Knowledge Graph Representation via Hierarchical Hyperbolic Neural Graph Embedding

Shen Wang[1§], Xiaokai Wei[2], Cicero Nogueira dos Santos[3], Zhiguo Wang[2], Ramesh Nallapati[2], Andrew Arnold[2] and Philip S. Yu[1]
[1]*Department of Computer Science, University of Illinois at Chicago*, USA
{swang224,psyu}@uic.edu
[2]*AWS AI Labs*, USA
{xiaokaiw, zhiguow, rnallapa, anarnld}@amazon.com
[3]*Google*, USA
{nogueirati}@gmail.com

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671651

***Abstract*—Knowledge graph enhanced information retrieval systems have attracted considerable attention due to their ability to improve performance and provide additional explainability. As the knowledge graphs usually include fruitful facts, they are also good sources of side information. However, recent studies have shown that the usefulness of knowledge graphs depends highly on their representation, e.g., the embeddings of entities and relations. Embedding entities and relations in low-dimensional space is a successful knowledge graph representation solution. Most of the works lie in modeling symmetry/asymmetry/composition/inversion relations but pay less attention to the hierarchical relations. Recent studies have observed the fact that there exist rich semantic hierarchical relations in knowledge graphs such as Freebase (entities are connected in a taxonomic hierarchy) and WordNet (entities are synsets linked together in a hierarchy).**

To address the above problems, we propose Hierarchical Hyperbolic Neural Graph Embedding (H^{2}E), a new knowledge graph representation approach, which is able to better preserve hierarchical relations. Specifically, the entities/relations representations are learned in a hyperbolic polar embedding space. In a hyperbolic polar embedding space, the entity and relation are modeled as a dual-embedding with modulus embedding part and phase embedding part, enabling the explicitly modeling of two types of hierarchies: inter-level hierarchy and intra-level hierarchy. As the polar embedding is defined in hyperbolic space, the ability of modeling and inferring hierarchical relations are mutual enhanced. In addition, by noticing the existence of the rich relational context, we propose an attentional neural context aggregation to adaptively integrate the relational context for further enhancing the ability to preserve the hierarchical relations. The empirical study on three benchmark datasets for the link prediction task demonstrates significant performance gains compared to some existing state-of-the-art methods and verifies the effectiveness of the proposed method on hierarchical relations.

***Index Terms*—knowledge graph representation, knowledge graph link prediction, graph neural network, hyperbolic space embedding**

I. Introduction

Knowledge Graphs (KGs) have emerged as an effective way to integrate disparate knowledge bases (KBs), which contain factual knowledge of the type (head entity, relation, tail entity). Recent studies have demonstrated the wide applications of KGs in information retrieval systems, such as semantic search (searching information with meaning and not only with lexical match) [3], [4], conversational search (taking conversational input and return fluent conversational responses to support multi-round conversations) [22], [25], question answering (answering questions posed in natural language automatically) [11], [17], and recommender systems (predicting a user's preference) [15], [33]–[37]. Incorporating KGs into information retrieval systems brings two-fold benefits: information retrieval systems can get boosted performance;

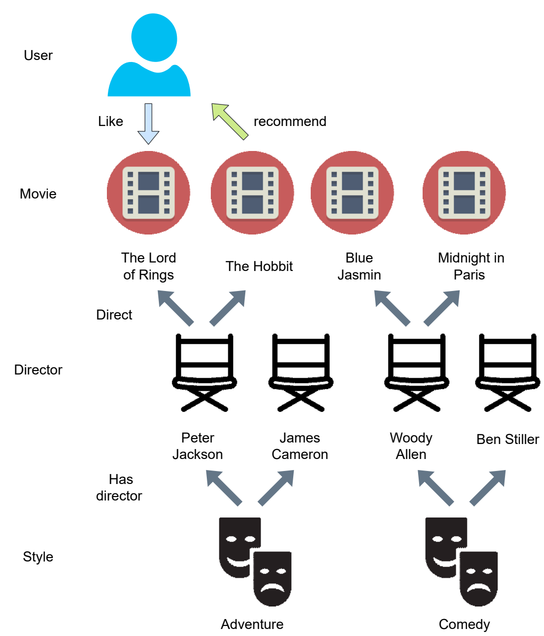

Fig. 1: An illustrative example of the knowledge graph used in a recommender system. There are three types of knowledge graph entities (*Movie*, *Director* and *Style*) and two types of relations (*Direct*, *Has director*), where entities are connected in a hierarchical (tree-like) structure. The style entities demonstrate the higher hierarchy than the director entities and movie entities.

§Work done during an internship at the AWS AI.

and the retrieval results are explainable. As KGs cover rich facts and these facts are connected as a graph, they are good sources of side information. However, recent studies [35], [36] have shown that the usefulness of knowledge graphs depends highly on their representations, e.g., the embedding of knowledge graph entities and relations. Improperly learned KG representations might even deteriorate the performance. Therefore, finding good knowledge graph representations is a task of fundamental importance.

Recent studies have demonstrated that embedding methods are effective and scalable for knowledge graph representation learning. The basic idea is to map the knowledge graph entities and relations to a low dimensional vector space, where the semantics and inherent structures are preserved. The relations in KGs demonstrate multiple properties, such as symmetry (e,g, *marriage*), anti-symmetry (e.g., *affiliation*), inversions (e.g., *hypernym* and *hyponym*), and compositions (e.g., *my mother's sister is my aunt*). Most works attempt to either implicitly or explicitly model one or a few of these relation patterns [6], [10], [28], [30], [41]. However, these methods fail to adequately model semantic hierarchical relations in knowledge graphs. Besides, they usually require high dimensionality to capture specific types of relations, which suffers from the curse of dimensionality and bad scalability. Therefore, an approach that is able to model and infer semantic hierarchical relations is needed.

Semantic hierarchical relations are widely presented in knowledge graphs [1], [7], [39], [42], [43]. For example, Figure 1 shows a subgraph taken from Freebase [5], which is used in a knowledge graph enhanced movie recommender system. We can see that entities are linked together in a hierarchical (tree-like) structure, e.g., in the triple (*Adventure*, *has director*, *Peter Jackson*), *adventure* is a parent of *Peter Jackson*, which demonstrates a higher level than *Peter Jackson*; in the triple (*Peter Jackson*, *direct*, *The Lord of Rings*), *Peter Jackson* is a parent of *The Lord of Rings*, which demonstrates a higher level than *The Lord of Rings*; based on previous two triples, we can infer that the *Adventure* shows a higher level than *The Lords of Rings*. A user's preference in a recommender system is hierarchical, e.g., a user might be interested in a specific director and a user might be interested in a specific style, as the director has a lower level, recommended candidates based on the director preference (lower hierarchy) should get higher rank than the ones based on the style preference (higher hierarchy). Therefore, modeling the hierarchical relations during the knowledge graph representation learning is meaningful and important to information retrieval task, such as recommender system.

