolic geometry offers an exciting approach to learn low-dimensional embeddings while preserving latent hierarchies. Hyperbolic space can embed trees with arbitrarily low distortion in just two dimensions. Recent research has proposed embedding hierarchical graphs into these spaces instead of conventional Euclidean space (Nickel and Kiela, 2017; Sala et al., 2018). However, these works focus on embedding simpler graphs (e.g., weighted trees) and cannot express the diverse and complex relationships in KGs.

We propose a new hyperbolic embedding ap-

*Work partially done during an internship at Google.
†Work done while at Google AI.

proach that captures such patterns to achieve the best of both worlds. Our proposed approach produces the parsimonious representations offered by hyperbolic space, especially suitable for hierarchical relations, and is effective even with low-dimensional embeddings. It also uses rich transformations to encode logical patterns in KGs, previously only defined in Euclidean space. To accomplish this, we (1) train hyperbolic embeddings with relation-specific curvatures to preserve multiple hierarchies in KGs; (2) parameterize hyperbolic isometries (distance-preserving operations) and leverage their geometric properties to capture relations' logical patterns, such as symmetry or anti-symmetry; (3) and use a notion of hyperbolic attention to combine geometric operators and capture multiple logical patterns.

We evaluate the performance of our approach, ATTH, on the KG link prediction task using the standard WN18RR (Dettmers et al., 2018; Bordes et al., 2013), FB15k-237 (Toutanova and Chen, 2015) and YAGO3-10 (Mahdisoltani et al., 2013) benchmarks. (1) In low (32) dimensions, we improve over Euclidean-based models by up to 6.1% in the mean reciprocical rank (MRR) metric. In particular, we find that hierarchical relationships, such as WordNet's *hypernym* and *member meronym*, significantly benefit from hyperbolic space; we observe a 16% to 24% relative improvement versus Euclidean baselines. (2) We find that geometric properties of hyperbolic isometries directly map to logical properties of relationships. We study symmetric and anti-symmetric patterns and find that reflections capture symmetric relations while rotations capture anti-symmetry. (3) We show that attention based-transformations have the ability to generalize to multiple logical patterns. For instance, we observe that ATTH recovers reflections for symmetric relations and rotations for the anti-symmetric ones.

In high (500) dimensions, we find that both hyperbolic and Euclidean embeddings achieve similar performance, and our approach achieves new state-of-the-art results (SotA), obtaining 49.6% MRR on WN18RR and 57.7% YAGO3-10. Our experiments show that trainable curvature is critical to generalize hyperbolic embedding methods to high-dimensions. Finally, we visualize embeddings learned in hyperbolic spaces and show that hyperbolic geometry effectively preserves hierarchies in KGs.

2 Related Work

Previous methods for KG embeddings also rely on geometric properties. Improvements have been obtained by exploiting either more sophisticated spaces (e.g., going from Euclidean to complex or hyperbolic space) or more sophisticated operations (e.g., from translations to isometries, or to learning graph neural networks). In contrast, our approach takes a step forward in both directions.

Euclidean embeddings In the past decade, there has been a rich literature on Euclidean embeddings for KG representation learning. These include translation approaches (Bordes et al., 2013; Ji et al., 2015; Wang et al., 2014; Lin et al., 2015) or tensor factorization methods such as RESCAL (Nickel et al., 2011) or DistMult (Yang et al., 2015). While these methods are fairly simple and have few parameters, they fail to encode important logical properties (e.g., translations can't encode symmetry).

Complex embeddings Recently, there has been interest in learning embeddings in complex space, as in the ComplEx (Trouillon et al., 2016) and RotatE (Sun et al., 2019) models. RotatE learns rotations in complex space, which are very effective in capturing logical properties such as symmetry, anti-symmetry, composition or inversion. The recent QuatE model (Zhang et al., 2019) learns KG embeddings using quaternions. However, a downside is that these embeddings require very high-dimensional spaces, leading to high memory costs.

Deep neural networks Another family of methods uses neural networks to produce KG embeddings. For instance, R-GCN (Schlichtkrull et al., 2018) extends graph neural networks to the multi-relational setting by adding a relation-specific aggregation step. ConvE and ConvKB (Dettmers et al., 2018; Nguyen et al., 2018) leverage the expressiveness of convolutional neural networks to learn entity embeddings and relation embeddings. More recently, the KBGAT (Nathani et al., 2019) and A2N (Bansal et al., 2019) models use graph attention networks for knowledge graph embeddings. A downside of these methods is that they are computationally expensive as they usually require pre-trained KG embeddings as input for the neural network.

Hyperbolic embeddings To the best of our knowledge, MuRP (Balažević et al., 2019) is the

only method that learns KG embeddings in hyperbolic space in order to target hierarchical data. MuRP minimizes hyperbolic distances between a re-scaled version of the head entity embedding and a translation of the tail entity embedding. It achieves promising results using hyperbolic embeddings with fewer dimensions than its Euclidean analogues. However, MuRP is a translation model and fails to encode some logical properties of relationships. Furthermore, embeddings are learned in a hyperbolic space with fixed curvature, potentially leading to insufficient precision, and training relies on cumbersome Riemannian optimization. Instead, our proposed method leverages expressive hyperbolic isometries to simultaneously capture logical patterns and hierarchies. Furthermore, embeddings are learned using tangent space (i.e., Euclidean) optimization methods and trainable hyperbolic curvatures per relationship, avoiding precision errors that might arise when using a fixed curvature, and providing flexibility to encode multiple hierarchies.

3 Problem Formulation and Background

We describe the KG embedding problem setting and give some necessary background on hyperbolic geometry.

3.1 Knowledge graph embeddings

In the KG embedding problem, we are given a set of triples $(h, r, t) \in \mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$, where $\mathcal{V}$ and $\mathcal{R}$ are entity and relationship sets, respectively. The goal is to map entities $v \in \mathcal{V}$ to embeddings $\mathbf{e}_v \in \mathcal{U}^{d_\mathcal{V}}$ and relationships $r \in \mathcal{R}$ to embeddings $\mathbf{r}_r \in \mathcal{U}^{d_\mathcal{R}}$, for some choice of space $\mathcal{U}$ (traditionally $\mathbb{R}$), such that the KG structure is preserved.

Concretely, the data is split into $\mathcal{E}_{Train}$ and $\mathcal{E}_{Test}$ triples. Embeddings are learned by optimizing a scoring function $s : \mathcal{V} \times \mathcal{R} \times \mathcal{V} \to \mathbb{R}$, which measures triples' likelihoods. $s(\cdot, \cdot, \cdot)$ is trained using triples in $\mathcal{E}_{Train}$ and the learned embeddings are then used to predict scores for triples in $\mathcal{E}_{Test}$. The goal is to learn embeddings such that the scores of triples in $\mathcal{E}_{Test}$ are high compared to triples that are not present in $\mathcal{E}$.