Some recent works take hierarchical relations into consideration when learning knowledge graph representations, including explicit [39], [42], [43] and implicit approaches [1], [7]. In [39], [42], [43], the authors explicitly modeled the hierarchical relations, where [39], [43] used additional data and preprocess to get the hierarchical patterns; and [42] proposed a polar coordinate system to model the hierarchical relations. On the other hand, [1], [7] implicitly modeled the hierarchical relations by mapping knowledge graph entities/relations into a hyperbolic space. In hyperbolic space, the surface area of a hypersphere with a specific center "increases" exponentially [9] and more room can be used to distinguish the leaf nodes in a hierarchy. As a benefit from this property, the hierarchical structure can be naturally preserved during the KG representation learning even in a very low dimensional space. Although there has been some attempts, it is still challenging to model hierarchical relations because: 1) explicit approaches require additional information or higher dimensionality; and 2) implicit approaches have limited power to model hierarchical relations.

To address the above issues, in this paper we propose a novel knowledge graph representation learning approach called Hierarchical Hyperbolic Neural Graph Embedding ($\mathbf{H^2E}$), which learns knowledge graph entity/relation representations in a hyperbolic polar embedding space. In a hyperbolic polar embedding space, the entities and relations are modeled as a dual-embedding with modulus embedding part and phase embedding part, which enables the explicit modeling of two types of hierarchies: inter-level hierarchy and intra-level hierarchy. As the polar embedding is defined in hyperbolic space, the ability of modeling and inferring hierarchical relations are mutual reinforced. In addition, by noticing the existence of the rich relational context, we propose an attentional neural context aggregation to adaptively integrate the hierarchical relational context in the hyperbolic polar embedding space to further enhance the ability of capturing hierarchical relations. To demonstrate the effectiveness of the proposed knowledge graph representation learning method, we conduct extensive experiments on three benchmark datasets for link prediction task. Our experimental results show significant performance gains compared to some existing state-of-the-art methods and demonstrate the effectiveness of the proposed method on hierarchical relations.

In a nutshell, this paper makes the following contributions:

- We formulate the problem of the hierarchy-preserving knowledge graph representation learning in a hyperbolic polar embedding space and develop a graph neural network based framework, which can model the hierarchical relations explicitly and implicitly in a joint manner.
- We propose the hyperbolic polar embedding space with trainable curvatures to represent knowledge graph entities and relations, which can better preserve the hierarchical structure in the KG.
- We propose attentional graph neural networks in hyperbolic polar embedding spaces, which can leverage relational context to enhance the ability of capturing hierarchical relations.
- We conduct extensive experiments on three different knowledge graph datasets and demonstrate that the proposed method can effectively capture hierarchical relations during the KG representation learning and outperforms existing state-of-the-art embedding methods on the link prediction task.

The rest of the paper is organized as follows. We introduce preliminaries and problem formulation in Section 2. Section 3 discusses our proposed Hierarchical Hyperbolic Neural Graph Embedding in detail. Section 4 provides the experiment results and ablation studies. Section 5 discusses related work. Finally, Section 6 concludes the paper.

II. Preliminaries and Problem Formulation

In this section, we first introduce the preliminaries and some notations for the proposed hierarchical hyperbolic neural graph embedding model, including hyperbolic space and Gyrovector space algebra. Then we give the problem formulation of hierarchy-preserving knowledge graph representation in a hyperbolic polar embedding space.

A. Preliminaries

Definition 1. (***Hyperbolic space***) Hyperbolic space $\mathcal{H}$ is a homogeneous, isotropic, negatively curved (with constant negative curvature c) Riemann space, defined by a Riemannian manifold [19]. Typically, there are types of the Riemannian manifold, including Euclidean space (constant vanishing sectional curvature), hyperbolic space (constant negative sectional curvature), and spherical space (constant positive sectional curvature). As the hyperbolic space is negatively curved, it expands faster (exponentially) than Euclidean space (polynomially).

Definition 2. (***Poincaré disk model***) Poincaré disk model is a model used for describing a n-dimensional hyperbolic space in which all points are embedded in a Riemannian manifold $\mathcal{H}_c^n = (\mathbb{P}_c^n, g_{\mathcal{H}})$. $\mathbb{P}_c^n = \{\mathbf{x} \in \mathbb{R}^n : ||x||^2 < \frac{1}{c}\}$ is an open n-dimensional ball equipped with a Riemannian distance metric $g_{\mathcal{H}}(x) = (\frac{2}{1-||x||^2})g_E$. Here $x \in \mathbb{P}_c^n$, $||\cdot||$ is the Euclidean norm, and g_E is the Euclidean distance metric. The induced distance between two points u and v is given by:

$$d^{\mathbb{P}_c^n}(u,v) = \frac{2}{\sqrt{c}} \tanh^{-1}(\sqrt{c}||-\frac{(1+2c\mathbf{u}^T\mathbf{v}+c||\mathbf{v}||^2)\mathbf{u}+(1-c||\mathbf{u}||^2)\mathbf{v}}{1+2c\mathbf{u}^T\mathbf{v}+c^2||\mathbf{u}||^2||\mathbf{v}||^2}||) \tag{1}$$

Definition 3. (***Gyrovector space algebra***) Gyrovector space algebra is an analogy to the Euclidean geometry vector space operations for hyperbolic geometry. Möbius operations [13], [31] are defined in Gyrovector space and used to perform vector operation. The commonly used Möbius operations are described in Table I.

B. Problem Formulation

A knowledge graph is a multi-relational directed graph denoted by $\mathcal{G} = (\mathcal{E}, \mathcal{R})$, with $\mathcal{E}$ and $\mathcal{R}$ representing the set of entities (nodes) and relations (edges), respectively. A triple $(e_h, r, e_t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is represented as an edge r between head entity e_h and tail entity e_t in $\mathcal{G}$. The aim is to map entities ($e \in \mathcal{E}$)/relations ($r \in \mathcal{R}$) to their corresponding embeddings ($\mathbf{e}^{(M)} \in \mathbb{P}_c^n$, $\mathbf{e}^{(P)} \in \mathbb{P}_c^n$)/($\mathbf{r}^{(M)} \in \mathbb{P}_c^n$, $\mathbf{r}^{(P)} \in \mathbb{P}_c^n$). Here $\mathbb{P}_c^n$ indicates the embedding space in hyperbolic space with dimensionality n and curvature c, where hierarchical structure can be preserved properly. M/P are modulus/phase part indicators. In particular, the learned KG representation are used to predict the target entity of a given query with head entity and relation, $q := (e_h, r, ?)$–such that the predicted tuple doesn't exist in $\mathcal{G}$.

III. Hierarchical Hyperbolic Neural Graph Embedding

In this section, we introduce the proposed Hierarchical Hyperbolic Neural Graph Embedding (H^{2}E). We start by providing a framework overview of the proposed method. Then we detail the different modules of our approach, including initial embedding construction, attentional neural context aggregation, hyperbolic polar translation, hyperbolic scoring, and loss.

A. Framework Overview

The framework overview of the Hierarchical Hyperbolic Neural Graph Embedding (H^{2}E) is shown in Figure 2. It consists of following five important modules:

- **Initial Embedding Construction** Given a triple $(e_1, r_3, e_3) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ and the relational contexts (e_1, r_1, e_2) and (e_1, r_2, e_4), we first get the initial embeddings of the head/tail entities and relation. We define the initial embedding in a dual-embedding setting with hyperbolic modulus embedding and hyperbolic phase embedding to explicitly model two types of hierarchies: inter-level hierarchy and intra-level hierarchy. (See Section 3.2 for details)
- **Attentional Neural Context Aggregation** We develop a graph neural network to gather the relational context information around the entities. As different contexts affect the entity differently, we designed a graph neural network with attention – attentional neural context aggregation to adaptively integrate the relational context. The output is a relational context enhanced embedding. As our embedding consists of hyperbolic modulus embedding and hyperbolic phase embedding, both embeddings go through the attentional neural context aggregation and get their corresponding relational context enhanced embeddings. (See Section 3.3 for details)
- **Hyperbolic Polar Translation** After we get the relational context enhanced embeddings for modulus embedding and phase embedding, we perform the hyperbolic modulus translation and hyperbolic phase translation to get the hyperbolic translated modulus embedding and phase embedding. (See Section 3.4 for details)
- **Hyperbolic Scoring** Based on the hyperbolic translated modulus/phase embedding and relational context enhanced modulus/phase embedding for entity tail, we compute their hyperbolic distances as the scores and combine them together to get the final score. (See Section 3.5 for details)
- **Loss** After the hyperbolic final score is obtained, we add it to a negative sampling loss to train the model. (See Section 3.6 for details)

TABLE I: Commonly used Möbius operations for hyperbolic space.

Operation	Notation	Explanation
Möbius addition	$\oplus_c$	$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1+2c\mathbf{x}^T\mathbf{y}+c\|\|\mathbf{y}\|\|^2)\mathbf{x}+(1-c\|\|\mathbf{x}\|\|^2)\mathbf{y}}{1+2c\mathbf{x}^T\mathbf{y}+c^2\|\|\mathbf{x}\|\|^2\|\|\mathbf{y}\|\|^2}$
Möbius scalar multiplication	$\otimes_c$	$r \otimes_c \mathbf{x} = \exp_o^c(r \log_o^c(\mathbf{x}))$
Möbius element-wise multiplication	$\odot_c$	$\mathbf{x} \odot_c \mathbf{y} = \exp_o^c(\mathbf{x} \circ \log_o^c(\mathbf{y}))$
Möbius matrix multiplication	$\otimes_c$	$\mathbf{M} \otimes_c \mathbf{x} = \exp_o^c(\mathbf{M} \log_o^c(\mathbf{x}))$
Möbius activation	σ_c	$\sigma_c(\mathbf{x}) = \exp_o^c(\sigma(\log_o^c(\mathbf{x})))$
Möbius exponential map	$\exp_o^c$	$\exp_o^c(\mathbf{u}) = \tanh(\sqrt{c}\|\|\mathbf{u}\|\|)\frac{\mathbf{u}}{\sqrt{c}\|\|\mathbf{u}\|\|}$
Möbius logarithmic map	$\log_o^c$	$\log_o^c(\mathbf{v}) = \tanh^{-1}(\sqrt{c}\|\|\mathbf{v}\|\|)\frac{\mathbf{v}}{\sqrt{c}\|\|\mathbf{v}\|\|}$
Möbius distance	$d^{\mathbb{P}_c^n}$	$d^{\mathbb{P}_c^n}(\mathbf{u}, \mathbf{v}) = \frac{2}{\sqrt{c}} \tanh^{-1}(\sqrt{c}\|\|-\mathbf{u} \oplus_c \mathbf{v}\|\|)$

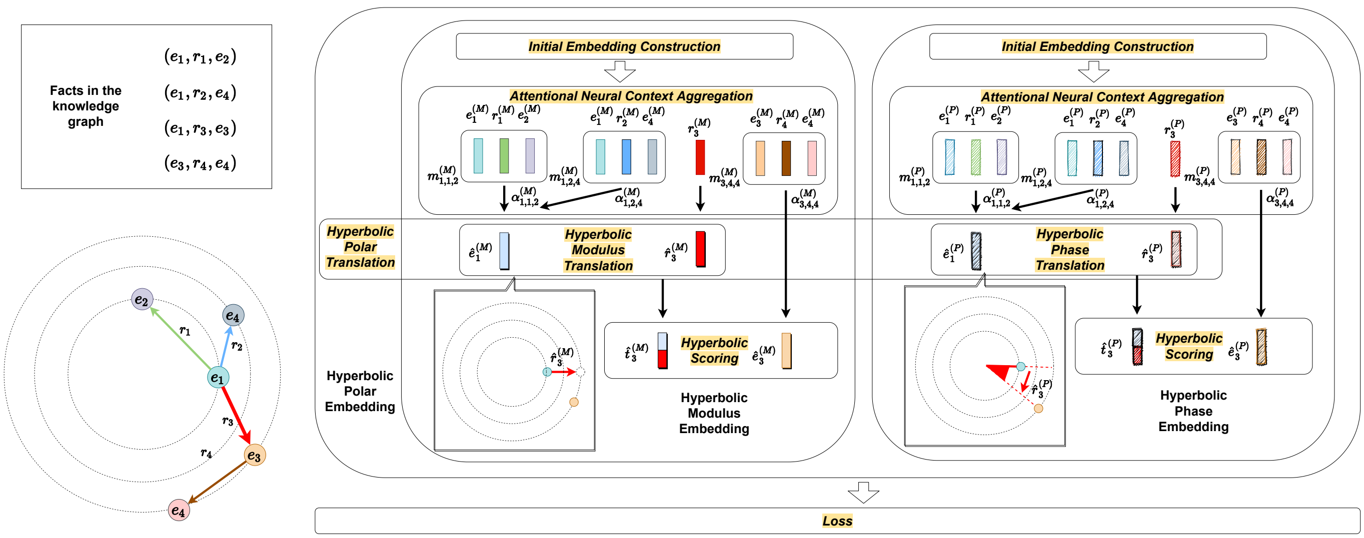

Fig. 2: The left part is a subgraph of the knowledge graph with four facts. The right part is the framework overview of the Hierarchical Hyperbolic Neural Graph Embedding (H^2E), which consists of five key modules: Initial Embedding Construction (See Section 3.2 for details), Attentional Neural Context Aggregation (See Section 3.3 for details), Hyperbolic Polar Translation (See Section 3.4 for details), Hyperbolic Scoring (See Section 3.5 for details), and Loss (See Section 3.6 for details). Hierarchical Hyperbolic Neural Graph Embedding is a dual-embedding, which consists of modulus part embedding and hyperbolic phase part. Modulus part embedding models the inter-level hierarchy. Phase Embedding models the intra-level hierarchy. Initial Embedding Construction prepares the initial embedding of the knowledge graph entities/relations as hyperbolic modulus embedding and hyperbolic phase embedding. Attentional Neural Context Aggregation takes the relational context into consideration and adaptively integrate these relational contexts to get the relational context enhanced embedding. Hyperbolic Polar Translation compute the translated head entity over the specific relation in both modulus and phase part. Hyperbolic Scoring measures the distance between the translated head entity over the specific relation and the tail entity. The Loss component combines the scores from Hyperbolic Scoring to compute the training loss.