3.2 Hyperbolic geometry

We briefly review key notions from hyperbolic geometry; a more in-depth treatment is available in standard texts (Robbin and Salamon). Hyperbolic geometry is a non-Euclidean geometry with constant negative curvature. In this work, we use the d-dimensional Poincaré ball model with negative curvature $-c$ ($c > 0$): $\mathbb{B}^{d,c} = \{\mathbf{x} \in \mathbb{R}^d : ||\mathbf{x}||^2 < \frac{1}{c}\}$, where $||\cdot||$ denotes the L_2 norm. For each point $\mathbf{x} \in \mathbb{B}^{d,c}$, the tangent space $\mathcal{T}^c_\mathbf{x}$ is a d-dimensional vector space containing all possible directions of paths in $\mathbb{B}^{d,c}$ leaving from $\mathbf{x}$.

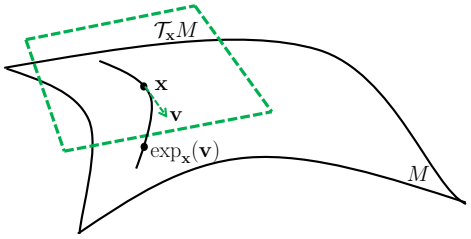


Figure 2: An illustration of the exponential map $\exp_\mathbf{x}(\mathbf{v})$, which maps the tangent space $\mathcal{T}_\mathbf{x}M$ at the point $\mathbf{x}$ to the hyperbolic manifold M.

The tangent space $\mathcal{T}^c_\mathbf{x}$ maps to $\mathbb{B}^{d,c}$ via the exponential map (Figure 2), and conversely, the logarithmic map maps $\mathbb{B}^{d,c}$ to $\mathcal{T}^c_\mathbf{x}$. In particular, we have closed-form expressions for these maps at the origin:

$$\exp^c_\mathbf{0}(\mathbf{v}) = \tanh(\sqrt{c}||\mathbf{v}||)\frac{\mathbf{v}}{\sqrt{c}||\mathbf{v}||}, \quad (1)$$

$$\log^c_\mathbf{0}(\mathbf{y}) = \operatorname{arctanh}(\sqrt{c}||\mathbf{y}||)\frac{\mathbf{y}}{\sqrt{c}||\mathbf{y}||}. \quad (2)$$

Vector addition is not well-defined in the hyperbolic space (adding two points in the Poincaré ball might result in a point outside the ball). Instead, Möbius addition $\oplus^c$ (Ganea et al., 2018) provides an analogue to Euclidean addition for hyperbolic space. We give its closed-form expression in Appendix A.1. Finally, the hyperbolic distance on $\mathbb{B}^{d,c}$ has the explicit formula:

$$d^c(\mathbf{x}, \mathbf{y}) = \frac{2}{\sqrt{c}}\operatorname{arctanh}(\sqrt{c}||-\mathbf{x} \oplus^c \mathbf{y}||). \quad (3)$$

4 Methodology

The goal of this work is to learn parsimonious hyperbolic embeddings that can encode complex logical patterns such as symmetry, anti-symmetry, or inversion while preserving latent hierarchies. Our model, ATTH, (1) learns KG embeddings in hyperbolic space in order to preserve hierarchies (Section 4.1), (2) uses a class of hyperbolic isometries parameterized by compositions of Givens transformations to encode logical patterns (Section 4.2), (3) combines these isometries with hyperbolic attention (Section 4.3). We describe the full model in Section 4.4.

4.1 Hierarchies in hyperbolic space

As described, hyperbolic embeddings enable us to represent hierarchies even when we limit ourselves to low-dimensional spaces. In fact, two-dimensional hyperbolic space can represent any tree with arbitrarily small error (Sala et al., 2018).

It is important to set the curvature of the hyperbolic space correctly. This parameter provides flexibility to the model, as it determines whether to embed relations into a more curved hyperbolic space (more "tree-like"), or into a flatter, more "Euclidean-like" geometry. For each relation, we learn a relation-specific absolute curvature c_r, enabling us to represent a variety of hierarchies. As we show in Section 5.5, fixing, rather than learning curvatures can lead to significant performance degradation.

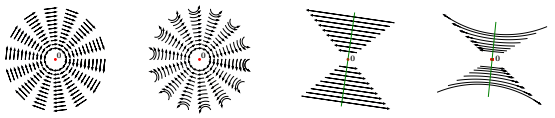
4.2 Hyperbolic isometries

Relationships often satisfy particular properties, such as symmetry: e.g., if (*Michelle Obama*, *married to*, *Barack Obama*) holds, then (*Barack Obama*, *married to*, *Michelle Obama*) does as well. These rules are not universal. For instance, (*Barack Obama*, *born in*, *Hawaii*) is not symmetric.

Creating and curating a set of deterministic rules is infeasible for large-scale KGs; instead, embedding methods represent relations as parameterized geometric operations that directly map to logical properties. We use two such operations in hyperbolic space: *rotations*, which effectively capture compositions or anti-symmetric patterns, and *reflections*, which naturally encode symmetric patterns.

Rotations Rotations have been successfully used to encode compositions in complex space with the RotatE model (Sun et al., 2019); we lift these to hyperbolic space. Compared to translations or tensor factorization approaches which can only infer some logical patterns, rotations can simultaneously model and infer *inversion*, *composition*, *symmetric* or *anti-symmetric* patterns.

Reflections These isometries reflect along a fixed subspace. While some rotations can represent symmetric relations (more specifically $\pi-$rotations), any reflection can naturally represent symmetric relations, since their second power is the identity. They provide a way to fill-in missing entries in symmetric triples, by applying the same operation to both the tail and the head entity. For instance, by modelling *sibling of* with a reflection, we can directly infer (*Bob*, *sibling of*, *Alice*) from (*Alice*, *sibling of*, *Bob*) and vice versa.



(a) Rotations (b) Reflections

Figure 3: Euclidean (left) and hyperbolic (right) isometries. In hyperbolic space, the distance between start and end points after applying rotations or reflections is much larger than the Euclidean distance; it approaches the sum of the distances between the points and the origin, giving more "room" to separate embeddings. This is similar to trees, where the shortest path between two points goes through their nearest common ancestor.