B. Initial Embedding Construction

Based on the hierarchical level, KG relations among entities can be classified in two categories: (i) Inter-level hierarchy: hierarchical relations among entities in different levels (e.g. *Adventure* vs *Director*, *Director* vs *Movie*). (ii) Intra-level hierarchy: hierarchical relations among entities in the same level (e.g. *Adventure* vs *Comedy*, *Peter Jackson* vs *James Cameron*). Implicit approaches [1], [7] model knowledge graph entities/relations in hyperbolic space, which can preserve the hierarchical structure automatically during the embedding stage. In order to enhance the modeling power of the hierarchical structure, we model the knowledge graph entities/relations in a hyperbolic polar embedding space, which can take the benefit of both explicit and implicit approaches and mutually enhance each other. In particular, hyperbolic polar embedding consists of hyperbolic modulus embedding and hyperbolic phase embedding, with hyperbolic modulus embedding modeling the inter-level hierarchy and hyperbolic phase embedding modeling the intra-level hierarchy.

Given a triple $(e_h, r, e_t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, we first get the initial embeddings of the head/tail entities and the relation. Specifically, there are two types of embeddings: hyperbolic modulus embedding and hyperbolic phase embedding. Their corresponding initial embeddings are defined as follows:

$$\begin{cases} e_h^{(M)}, e_t^{(M)} \in \mathbb{P}_c^n & \text{hyperbolic modulus entity embedding} \\ r^{(M)} \in \mathbb{P}_c^n & \text{hyperbolic modulus relation embedding} \\ e_h^{(P)}, e_t^{(P)} \in \mathbb{P}_c^n & \text{hyperbolic phase entity embedding} \\ r^{(P)} \in \mathbb{P}_c^n & \text{hyperbolic phase relation embedding} \end{cases}$$

C. Attentional Neural Context Aggregation

Recent studies [2], [23], [26], [32] on graph neural networks have demonstrated improved KG embedding quality by leveraging relational context. In addition, the context can help to model hierarchical relations with good explainability. For example, entities are more similar if they have deeper least common ancestors. Furthermore, a triple may have multiple contexts, but not all the contexts are helpful. For example, to determine if two entities are similar, the higher hierarchical-level common contexts are more useful. Motivated by these ideas, we propose attentional neural context aggregation, a graph-neural-network-based approach to get the relational context enhanced embedding by selectively aggregating the relational context information. Formally, it can be described as follows:

$$\hat{\mathbf{e}}_i^{(\mathbb{P}_c^n)(M/P)} = \mathrm{Agg}(\mathbf{e}_j^{(\mathbb{P}_c^n)(M/P)}, \mathbf{r}_k^{(\mathbb{P}_c^n)(M/P)} | j \in \mathcal{N}_i, k \in \mathcal{R}_{ij}) \tag{2}$$

where n, c, $\mathbb{P}_c^n$ and M/P are the dimension indicator, curvature indicator, space indicator and modulus/phase indicator respectively. $\hat{\mathbf{e}}_i^{(\mathbb{P}_c^n)(M/P)}$ denotes the output relational context enhanced embedding, $\mathbf{e}_j^{(\mathbb{P}_c^n)(M/P)}$ denotes the embedding of a tail entity e_j corresponding to the target entity e_i with the embedding $\mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)}$. $\mathcal{N}_i$ represents set of entity typed relational context of entity e_i and $\mathcal{R}_{ij}$ denotes the set of relations between entities e_i and e_j. Both hyperbolic modulus embedding and hyperbolic phase embedding go through the attentional neural context aggregation to get their relational context enhanced embeddings for the next module.

Different from traditional graph, the context of an entity is more than a neighbourhood entity. It also includes the relation between the neighbouring entities. Therefore, we gather the relational context $\mathbf{m}_{i,j,k}^{(\mathbb{P}_c^n)(M/P)}$ with both neighbourhood entity and corresponding relation to update the entity embedding. The relational context is computed by concatenating and linear transforming the entities and relation embeddings as follows:

$$\mathbf{m}_{i,j,k}^{(\mathbb{P}_c^n)(M/P)} = \mathbf{W'}_1 \otimes_c [\mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)} || \mathbf{e}_j^{(\mathbb{P}_c^n)(M/P)} || \mathbf{r}_k^{(\mathbb{P}_c^n)(M/P)}] \tag{3}$$

where $\mathbf{W'}_1$ denotes the linear transformation matrix and $\otimes_c$ indicates the Möbius matrix multiplication as described in Table I. [|| ||] represents the concatenation operation.

As different context might have different levels of importance, we gather the relation context in an attentional manner. In particular, we leverage the multi-head attention and perform two iterations of embedding update to gather two-hop relational context. The new entity embedding is computed as follows:

$$b_{i,j,k}^{(\mathbb{P}_c^n)(M/P)} = \exp_o^c(\mathrm{LeakyReLU}(\log_o^c(\mathbf{W'}_2 \otimes_c \mathbf{m}_{ijk}^{(\mathbb{P}_c^n)(M/P)})) \tag{4}$$

$$\alpha_{i,j,k}^{(\mathbb{P}_c^n)(M/P)} = \mathrm{softmax}_{jk}(b_{i,j,k}^{(\mathbb{P}_c^n)(M/P)}) \tag{5}$$

$$\mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)'} = \|_{l=1}^{L} \, \sigma_c \left(\sum_{j \in \mathcal{N}_i} \sum_{k \in \mathcal{R}_{ij}} \alpha_{i,j,k}^{(\mathbb{P}_c^n)(M/P)(l)} \otimes_c \mathbf{m}_{i,j,k}^{(\mathbb{P}_c^n)(M/P)(l)} \right) \tag{6}$$

$$\mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)''} = \sigma_c \left(\frac{1}{L} \sum_{l=1}^{L} \sum_{j \in \mathcal{N}_i} \sum_{k \in \mathcal{R}_{ij}} \alpha_{ijk}^{(\mathbb{P}_c^n)(M/P)(l)'} \otimes_c \mathbf{m}_{ijk}^{(\mathbb{P}_c^n)(M/P)(l)'} \right) \tag{7}$$

Equation 4 computes the context importance $b_{i,j,k}^{(\mathbb{P}_c^n)(M/P)}$ corresponding to relation embedding $\mathbf{m}_{i,j,k}^{(\mathbb{P}_c^n)(M/P)}$, Equation 5 computes the attention value corresponding to relation embedding $\mathbf{m}_{i,j,k}^{(\mathbb{P}_c^n)(M/P)}$ and Equation 6 computes the 1-hop new entity embedding after the attentional relational context embedding fusion. Equation 7 computes the 2-hop new entity embedding after the attentional relational context embedding fusion. Here $\mathbf{W'}_2$ is a linear transformation matrix. L indicates the number of heads.