Parameterization Unlike RotatE which models rotations via unitary complex numbers, we learn relationship-specific isometries using Givens transformations, 2×2 matrices commonly used in numerical linear algebra. Let $\Theta_r \coloneqq (\theta_{r,i})_{i \in \{1,\dots \frac{d}{2}\}}$ and $\Phi_r \coloneqq (\phi_{r,i})_{i \in \{1,\dots \frac{d}{2}\}}$ denote relation-specific parameters. Using an even number of dimensions d, our model parameterizes rotations and reflections with block-diagonal matrices of the form:

$$\mathrm{Rot}(\Theta_r) = \mathrm{diag}(G^+(\theta_{r,1}), \dots, G^+(\theta_{r,\frac{d}{2}})), \quad (4)$$

$$\mathrm{Ref}(\Phi_r) = \mathrm{diag}(G^-(\phi_{r,1}), \dots, G^-(\phi_{r,\frac{n}{2}})), \quad (5)$$

$$\text{where } G^{\pm}(\theta) \coloneqq \begin{bmatrix} \cos(\theta) & \mp\sin(\theta) \\ \sin(\theta) & \pm\cos(\theta) \end{bmatrix}. \quad (6)$$

Rotations and reflections of this form are hyperbolic isometries (distance-preserving). We can therefore directly apply them to hyperbolic embeddings while preserving the underlying geometry. Additionally, these transformations are computationally efficient and can be computed in linear time in the dimension. We illustrate two-dimensional isometries in both Euclidean and hyperbolic spaces in Figure 3.

4.3 Hyperbolic attention

Of our two classes of hyperbolic isometries, one or the other may better represent a particular relation. To handle this, we use an attention mechanism to learn the right isometry. Thus we can represent symmetric, anti-symmetric or mixed-behaviour relations (i.e. neither symmetric nor anti-symmetric) as a combination of rotations and reflections.

Let $\mathbf{x}^H$ and $\mathbf{y}^H$ be hyperbolic points (e.g., reflection and rotation embeddings), and $\mathbf{a}$ be an

attention vector. Our approach maps hyperbolic representations to tangent space representations, $\mathbf{x}^E = \log_{\mathbf{0}}^c(\mathbf{x}^H)$ and $\mathbf{y}^E = \log_{\mathbf{0}}^c(\mathbf{y}^H)$, and computes attention scores:

$$(\alpha_{\mathbf{x}}, \alpha_{\mathbf{y}}) = \text{Softmax}(\mathbf{a}^T\mathbf{x}^E, \mathbf{a}^T\mathbf{y}^E).$$

We then compute a weighted average using the recently proposed tangent space average (Chami et al., 2019; Liu et al., 2019):

$$\text{Att}(\mathbf{x}^H, \mathbf{y}^H; \mathbf{a}) \coloneqq \exp_{\mathbf{0}}^c(\alpha_{\mathbf{x}}\mathbf{x}^E + \alpha_{\mathbf{y}}\mathbf{y}^E). \quad (7)$$

4.4 The ATTH model

We have all of the building blocks for ATTH, and can now describe the model architecture. Let $(\mathbf{e}_v^H)_{v\in\mathcal{V}}$ and $(\mathbf{r}_r^H)_{r\in\mathcal{R}}$ denote entity and relationship hyperbolic embeddings respectively. For a triple $(h, r, t) \in \mathcal{V} \times \mathcal{R} \times \mathcal{V}$, ATTH applies relation-specific rotations (Equation 4) and reflections (Equation 5) to the head embedding:

$$\mathbf{q}_{\text{Rot}}^H = \text{Rot}(\Theta_r)\mathbf{e}_h^H, \ \mathbf{q}_{\text{ref}}^H = \text{Ref}(\Phi_r)\mathbf{e}_h^H. \quad (8)$$

ATTH then combines the two representations using hyperbolic attention (Equation 7) and applies a hyperbolic translation:

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

Intuitively, rotations and reflections encode logical patterns while translations capture tree-like structures by moving between levels of the hierarchy. Finally, query embeddings are compared to target tail embeddings via the hyperbolic distance (Equation 3). The resulting scoring function is:

$$s(h, r, t) = -d^{c_r}(Q(h, r), \mathbf{e}_t^H)^2 + b_h + b_t, \quad (10)$$

where $(b_v)_{v\in\mathcal{V}}$ are entity biases which act as margins in the scoring function (Tifrea et al., 2019; Balažević et al., 2019).

The model parameters are then $\{(\Theta_r, \Phi_r, \mathbf{r}_r^H, \mathbf{a}_r, c_r)_{r\in\mathcal{R}}, (\mathbf{e}_v^H, b_v)_{v\in\mathcal{V}}\}$. Note that the total number of parameters in ATTH is $\mathcal{O}(|\mathcal{V}|d)$, similar to traditional models that do not use attention or geometric operations. The extra cost is proportional to the number of relations, which is usually much smaller than the number of entities.

Dataset	#entities	#relations	#triples	ξ_G
WN18RR	41k	11	93k	-2.54
FB15k-237	15k	237	310k	-0.65
YAGO3-10	123k	37	1M	-0.54

Table 1: Datasets statistics. The lower the metric ξ_G is, the more tree-like the knowledge graph is.

5 Experiments

In low dimensions, we hypothesize (1) that hyperbolic embedding methods obtain better representations and allow for improved downstream performance for hierarchical data (Section 5.2). (2) We expect the performance of relation-specific geometric operations to vary based on the relation's logical patterns (Section 5.3). (3) In cases where the relations are neither purely symmetric nor antisymmetric, we anticipate that hyperbolic attention outperforms the models which are based on solely reflections or rotations (Section 5.4). Finally, in high dimensions, we expect hyperbolic models with trainable curvature to learn the best geometry, and perform similarly to their Euclidean analogues (Section 5.5).

5.1 Experimental setup

Datasets We evaluate our approach on the link prediction task using three standard competition benchmarks, namely WN18RR (Bordes et al., 2013; Dettmers et al., 2018), FB15k-237 (Bordes et al., 2013; Toutanova and Chen, 2015) and YAGO3-10 (Mahdisoltani et al., 2013). WN18RR is a subset of WordNet containing 11 lexical relationships between 40,943 word senses, and has a natural hierarchical structure, e.g., (*car*, *hypernym of*, *sedan*). FB15k-237 is a subset of Freebase, a collaborative KB of general world knowledge. FB15k-237 has 14,541 entities and 237 relationships, some of which are non-hierarchical, such as *born-in* or *nationality*, while others have natural hierarchies, such as *part-of* (for organizations). YAGO3-10 is a subset of YAGO3, containing 123,182 entities and 37 relations, where most relations provide descriptions of people. Some relationships have a hierarchical structure such as *playsFor* or *actedIn*, while others induce logical patterns, like *isMarriedTo*.