The new representation of the relation is computed by applying two linear transformations:

$$\mathbf{r}_k^{(\mathbb{P}_c^n)(M/P)''} = \mathbf{W''}_{rel} \otimes_c (\mathbf{W'}_{rel} \otimes_c \mathbf{r}_k^{(\mathbb{P}_c^n)(M/P)}) \tag{8}$$

where $\mathbf{W'}_{rel}$ and $\mathbf{W''}_{rel}$ denotes the 1-hop and 2-hop linear transformation matrices respectively for the relation embedding.

As the initial embedding is also important, we further employ a linear transformation layer to fuse the initial embedding and the new embedding as follows:

$$\mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)'''} = (\mathbf{W'}_f \otimes_c \mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)}) \oplus_c \mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)''} \tag{9}$$

where $\mathbf{W'}_f$ is a weight matrix to trade-off the old and new representations. In the end, the resulting relational context enhanced embeddings of entity and relation are described as follows:

$$\hat{\mathbf{e}}_i^{(\mathbb{P}_c^n)(M/P)} = \mathbf{e}_i^{(\mathbb{P}_c^n)(M/P)'''} \tag{10}$$

$$\hat{\mathbf{r}}_k^{(\mathbb{P}_c^n)(M/P)} = \mathbf{r}_k^{(\mathbb{P}_c^n)(M/P)''} \tag{11}$$

D. Hyperbolic Polar Translation

After the relational context enhanced embeddings are obtained, we feed them to a translational distance framework, such that the new head entity embedding is combined with the relation embedding to compute the translated embedding. The hyperbolic polar embedding consists of two sub-parts: hyperbolic modulus embedding (to model inter-level hierarchies) and hyperbolic phase embedding (to model intra-level hierarchies). In this section, we introduce the proposed hyperbolic modulus translation and hyperbolic phase translation.

The hyperbolic modulus translation aims to translate the head entity to its translated tail entity over a specific relation in hyperbolic modulus embedding space. Entities linked in a hierarchical structure, usually demonstrate the different level, e.g., in the triple (*Adventure*, *has director*, *Peter Jackson*), *Adventure* is a parent of *Peter Jackson*, which demonstrates a higher level than *Peter Jackson*; in the triple (*Peter Jackson*, *direct*, *The Lord of Rings*), *Peter Jackson* is a parent of *The Lord of Rings*, which demonstrates a higher level than *The Lord of Rings*. Therefore, we can use depth to model different levels of hierarchies. In this way, each entry of entity corresponds to a modulus and each entry of relation corresponds to a scaling transformation between two entities in the hyperbolic space. Formally, we describe the hyperbolic modulus translation as follows:

$$\begin{aligned}\mathbf{t}^{(\mathbb{P}_c^n)(M)} &= \text{trans}^{(\mathbb{P}_c^n)(M)}(\hat{\mathbf{e}}_h^{(\mathbb{P}_c^n)(M)}, \mathbf{r}^{(\mathbb{P}_c^n)(M)}) \\ &= \hat{\mathbf{e}}_h^{(\mathbb{P}_c^n)(M)} \odot_c \mathbf{r}^{(\mathbb{P}_c^n)(M)}\end{aligned} \tag{12}$$

where $\odot_c$ denotes the Möbius element-wise product. The output $\mathbf{t}^{(\mathbb{P}_c^n)(M)}$ is the hyperbolic translated modulus entity embedding.

The hyperbolic phase translation intend to translate the head entity to its translated tail entity over a specific relation in hyperbolic phase embedding space. Entities in the same hierarchical level can lie on a circle, as they share the same radius but have different angles. In particular, each entry of entity corresponds to a phase and each entry of relation corresponds to a phase transformation between two phases. Formally, we describe the hyperbolic phase translation as follows:

$$\begin{aligned}\mathbf{t}^{(\mathbb{P}_c^n)(P)} &= \text{trans}^{(\mathbb{P}_c^n)(P)}(\hat{\mathbf{e}}_h^{(\mathbb{P}_c^n)(P)}, \mathbf{r}^{(\mathbb{P}_c^n)(P)}) \\ &= (\hat{\mathbf{e}}_h^{(\mathbb{P}_c^n)(P)} \oplus_c \mathbf{r}^{(\mathbb{P}_c^n)(P)}) \text{mod } 2\pi\end{aligned} \tag{13}$$

where $\oplus_c$ denotes the Möbius addition. The output $\mathbf{t}^{(\mathbb{P}_c^n)(P)}$ is the hyperbolic translated phase entity embedding.

E. Hyperbolic Scoring

Based on the hyperbolic translated modulus/phase entity embeddings, we can perform the hyperbolic scoring to compute the distance between the translated modulus/phase entity embeddings and the tail entity embedding. In particular, the hyperbolic scoring consists of two parts: hyperbolic modulus scoring and hyperbolic phase scoring.

For the hyperbolic modulus scoring, the distance function is defined as follows:

$$\begin{aligned}s^{(\mathbb{P}_c^n)(M)}(e_h, e_t) &= d^{(\mathbb{P}_c^n)(M)}(e_h, e_t) \\ &= d^{(\mathbb{P}_c^n)(M)}(\mathbf{t}^{(\mathbb{P}_c^n)(M)}, \hat{\mathbf{e}}_t^{(\mathbb{P}_c^n)(M)})\end{aligned} \tag{14}$$

As for the scoring is in hyperbolic space, the Möbius distance $d^{(\mathbb{P}_c^n)}$ is used. Similarly, for the hyperbolic phase scoring, the distance function is defined as follows:

$$\begin{aligned}s^{(\mathbb{P}_c^n)(P)}(e_h, e_t) &= d^{(\mathbb{P}_c^n)(P)}(e_h, e_t) \\ &= d^{(\mathbb{P}_c^n)(P)}(\mathbf{t}^{(\mathbb{P}_c^n)(P)}, \hat{\mathbf{e}}_t^{(\mathbb{P}_c^n)(P)})\end{aligned} \tag{15}$$

Combining the hyperbolic modulus scoring and the hyperbolic phase scoring, we can obtain the overall hyperbolic polar scoring, whose distance function is given by:

$$s^{(\mathbb{P}_c^n)}(e_h, e_t) = s^{(\mathbb{P}_c^n)(M)}(e_h, e_t) + \lambda s^{(\mathbb{P}_c^n)(P)}(e_h, e_t) \tag{16}$$

where λ is a parameter learned by the model.

F. Loss

To train our proposed H^{2}E model, we minimize a negative sampling loss function as [28], where Ne negative samples are constructed for each triple (e_h, r, e_t) by performing the negative triple sampling with a distribution given by:

$$\begin{aligned}&p(e'_{h(j)}, r, e'_{t(j)} | \{(e_{h(i)}, r_i, e_{t(i)})\}) = \\ &\qquad \frac{\exp \alpha s^{(\mathbb{P}_c^n)}(e'_{h(j)}, e'_{t(j)})}{\sum_i \exp \alpha s^{(\mathbb{P}_c^n)}(e'_{h(i)}, e'_{t(i)})}\end{aligned} \tag{17}$$

where α is the temperature of sampling, $(e'_{h(i)}, r_i, e'_{t(i)})$ is the i-th negative sample. Formally, the loss function is given by:

$$\begin{aligned}\mathcal{L} = &- \log \sigma(\gamma - s^{(\mathbb{P}_c^n)}(e_h, e_t)) \\ &- \sum_{i=1}^{Ne} p(e'_{h(i)}, r_i, e'_{t(i)}) \log \sigma(s^{(\mathbb{P}_c^n)}(e'_{h(i)}, e'_{t(i)}) - \gamma)\end{aligned} \tag{18}$$

where γ denotes the fixed margin, σ denotes the sigmoid function. To optimize the proposed model properly, following [8], all parameters are defined in the tangent space at the origin. Moreover, we adopt the trainable curvature as [8] and set curvature c as a trainable parameter in order to better model the hierarchical structure.

IV. Experiments and Results

In this section, we evaluate the proposed methods on three KG benchmark datasets. We first introduce the datasets used in the experiments, the compared methods and the experimental setup. Next,we show the overall results. We further conduct several ablation studies, including dimensionality, graph neural network, and relation type.

A. Datasets

To evaluate the performace of the proposed method on the link prediction task and in order to cover different levels of hierarchical structure and different scales, we perform experiments on three commonly used benchmark datasets: WN18RR [29], FB15k-237 [10], and YAGO3-10 [21]. We provide the data statistics in Table II.

B. Compared Methods

To evaluate the performance of the proposed method, we compare with the following state-of-the-art KG representation methods:

- **RESCAL** [24]: Euclidean embedding models with each relations as a full rank matrix.
- **TransE** [6]: First translational distance Euclidean embedding.
- **DisMult** [6]: Euclidean embedding models with relational matrix being diagonal.
- **MuRE** [1]: Translational distance Euclidean embedding with diagonal relation matrix.
- **complEx** [30]: Extension of DisMult in complex space.
- **RotatE** [28]: Extension of TransE in a complex space with modulus part and phase part.

TABLE II: Statistics of datasets.

Dataset	#Entity	#Relation	#Train	#Valid	#Test	Hierarchical Structure	Scale
WN18RR	40,943	11	86,835	3,034	3,134	Rich	Small
FB15-237	14,541	237	272,115	17,535	20,466	Medium	Medium
YAGO3-10	123,182	37	1,079,040	5,000	5,000	Rich	Large

- **Conve** [10]: NN-based method with score function defined by the convolutional neural network.
- **CompGCN** [32]: NN-based method with score function defined by the graph convolutional network.
- **A2N** [2]: NN-based method with score function defined by the graph attentional network.
- **HAKE** [42]: Hierarchical aware translational distance model in the polar embedding space.
- **MuRP** [1]: Translational distance hyperbolic embedding with diagonal relation matrix.

We report the results taken from the original papers in Section 4.4 for these baseline methods.

Since the proposed H^{2}E consists of several components, to analyze the advantage of hyperbolic modeling/polar embedding/graph neural network, we consider several non-hyperbolic variations of H^{2}E as follow:

- **HE**: polar embedding with attentional GNN.
- **PE**: polar embedding only without GNN.
- **HGCNE**: polar embedding only with GCN.
- **HGATE-khead**: polar embedding only with k-head-attentional GNN.

C. *Experimental Setup*

Following previous KG link prediction work [6], MRR and Hits@K are used as evaluation metrics. We perform the optimization in tangent space as [8] and use standard Euclidean optimizers. We implement the proposed method in PyTorch and conduct the experiments on NVIDIA Tesla V100 GPU. For our proposed model, we conduct a hyperparameter search on dimensionality, learning rate, optimizer, negative sample size, batch size and number of attention heads. We report the best hyperparameters (dimensionality, learning rate, optimizer, negative sample size, batch size and number of attention head) for each dataset as follows: {WN18RR: 200, 0.001, Adam, 500, 500, 4}, {FB15k-237: 200, 0.05, Adagrad, 500, 500, 4}, {YAGO3-10: 500, 0.005, Adam, 250, 500, 4}.

D. *Overall Results*

In this subsection, we compare the proposed method with existing state-of-the-art methods. The experimental results are shown in Table III, where results of the baseline methods are reported as indicated in the original paper with the best hyper-parameters. Both proposed HE and H^{2}E almost outperforms all the baseline on datasets with rich hierarchical structure, such as WN18RR and YAGO3-10, and achieves the second best results on hierarchy-medium dataset FB15k-237. These experimental results confirm the effectiveness of the proposed methods in preserving hierarchical relations in KG. H^{2}E outperforms its variant HE, which demonstrates the effectiveness of hyperbolic embeddings. Besides, we can see hyperbolic methods achieve top performance on WN18RR and YAGO3-10 and beat the non-hyperbolic methods. Another interesting experiment result is that the GNN-based methods demonstrate strong performance on all the datasets, which means levering that relational contexts is beneficial to the KG representation.

E. *Analysis on Dimensionality*

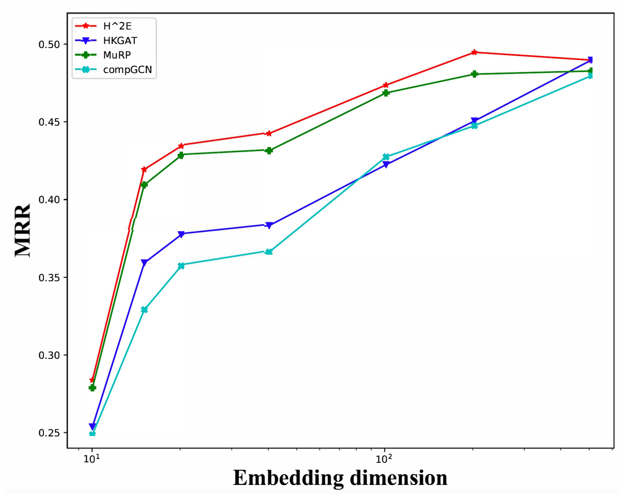

Fig. 3: MRR per embedding dimensionality with $d \in \{10, 15, 20, 40, 100, 200, 500\}$ on WN18RR dataset. Average computed over 10 runs.