For each KG, we follow the standard data augmentation protocol by adding inverse relations (Lacroix et al., 2018) to the datasets. Additionally, we estimate the global graph curvature ξ_G (Gu et al., 2019) (see Appendix A.2 for more details),

$\mathcal{U}$	Model	WN18RR MRR	WN18RR H@1	WN18RR H@3	WN18RR H@10	FB15k-237 MRR	FB15k-237 H@1	FB15k-237 H@3	FB15k-237 H@10	YAGO3-10 MRR	YAGO3-10 H@1	YAGO3-10 H@3	YAGO3-10 H@10
$\mathbb{R}^d$	RotatE	.387	.330	.417	.491	.290	.208	.316	.458	-	-	-	-
	MuRE	.458	.421	.471	.525	.313	.226	.340	.489	.283	.187	.317	.478
$\mathbb{C}^d$	ComplEx-N3	.420	.390	.420	.460	.294	.211	.322	.463	.336	.259	.367	.484
$\mathbb{B}^{d,1}$	MuRP	.465	.420	.484	.544	.323	.235	.353	**.501**	.230	.150	.247	.392
$\mathbb{R}^d$	REFE	.455	.419	.470	.521	.302	.216	.330	.474	.370	.289	.403	.527
	ROTE	.463	.426	.477	.529	.307	.220	.337	.482	.381	.295	.417	.548
	ATTE	.456	.419	.471	.526	.311	.223	.339	.488	.374	.290	.410	.537
$\mathbb{B}^{d,c}$	REFH	.447	.408	.464	.518	.312	.224	.342	.489	.381	.302	.415	.530
	ROTH	**.472**	**.428**	**.490**	**.553**	.314	.223	.346	.497	.393	.307	.435	559
	ATTH	.466	.419	.484	.551	**.324**	**.236**	**.354**	**.501**	**.397**	**.310**	**.437**	**.566**

Table 2: Link prediction results for low-dimensional embeddings ($d = 32$) in the filtered setting. Best score in **bold** and best published underlined. Hyperbolic isometries significantly outperform Euclidean baselines on WN18RR and YAGO3-10, both of which exhibit hierarchical structures.

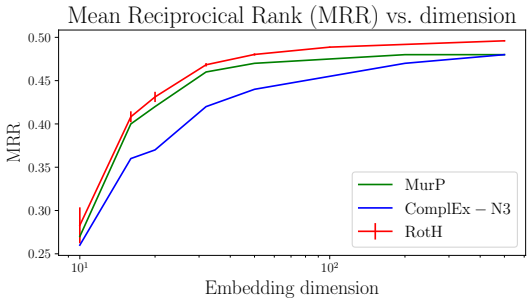


Figure 4: WN18RR MRR dimension for $d \in \{10, 16, 20, 32, 50, 200, 500\}$. Average and standard deviation computed over 10 runs for ROTH.

which is a distance-based measure of how close a given graph is to being a tree. We summarize the datasets' statistics in Table 1.

Baselines We compare our method to SotA models, including MurP (Balazevic et al., 2019), MurE (which is the Euclidean analogue or MurP), RotatE (Sun et al., 2019), ComplEx-N3 (Lacroix et al., 2018) and TuckER (Balazevic et al., 2019). Baseline numbers in high dimensions (Table 5) are taken from the original papers, while baseline numbers in the low-dimensional setting (Table 2) are computed using open-source implementations of each model. In particular, we run hyper-parameter searches over the same parameters as the ones in the original papers to compute baseline numbers in the low-dimensional setting.

Ablations To analyze the benefits of hyperbolic geometry, we evaluate the performance of ATTE, which is equivalent to ATTH with curvatures set to zero. Additionally, to better understand the role of attention, we report scores for variants of ATTE/H using only rotations (ROTE/H) or reflections (REFE/H).

Evaluation metrics At test time, we use the scoring function in Equation 10 to rank the correct tail or head entity against all possible entities, and use in use inverse relations for head prediction (Lacroix et al., 2018). Similar to previous work, we compute two ranking-based metrics: (1) mean reciprocal rank (MRR), which measures the mean of inverse ranks assigned to correct entities, and (2) hits at K (H@K, $K \in \{1, 3, 10\}$), which measures the proportion of correct triples among the top K predicted triples. We follow the standard evaluation protocol in the filtered setting (Bordes et al., 2013): all true triples in the KG are filtered out during evaluation, since predicting a low rank for these triples should not be penalized.

Training procedure and implementation We train ATTH by minimizing the full cross-entropy loss with uniform negative sampling, where negative examples for a triple (h, r, t) are sampled uniformly from all possible triples obtained by perturbing the tail entity:

$$\mathcal{L} = \sum_{t' \sim \mathcal{U}(\mathcal{V})} \log(1+\exp(y_{t'} s(h, r, t'))), \quad (11)$$

$$\text{where } y_{t'} = \begin{cases} -1 & \text{if } t' = t \\ 1 & \text{otherwise.} \end{cases}$$

Since optimization in hyperbolic space is practically challenging, we instead define all parameters in the tangent space at the origin, optimize embeddings using standard Euclidean techniques, and use the exponential map to recover the hyperbolic parameters (Chami et al., 2019). We provide more details on tangent space optimization in Appendix A.4. We conducted a grid search to select the learning rate, optimizer, negative sample size, and batch size, using the validation set to select the best hy-

Relation	Khs_G	ξ_G	RotE	RotH	Improvement
member meronym	1.00	-2.90	.320	**.399**	24.7%
hypernym	1.00	-2.46	.237	**.276**	16.5%
has part	1.00	-1.43	.291	**.346**	18.9%
instance hypernym	1.00	-0.82	.488	**.520**	6.56%
member of domain region	1.00	-0.78	**.385**	.365	-5.19%
member of domain usage	1.00	-0.74	**.458**	.438	-4.37%
synset domain topic of	0.99	-0.69	.425	**.447**	5.17%
also see	0.36	-2.09	.634	**.705**	11.2%
derivationally related form	0.07	-3.84	.960	**.968**	0.83%
similar to	0.07	-1.00	**1.00**	**1.00**	0.00%
verb group	0.07	-0.50	**.974**	**.974**	0.00%

Table 3: Comparison of H@10 for WN18RR relations. Higher Khs_G and lower ξ_G means more hierarchical.

Relation	Anti-symmetric	Symmetric	RotH	RefH	AttH
hasNeighbor	✗	✓	.750	**1.00**	**1.00**
isMarriedTo	✗	✓	.941	.941	**1.00**
actedIn	✓	✗	.145	.110	**.150**
hasMusicalRole	✓	✗	.431	.375	**.458**
directed	✓	✗	.500	.450	**.567**
graduatedFrom	✓	✗	.262	.167	**.274**
playsFor	✓	✗	**.671**	.642	.664
wroteMusicFor	✓	✗	**.281**	.188	.266
hasCapital	✓	✗	.692	**.731**	**.731**
dealsWith	✗	✗	.286	.286	**.429**
isLocatedIn	✗	✗	.404	.399	**.420**

Table 4: Comparison of geometric transformations on a subset of YAGO3-10 relations.

perparameters. Our best model hyperparameters are detailed in Appendix A.3. We conducted all our experiments on NVIDIA Tesla P100 GPUs and make our implementation publicly available*.

5.2 Results in low dimensions

We first evaluate our approach in the low-dimensional setting for $d = 32$, which is approximately one order of magnitude smaller than SotA Euclidean methods. Table 2 compares the performance of AttH to that of other baselines, including the recent hyperbolic (but not rotation-based) MuRP model. In low dimensions, hyperbolic embeddings offer much better representations for hierarchical relations, confirming our hypothesis. AttH improves over previous Euclidean and hyperbolic methods by 0.7% and 6.1% points in MRR on WN18RR and YAGO3-10 respectively. Both datasets have multiple hierarchical relationships, suggesting that the hierarchical structure imposed by hyperbolic geometry leads to better embeddings. On FB15k-237, AttH and MurP achieve similar performance, both improving over Euclidean baselines. We conjecture that translations are sufficient to model relational patterns in FB15k-237.

To understand the role of dimensionality, we also conduct experiments on WN18RR against SotA methods under varied low-dimensional settings (Figure 4). We include error bars for our method with average MRR and standard deviation computed over 10 runs. Our approach consistently outperforms all baselines, suggesting that hyperbolic embeddings still attain high-accuracy across a broad range of dimensions.

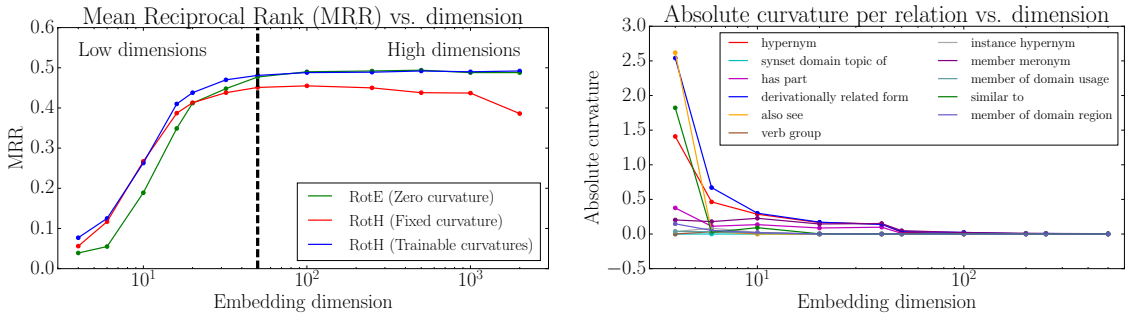
Additionally, we measure performance per relation on WN18RR in Table 3 to understand the benefits of hyperbolic geometric on hierarchical relations. We report the Krackhardt hierarchy score (Khs_G) (Balažević et al., 2019) and estimated curvature per relation (see Appendix A.2 for more details). We consider a relation to be hierarchical when its corresponding graph is close to tree-like (low curvature, high Khs_G). We observe that hyperbolic embeddings offer much better performance on hierarchical relations such as *hypernym* or *has part*, while Euclidean and hyperbolic embeddings have similar performance on non-hierarchical relations such as *verb group*. We also plot the learned curvature per relation versus the embedding dimension in Figure 5b. We note that the learned curvature in low dimensions directly correlates with the estimated graph curvature ξ_G in Table 3, suggesting that the model with learned curvatures learns more "curved" embedding spaces for tree-like relations.

Finally, we observe that MurP achieves lower performance than MurE on YAGO3-10, while AttH improves over AttE by 2.3% in MRR. This suggests that trainable curvature is critical to learn embeddings with the right amount of curvature, while fixed curvature might degrade performance. We elaborate further on this point in Section 5.5.

5.3 Hyperbolic rotations and reflections

In our experiments, we find that rotations work well on WN18RR, which contains multiple hierarchical and anti-symmetric relations, while reflections work better for YAGO3-10 (Table 5). To better understand the mechanisms behind these observations, we analyze two specific patterns: relation symmetry and anti-symmetry. We report performance per-relation on a subset of YAGO3-10 relations in Table 4. We categorize relations into symmetric, anti-symmetric, or neither symmetric nor anti-symmetric categories using data statistics. More concretely, we consider a relation to satisfy a logical pattern when the logical condition is satisfied by most of the triplets (e.g., a relation r is symmetric if for most KG triples (h, r, t), (t, r, h) is also in the KG). We observe that reflections encode

*Code available at https://github.com/tensorflow/neural-structured-learning/tree/master/research/kg_hyp_emb



(a) MRR for fixed and trainable curvatures on WN18RR.

(b) Curvatures learned by with RoTH on WN18RR.

Figure 5: (a): RoTH offers improved performance in low dimensions; in high dimensions, fixed curvature degrades performance, while trainable curvature approximately recovers Euclidean space. (b): As the dimension increases, the learned curvature of hierarchical relationships tends to zero.

symmetric relations particularly well, while rotations are well suited for anti-symmetric relations. This confirms our intuition—and the motivation for our approach—that particular geometric properties capture different kinds of logical properties.

5.4 Attention-based transformations

One advantage of using relation-specific transformations is that each relation can learn the right geometric operators based on the logical properties it has to satisfy. In particular, we observe that in both low- and high-dimensional settings, attention-based models can recover the performance of the best transformation on all datasets (Tables 2 and 5). Additionally, per-relationship results on YAGO3-10 in Table 4 suggest that AttH indeed recovers the best geometric operation.

Furthermore, for relations that are neither symmetric nor anti-symmetric, we find that AttH can outperform rotations and reflections, suggesting that combining multiple operators with attention can learn more expressive operators to model mixed logical patterns. In other words, attention-based transformations alleviate the need to conduct experiments with multiple geometric transformations by simply allowing the model to choose which one is best for a given relation.

5.5 Results in high dimensions

In high dimensions (Table 5), we compare against a variety of other models and achieve new SotA results on WN18RR and YAGO3-10, and third-best results on FB15k-237. As we expected, when the embedding dimension is large, Euclidean and hyperbolic embedding methods perform similarly across all datasets. We explain this behavior by noting that when the dimension is sufficiently large, both Euclidean and hyperbolic spaces have enough capacity to represent complex hierarchies in KGs. This is further supported by Figure 5b, which shows the learned absolute curvature versus the dimension. We observe that curvatures are close to zero in high dimensions, confirming our expectation that RoTH with trainable curvatures learns a roughly Euclidean geometry in this setting.

In contrast, fixed curvature degrades performance in high dimensions (Figure 5a), confirming the importance of trainable curvatures and its impact on precision and capacity (previously studied by (Sala et al., 2018)). Additionally, we show the embeddings' norms distribution in the Appendix (Figure 7). Fixed curvature results in embeddings being clustered near the boundary of the ball while trainable curvatures adjusts the embedding space to better distribute points throughout the ball. Precision issues that might arise with fixed curvature could also explain MurP's low performance in high dimensions. Trainable curvatures allow RoTH to perform as well or better than previous methods in both low and high dimensions.