In this subsection, the role of the dimensionality is investigated. We conduct experiments on WN18RR and report the MRR of HE, H^{2}E against state-of-the-art methods MuRP, compGCN at different dimensions $d \in \{10, 15, 20, 40, 100, 200, 500\}$. Fig. 3 shows the results, which are obtained by averaging over 10 runs. H^{2}E achieves better performance across a broad range of dimensions, especially in lower dimensionality.

F. *Analysis on Graph Neural Network*

In this section, the effectiveness of the graph neural network module is studied. We show the ablation results on WN18RR and FB15k-237 datasets in Table IV. We can see that the model with graph neural network outperforms the non-GNN model. The graph attention network with one head has similar performance compared to the graph convolutional variants. When the number of heads increases, the performance also increases.

To further study the effectiveness of the graph attention mechanism, we take some qualitative examples from FB15-k-237. The examples in Fig. 4 demonstrate how the model can benefit from the relational contexts. In the first example queries, to predict the tail of the head entity

TABLE III: Link prediction results for embeddings on WN18RR, FB15k-237 and YAGO3-10. Best results are in bold and the second best results are in underlined.

	WN18RR				FB15k-237				YAGO3-10			
Model	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@1	H@10
RESCAL	.420	-	-	.447	.270	-	-		-	-	-	-
TransE	.226	-	-	.501	.294	-	-	.465	-	-	-	-
DisMult	.430	.390	.440	.490	.241	.155	.263	.419	.340	.240	.380	.540
ComplEx	.440	.410	.460	.510	.247	.158	.275	.428	.360	.260	.400	.550
ConvE	.430	.400	.440	.520	.325	.237	.356	.501	.440	.350	.490	.620
RotatE	.476	.428	.492	.571	.338	.241	.375	.533	.495	.402	.550	.670
MuRE	.465	.436	.487	.554	.336	.245	.370	.521	-	-	-	-
MuRP	.481	.440	.495	.566	.335	.243	.367	.518	-	-	-	-
CompGCN	.479	.443	.494	.546	**.355**	**.264**	**.390**	**.535**	.489	.395	.500	.582
A2N	.430	.410	.440	.510	.317	.232	.348	.486	.445	.349	.482	.501
HAKE	.497	.452	.516	.582	.346	.250	.348	.486	.445	.349	.482	.501
HE	.490	.451	.517	.583	.345	.249	.384	**.535**	.520	.431	.581	.675
H^2E	**.500**	**.456**	**.522**	**.593**	**.355**	.260	.386	.533	**.559**	**.466**	**.604**	**.694**

TABLE IV: Ablation results on WN18RR and FB15k-237 datasets. We compare with variants of the proposed model using various graph neural network setting.

	WN18RR				FB15k-237			
Model	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
CompGCN	.479	.443	.494	.546	**.355**	**.264**	**.390**	**.535**
A2N	.430	.410	.440	.510	.317	.232	.348	.486
PE	.458	.421	.452	.523	.344	.248	.381	.538
HGCNE	.485	.444	.508	.580	.347	.243	.386	.527
HGATE-1head	.482	.442	.509	.578	.347	.241	.387	.526
HGATE-4head	.492	.448	.518	.588	.350	.248	.388	.531
HGATE-8head	**.497**	**.450**	**.521**	**.591**	.353	.263	**.390**	.532

(Fantastic_Four:_Rise_of_the_Silver_Surfer, genre, ?)
Prediction: Fantasy Prob:0.112
High score context:
(Fantastic_Four:_Rise_of_the_Silver_Surfer, genre, Superhero_film) weight: 0.070
(Fantastic_Four:_Rise_of_the_Silver_Surfer, genre: Superhero) weight: 0.072
(Fantastic_Four:_Rise_of_the_Silver_Surfer, genre: Science_fiction_film) weight: 0.053
Low score context:
(Fantastic_Four:_Rise_of_the_Silver_Surfer, language, Arabic_language) weight: 0.009
(Fantastic_Four:_Rise_of_the_Silver_Surfer, language, English_language) weight: 0.009

(Burt_Young, nationality, ?)
Prediction: US Prob:0.101
High score context:
(Burt_Young, place_of_birth, Queens) weight: 0.282
(Burt_Young, place_lived, Queens) weight: 0.205
(Burt_Young, Inverse:ethnicity/people, Italian_American) weight: 0.158
Low score context:
(Burt_Young, performance/film, Transamerica) weight: 0.038
(Burt_Young, gender, Male) weight:0.037

Fig. 4: Example of queries. We provide their top prediction and the set of top 3, bottom 2 attention relational contexts with corresponding attention probabilities for the H^2E model.

Fantastic_Four:_Rise_of_the_Silver_Surfer and relation *genre*, the contextual triples with the same relation are assigned the high attention scores, while the contextual triples with the relations irrelevant to the given relation *genre* are assigned the lower scores. In the second example, given the head entity *Burt_Young* and relation *nationality*, the contextual triples with the relation highly correlating with *nationality* are assigned the high scores. Also, the prediction can be the tails of these relations. Therefore, the attention mechanism helps to identify the useful relational contexts, and the relational contexts with high attention scores play key roles in identifying the correct prediction.

G. Analysis on Relation Type

In this subsection, we investigate how the performance of the proposed method is affected by relation types on WN18RR. We report a number of metrics to describe each relation, including the measures of hierarchy-level (global graph curvature ξ_G [14] and Krackhardt hierarchy score (Khs) [18]), and maximum/average shortest path between any two nodes in the KG for hierarchical relations. Specifically, we compare hits@10 for each relation of RotatE, MuRE, MuRP, HE and H^2E for entity embeddings at a lower dimension with $n = 20$. Table V demonstrates that all the models achieve comparable performance on non-hierarchical, symmetric relations with the low hierarchy-level, such as *verb_group*, whereas H^2E generally outperforms the other models on hierarchical relations. In additional, the performance gap between HE and H^2E is generally bigger for relations with deep hierarchy.

V. Related Work

Our work is related to non-hierarchy-aware knowledge graph representation and hierarchy-aware knowledge graph representation. We briefly discuss them in the following subsections.

A. Non-hierarchy-aware Knowledge Graph Representation

KG representation has been extensively studied, which typically project entity/relation in the Euclidean embedding space. Many approaches have been proposed, such as TransE [6], TransD [16], TransH [38] and TransR [20] and bilinear models, such as RESCAL [24] and DistMult [40]. Recently,

TABLE V: Comparison of hits@10 for WN18RR relations for d = 20.