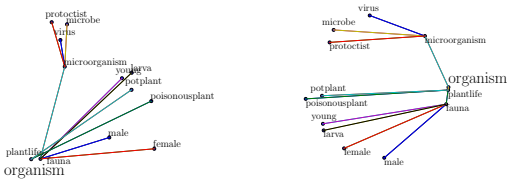
5.6 Visualizations

In Figure 6, we visualize the embeddings learned by RoTE versus RoTH for a sub-tree of the *organism* entity in WN18RR. To better visualize the hierarchy, we apply k inverse rotations for all nodes at level k in the tree.

By contrast to RoTE, RoTH preserves the tree structure in the embedding space. Furthermore, we note that RoTE cannot simultaneously preserve the tree structure and make non-neighboring nodes far from each other. For instance, *virus* should be far from *male*, but preserving the tree structure (by going one level down in the tree) while making

$\mathcal{U}$	Model	WN18RR MRR	WN18RR H@1	WN18RR H@3	WN18RR H@10	FB15k-237 MRR	FB15k-237 H@1	FB15k-237 H@3	FB15k-237 H@10	YAGO3-10 MRR	YAGO3-10 H@1	YAGO3-10 H@3	YAGO3-10 H@10
$\mathbb{R}^d$	DistMult	.430	.390	.440	.490	.241	.155	.263	.419	.340	.240	.380	.540
	ConvE	.430	.400	.440	.520	.325	.237	.356	.501	.440	.350	.490	.620
	TuckER	.470	.443	.482	.526	.358	.266	.394	.544	-	-	-	-
	MurE	.475	.436	.487	.554	.336	.245	.370	.521	.532	.444	.584	.694
$\mathbb{C}^d$	ComplEx-N3	.480	.435	.495	.572	.357	.264	.392	.547	.569	.498	.609	.701
	RotatE	.476	.428	.492	.571	.338	.241	.375	.533	.495	.402	.550	.670
$\mathbb{H}^d$	Quaternion	.488	.438	.508	.582	.348	.248	.382	**.550**	-	-	-	-
$\mathbb{B}^{d,1}$	MurP	.481	.440	.495	.566	.335	.243	.367	.518	.354	.249	.400	567
$\mathbb{R}^d$	REFE	.473	.430	.485	.561	.351	.256	.390	.541	**.577**	**.503**	**.621**	**.712**
	ROTE	.494	.446	.512	.585	.346	.251	.381	.538	.574	.498	**.621**	.711
	ATTE	.490	.443	.508	.581	.351	.255	.386	.543	.575	.500	**.621**	.709
$\mathbb{B}^{d,c}$	REFH	.461	.404	.485	.568	.346	.252	.383	.536	.576	.502	.619	.711
	ROTH	**.496**	**.449**	**.514**	**.586**	.344	.246	.380	.535	.570	.495	.612	.706
	ATTH	.486	.443	.499	.573	.348	.252	.384	.540	.568	.493	.612	.702

Table 5: Link prediction results for high-dimensional embeddings (best for $d \in \{200, 400, 500\}$) in the filtered setting. DistMult, ConvE and ComplEx results are taken from (Dettmers et al., 2018). Best score in **bold** and best published underlined. ATTE and ATTH have similar performance in the high-dimensional setting, performing competitively with or better than state-of-the-art methods on WN18RR, FB15k-237 and YAGO3-10.



(a) ROTE embeddings. (b) ROTH embeddings.

Figure 6: Visualizations of the embeddings learned by ROTE and ROTH on a sub-tree of WN18RR for the *hypernym* relation. In contrast to ROTE, ROTH preserves hierarchies by learning tree-like embeddings.

these two nodes far from each other is difficult in Euclidean space. In hyperbolic space, however, we observe that going one level down in the tree is achieved by translating embeddings towards the left. This pattern essentially illustrates the translation component in ROTH, allowing the model to simultaneously preserve hierarchies while making non-neighbouring nodes far from each other.

6 Conclusion

We introduce ATTH, a hyperbolic KG embedding model that leverages the expressiveness of hyperbolic space and attention-based geometric transformations to learn improved KG representations in low-dimensions. ATTH learns embeddings with trainable hyperbolic curvatures, allowing it to learn the right geometry for each relationship and generalize across multiple embedding dimensions. ATTH achieves new SotA on WN18RR and YAGO3-10, real-world KGs which exhibit hierarchical structures. Future directions for this work include exploring other tasks that might benefit from hyperbolic geometry, such as hypernym detection. The proposed attention-based transformations can also be extended to other geometric operations.

Acknowledgements

We thank Avner May for their helpful feedback and discussions. We gratefully acknowledge the support of DARPA under Nos. FA86501827865 (SDH) and FA86501827882 (ASED); NIH under No. U54EB020405 (Mobilize), NSF under Nos. CCF1763315 (Beyond Sparsity), CCF1563078 (Volume to Velocity), and 1937301 (RTML); ONR under No. N000141712266 (Unifying Weak Supervision); the Moore Foundation, NXP, Xilinx, LETI-CEA, Intel, IBM, Microsoft, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, the Okawa Foundation, American Family Insurance, Google Cloud, Swiss Re, the HAI-AWS Cloud Credits for Research program, TOTAL, and members of the Stanford DAWN project: Teradata, Facebook, Google, Ant Financial, NEC, VMWare, and Infosys. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of DARPA, NIH, ONR, or the U.S. Government.

References

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

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5188–5197.

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*, pages 4387–4392.

Silvere Bonnabel. 2013. Stochastic gradient descent on Riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229.

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*, pages 2787–2795.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 4869–4880.

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

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*.

Albert Gu, Fred Sala, Beliz Gunel, and Christopher Ré. 2019. Learning mixed-curvature representations in product spaces. In *International Conference on Learning Representations*.

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)*, pages 687–696.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference for Learning Representations*.

David Krackhardt. 1994. Graph theoretical dimensions of informal organizations. In *Computational organization theory*, pages 107–130. Psychology Press.

Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. *International Conference on Machine Learning*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI Conference on Artificial Intelligence*.

Qi Liu, Maximilian Nickel, and Douwe Kiela. 2019. Hyperbolic graph neural networks. In *Advances in Neural Information Processing Systems*, pages 8228–8239.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. 2013. Yago3: A knowledge base from multilingual wikipedias.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

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*. Association for Computational Linguistics.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 327–333.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *International Conference on Machine Learning*, pages 809–816. Omnipress.

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

Joel W Robbin and Dietmar A Salamon. Introduction to differential geometry.

Frederic Sala, Chris De Sa, Albert Gu, and Christopher Ré. 2018. Representation tradeoffs for hyperbolic embeddings. In *International Conference on Machine Learning*, pages 4457–4466.

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*, pages 593–607. Springer.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

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*.

Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. 2019. Poincaré GloVe: Hyperbolic word embeddings. In *International Conference on Learning Representations*.

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*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080.

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*.

Canran Xu and Ruijiang Li. 2019. Relation embedding with dihedral group in knowledge graph. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations*.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems*, pages 2731–2741.

A Appendix

Below, we provide additional details. We start by providing the formula for the hyperbolic analogue of addition that we use, along with additional hyperbolic geometry background. Next, we provide more information about the metrics that are used to determine how hierarchical a dataset is. Afterwards, we give additional experimental details, including the table of hyperparameters and further details on tangent space optimization. Lastly, we include an additional comparison against the Dihedral model (Xu and Li, 2019).

A.1 Möbius addition

The Möbius addition operation (Ganea et al., 2018) has the closed-form expression:

$$\begin{aligned} \mathbf{x} \oplus^c \mathbf{y} &= \frac{\alpha_{\mathbf{xy}}\mathbf{x} + \beta_{\mathbf{xy}}\mathbf{y}}{1 + 2c\mathbf{x}^T\mathbf{y} + c^2||\mathbf{x}||^2||\mathbf{y}||^2}, \\ \text{where } \alpha_{\mathbf{xy}} &= 1 + 2c\mathbf{x}^T\mathbf{y} + c||\mathbf{y}||^2, \\ \text{and } \beta_{\mathbf{xy}} &= 1 - c||\mathbf{x}||^2. \end{aligned}$$

In contrast to Euclidean addition, it is neither commutative nor associative. However, it provides an analogue through the lens of parallel transport: given two points $\mathbf{x}, \mathbf{y}$ and a vector $\mathbf{v}$ in $\mathcal{T}^c_{\mathbf{x}}$, there is a unique vector in $\mathcal{T}^c_{\mathbf{y}}$ which creates the same angle as $\mathbf{v}$ with the direction of the geodesic (shortest path) connecting $\mathbf{x}$ to $\mathbf{y}$. This map is the parallel transport $P^c_{\mathbf{x}\to\mathbf{y}}(\cdot)$; Euclidean parallel transport is the standard Euclidean addition. Analogously, the Möbius addition satisfies (Ganea et al., 2018): $\mathbf{x} \oplus^c \mathbf{y} = \exp^c_{\mathbf{x}}(P^c_{\mathbf{0}\to\mathbf{x}}(\log^c_{\mathbf{0}}(\mathbf{y})))$.

A.2 Hierarchy estimates

We use two metrics to estimate how hierarchical a relation is: the curvature estimate ξ_G and the Krackhardt hierarchy score Khs_G. While the curvature estimate captures global hierarchical behaviours (how much the graph is tree-like when zooming-out), the Krackhardt score captures a more local behaviour (how many small loops the graph has). See Figure 8 for examples.

Curvature estimate To estimate the curvature of a relation r, we restrict to the undirected graph G_r spanned by the edges labeled as r. Following (Gu et al., 2019), let $\xi_{G_r}(a, b, c)$ be the curvature estimate of a triangle in G_r with vertices $\{a, b, c\}$,

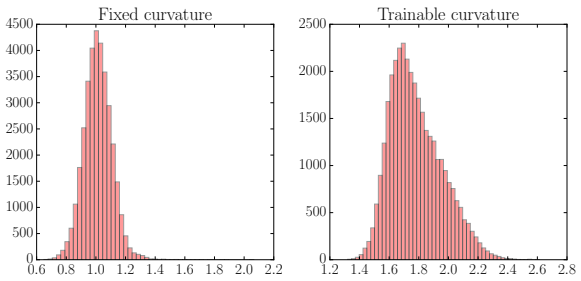


Figure 7: Histogram of embeddings norm learned with fixed and trainable curvatures for the hypernym relation in WN18RR.

which is given by:

$$\begin{aligned} \xi_{G_r}(a, b, c) = \frac{1}{2d_{G_r}(a, m)} \big(&d_{G_r}(a, m)^2 \\ &+ d_{G_r}(b, c)^2/4 \\ &- (d_{G_r}(a, b)^2 + d_{G_r}(a, c)^2)/2\big), \end{aligned}$$

where m is the midpoint of the shortest path connecting b to c. This estimate is positive for triangles in circles, negative for triangles in trees, and zero for triangles in lines. Moreover, for a triangle in a Riemannian manifold M, $\xi_M(a, b, c)$ estimates the sectional curvature of the plane on which the triangle lies (see (Gu et al., 2019) for more details). Let m_r be the total number of connected components in G_r. We sample 1000 $w_{i,r}$ triangles from each connected component $c_{i,r}$ of G_r where $w_{i,r} = \frac{N^3_{i,r}}{\sum_{i=1}^{m_r} N^3_{i,r}}$, and $N_{i,r}$ is the number of nodes in the component $c_{i,r}$. ξ_{G_r} is the mean of the estimated curvatures of the sampled triangles. For the full graph, we take the weighted average of the relation curvatures ξ_{G_r} with respect to the weights $\frac{\sum_{i=1}^{m_r} N^3_{i,r}}{\sum_r \sum_{i=1}^{m_r} N^3_{i,r}}$.

Krackhardt hierarchy score For the directed graph G_r spanned by the relation r, we let R be the adjacency matrix ($R_{i,j} = 1$ if there is an edge from node i to node j and 0 otherwise). Then:

$$\text{Khs}_{G_r} = \frac{\sum_{i,j=1}^{n} R_{i,j}(1 - R_{j,i})}{\sum_{i,j=1}^{n} R_{i,j}}.$$

See (Krackhardt, 1994) for more details. We note that for fully observed symmetric relations (each edge is in a two-edge loop), $\text{Khs}_{G_r} = 0$ while for anti-symmetric relations (no small loops), $\text{Khs}_{G_r} = 1$.