WN relation name	ξ_G	Khs	MaxPath	AvgPath	RotatE	MuRE	MuRP	HE	H^{2}E
also_see	-2.09	.24	44	15.2	.473	.634	.705	.694	**.726**
hypernym	-2.46	.99	18	4.5	.102	.161	.228	.230	**.236**
has_part	-1.43	1	13	2.2	.198	.215	.282	.301	**.316**
member_meronym	-2.90	1	10	3.9	.201	.272	.346	.342	**.350**
synset_domain_topic_of	-0.69	.99	3	1.1	.225	.316	.430	.421	**.445**
instance_hypernum	-0.82	1	3	1.0	.453	**.488**	.471	.475	.477
member_of_domain_region	-0.78	1	2	1.0	.287	.308	.347	.345	**.352**
member_of_domain_usage	-0.74	1	2	1.0	.379	.396	.417	.409	**.421**
derivationally_related _form	-3.84	.04	-	-	.936	.954	.967	.965	**.968**
similar_to	-1.00	0	-	-	.997	**1**	**1**	**1**	**1**
verb_group	-0.5	0	-	-	.958	.974	.974	.980	**.982**

neural network models have attracted considerable research interest, which map the entities and relations into the embedding space via neural network in an end-to-end manner. Based on the neural network the model use, we can categorize these methods in three groups: NN-based model [12], [27] (using fully connected neural network), CNN-based model [10], [10] (using convolutional neural networks), and GNN-based model [2], [23], [26] (using graph neural networks). However, all these approaches overlook the hierarchical nature of relations and pay less attention on the hierarchical relations during the KG representation learning.

B. Hierarchy-aware Knowledge Graph Representation

Some recent works take hierarchical relations into consideration when learning knowledge graph representations, they can be categorized in two categories: implicit model and explicit model.

1) Implicit model: Recently, a few attempts [1], [7] learn KG representation in hyperbolic space. MuRP [1] initially developed the hyperbolic analogy of translational distance model for KG embedding. It has three limitations: as a translation model, it cannot encode complex relations; the hyperbolic space is defined with a fixed curvature, which has less flexibility to the data; it ignores the rich context information. AttH [7] built a rotation based model in hyperbolic space for KG embedding. However, it overlooks the rich relational context during the embedding learning.

2) Explicit model: [39], [42], [43] explicit model the hierarchical relations, where [39], [43] used additional data and preprocess to get the hierarchical pattern; [42] proposed a polar coordinate system to model the hierarchical relations. Although a number of attempts have been done, it is still challenging to model hierarchical relations. The explicit approaches require additional information or higher dimensionality and they also overlook the rich relational context during the embedding learning.

A few recent works take hierarchical relations into consideration. [39], [43] explicitly model the semantic hierarchy. However, they require additional data or process and cannot capture the hierarchy automatically. [42] uses the polar coordinate system to model the hierarchical structure, while its embeddings are defined in the Euclidean space, which has limited modeling power. On the other hand, embedding hierarchical data in hyperbolic space has attracted considerable research interest, due to its ability of naturally capturing the semantic hierarchies. [1] proposes MuRP to learn KG embeddings on hyperbolic space in order to capture the hierarchical structure automatically. However, MuRP has a number of limitations. First, it is a translation model, which cannot encode complex relations. Second, it uses a fixed curvature, which may not adapt to the data. Third, it ignores the rich context information. More recently, [7] combines reflections and rotations in hyperbolic space with attention to capture both hierarchical and logical patterns. However, it overlooks the rich relational context during the embedding learning. Instead, our proposed method overcomes these limitations.

VI. Conclusion

In this work, we propose H^{2}E, a new knowledge graph representation approach to better preserve the hierarchical relations. H^{2}E learns the entity/relation representation in a hyperbolic polar embedding space and further enhances the hierarchy preserving ability via attentional neural context aggregation. With the help of the H^{2}E, the hierarchical relations can be modeled in an automatic and effective way. Experiments show the significant performance gains over some state-of-the-art methods on several benchmark datasets in terms of link prediction and verify the effectiveness of modeling hierarchical relations.

Acknowledgment

Shen Wang and Philip S. Yu are supported in part by NSF grants III-1763325, III-1909323, and SaTC-1930941.

References

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

[2] Trapit Bansal, Da-Cheng Juan, Sujith Ravi, and Andrew McCallum. A2n: Attending to neighbors for knowledge graph inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4387–4392, 2019.

[3] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544, 2013.

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

[5] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, 2008.

[6] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[7] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6901–6914, 2020.

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

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

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

[11] Dennis Diefenbach, Kamal Singh, and Pierre Maret. Wdaqua-core1: a question answering service for rdf knowledge bases. In *Companion Proceedings of the The Web Conference 2018*, pages 1087–1091, 2018.

[12] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610, 2014.

[13] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic entailment cones for learning hierarchical embeddings. *arXiv preprint arXiv:1804.01882*, 2018.

[14] Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. Learning mixed-curvature representations in product spaces. 2018.

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

[16] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 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, 2015.

[17] Simon Keizer, Markus Guhe, Heriberto Cuayáhuitl, Ioannis Efstathiou, Klaus-Peter Engelbrecht, Mihai Dobre, Alex Lascarides, Oliver Lemon, et al. Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents. ACL, 2017.

[18] David Krackhardt. Graph theoretical dimensions of informal organizations. In *Computational organization theory*, pages 107–130. Psychology Press, 2014.

[19] Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

[20] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

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

[22] Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, 2019.

[23] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. Learning attention-based embeddings for relation prediction in knowledge graphs. *arXiv preprint arXiv:1906.01195*, 2019.

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

[25] Zheng-Yu Niu, Hua Wu, Haifeng Wang, et al. Knowledge aware conversation generation with explainable reasoning over augmented graphs. 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 1782–1792, 2019.

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

[27] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

[28] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[29] Kristina Toutanova and Danqi Chen. 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, 2015.

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

[31] Abraham A Ungar. Hyperbolic trigonometry and its application in the poincaré ball model of hyperbolic geometry. *Computers & Mathematics with Applications*, 41(1-2):135–147, 2001.

[32] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082*, 2019.

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

[34] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 968–977, 2019.

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

[36] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5329–5336, 2019.

[37] Xiang Wang, Yaokun Xu, Xiangnan He, Yixin Cao, Meng Wang, and Tat-Seng Chua. Reinforced negative sampling over knowledge graph for recommendation. In *Proceedings of The Web Conference 2020*, pages 99–109, 2020.

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

[39] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Representation learning of knowledge graphs with hierarchical types. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2965–2971, 2016.

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

[41] Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems*, pages 2319–2328, 2017.

[42] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Thirty-Fourth AAAI conference on artificial intelligence*, 2020.

[43] Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. Knowledge graph embedding with hierarchical relation structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3198–3207, 2018.