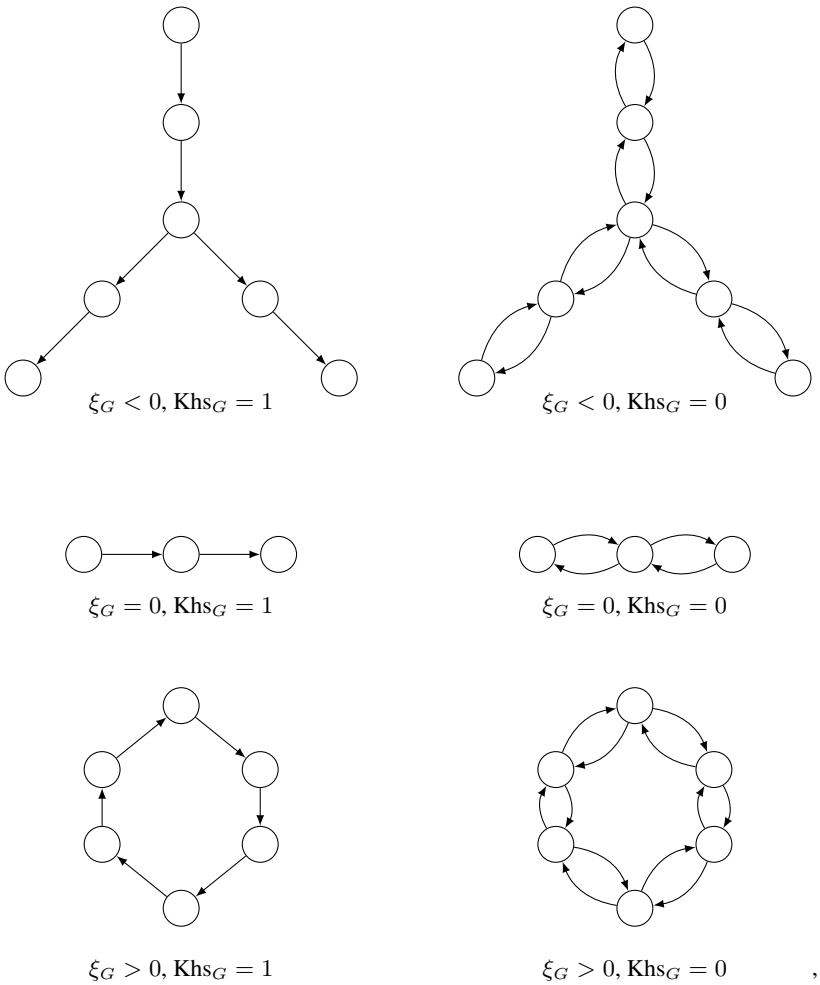


Figure 8: The curvature estimate ξ_G and the Krackhardt hierarchy score Khs_G for several simple graphs. The top-left graph is the most hierarchical, while the bottom-right graph is the least hierarchical.

	WN18RR		FB15k-237		YAGO3-10	
Model	MRR	H@10	MRR	H@10	MRR	H@10
Dihedral	.486	557	.300	.496	.388	.573
ATTE	**.490**	**.581**	**.351**	**.543**	**.575**	**.709**

Table 6: Comparison of Dihedral and ATTE in high-dimensions.

A.3 Experimental details

For all our Euclidean and hyperbolic models, we conduct a hyperparameter search for the learning rate, optimizer (Adam (Kingma and Ba, 2015) or Adagrad (Duchi et al., 2011)), negative sample size and batch size. We train each model for 500 epochs and use early stopping after 100 epochs if the validation MRR stops increasing. We report the best hyperparameters for each dataset in Table 7.

A.4 Tangent space optimization

Optimization in hyperbolic space normally requires Riemannian Stochastic Gradient Descent (RSGD) (Bonnabel, 2013), as was used in MuRP. RSGD is challenging in practice. Instead, we use tangent space optimization (Chami et al., 2019). We define all the ATTH parameters in the tangent space at the origin (our parameter space), optimize embeddings using standard Euclidean techniques, and use the exponential map to recover the hyperbolic parameters.

Note that tangent space optimization is an exact procedure, which does not incur losses in representational power. This is the case in hyperbolic space specifically because of a completeness property: there is always a global bijection between the tangent space and the manifold.

Concretely, ATTH optimizes the entity and relationship embeddings $(\mathbf{e}_v^E)_{v\in\mathcal{V}}$ and $(\mathbf{r}_r^E)_{r\in\mathcal{R}}$, which are mapped to the Poincaré ball with:

$$\mathbf{e}_v^H = \exp_{\mathbf{0}}^{c_r}(\mathbf{e}_v^E) \text{ and } \mathbf{r}_r^H = \exp_{\mathbf{0}}^{c_r}(\mathbf{r}_r^E), \quad (12)$$

The trainable model parameters are then

Dataset	embedding dimension	model	learning rate	optimizer	batch size	negative samples
WN18RR	32	REFE	0.001	Adam	100	250
		ROTE	0.001	Adam	100	250
		ATTE	0.001	Adam	100	250
		REFH	0.0005	Adam	250	250
		ROTH	0.0005	Adam	500	50
		ATTH	0.0005	Adam	500	50
	500	REFE	0.1	Adagrad	500	50
		ROTE	0.001	Adam	100	500
		ATTE	0.001	Adam	1000	50
		REFH	0.05	Adagrad	500	50
		ROTH	0.001	Adam	1000	50
		ATTH	0.001	Adam	1000	50
FB15k-237	32	REFE	0.075	Adagrad	250	250
		ROTE	0.05	Adagrad	500	50
		ATTE	0.05	Adagrad	500	50
		REFH	0.05	Adagrad	500	250
		ROTH	0.1	Adagrad	100	50
		ATTH	0.05	Adagrad	500	100
	500	REFE	0.05	Adagrad	500	50
		ROTE	0.05	Adagrad	100	50
		ATTE	0.05	Adagrad	500	50
		REFH	0.05	Adagrad	500	50
		ROTH	0.05	Adagrad	1000	50
		ATTH	0.05	Adagrad	500	50
YAGO3-10	32	REFE	0.005	Adam	2000	NA
		ROTE	0.005	Adam	2000	NA
		ATTE	0.005	Adam	2000	NA
		REFH	0.005	Adam	1000	NA
		ROTH	0.001	Adam	1000	NA
		ATTH	0.001	Adam	1000	NA
	500	REFE	0.005	Adam	4000	NA
		ROTE	0.005	Adam	4000	NA
		ATTE	0.005	Adam	2000	NA
		REFH	0.001	Adam	1000	NA
		ROTH	0.0005	Adam	1000	NA
		ATTH	0.0005	Adam	1000	NA

Table 7: Best hyperparameters in low- and high-dimensional settings. NA negative samples indicates that the full cross-entropy loss is used, without negative sampling.

$\{(\Theta_r, \Phi_r, \mathbf{r}_r^E, \mathbf{a}_r, c_r)_{r\in\mathcal{R}}, (\mathbf{e}_v^E, b_v)_{v\in\mathcal{V}}\}$, which are all Euclidean parameters that can be learned using standard Euclidean optimization techniques.

A.5 Comparison to Dihedral

We compare the performance of Dihedral (Xu and Li, 2019) versus that of ATTE in Table 6. Both methods combine rotations and reflections, but our approach learns attention-based transformations, while Dihedral learns a single parameter to determine which transformation to use. ATTE significantly outperforms Dihedral on all datasets, suggesting that using attention-based representations is important in order to learn the right geometric transformation for each relation.