Information Sciences 662 (2024) 120197

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

CoPE: Composition-based Poincaré embeddings for link prediction in knowledge graphs

Adnan Zeb [a,b], Summaya Saif [c], Junde Chen [d], James Jianqiao Yu [b,e], Qingshan Jiang [f], Defu Zhang [a,*]

[a] School of Informatics, Xiamen University, Xiamen 361005, Fujian, China
[b] Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518000, Guangdong, China
[c] Department of Mathematics, COMSATS University, Islamabad 44000, Pakistan
[d] Dale E. and Sarah Ann Fowler School of Engineering, Chapman University, United States
[e] Department of Computer Science, University of York, YO10 5DD York, United Kingdom
[f] Shenzhen Key Laboratory for High Performance Data Mining, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

ARTICLE INFO

Keywords:
Knowledge graph
Link prediction
Hyperbolic geometry
Poincaré embeddings
Compositional operators

ABSTRACT

Knowledge graph (KG) embedding methods predict missing links by computing the similarities between entities. The existing embedding methods are designed with either shallow or deep architectures. Shallow methods are scalable to large KGs but are limited in capturing fine-grained semantics. Deep methods can capture rich semantic interactions, but they require numerous model parameters. This study proposes a novel embedding model that effectively combines the strengths of both shallow and deep models. In particular, the proposed model adopts the design principles of shallow models and incorporates an expressive compositional operator inspired by deep models. This approach maintains the scalability while significantly enhancing the expressive capacity of the proposed model. Moreover, the proposed model learns embeddings using the Poincaré ball model of hyperbolic geometry to preserve the hierarchies between entities. The experimental results demonstrated the effectiveness of learning Poincaré embeddings with an expressive compositional operator. Notably, a substantial improvement of 2.4% in the Mean Reciprocal Rank (MRR) and a 1.4% improvement in hit@1 was observed on the CoDEx-m and CoDEx-s datasets, respectively, when compared to the current state-of-the-art methods. The proposed model was implemented using PyTorch 1.8.1, and experiments were conducted on a server with an NVIDIA GeForce RTX 2080 Ti GPU.

1. Introduction

Knowledge graphs (KGs) represent factual information in the form of relationship triples, typically denoted as (*subject entity*, *relation*, *object entity*). For instance, consider that 'Martin Scorsese produced Hugo'. In the context of the KG, this fact is encapsulated as a triple (*Martin Scorsese*, *produced*, *Hugo*). This triple encompasses the subject entity '*Martin Scorsese*', the object entity '*Hugo*', and the relation '*produced*' that links the subject and object entities. A collection of such triples forms a KG, where nodes correspond to

* Corresponding author.
E-mail address: dfzhang@xmu.edu.cn (D. Zhang).

https://doi.org/10.1016/j.ins.2024.120197
Received 3 January 2023; Received in revised form 16 January 2024; Accepted 20 January 2024

entities and edges symbolize the relations between entities. This graph-based knowledge representation addresses various challenges in handling heterogeneous data with complex linked structures and supports numerous applications in dialogue generation, semantic search, recommendations, and question answering [47].

KG construction approaches typically focus on extracting triples from multiple knowledge sources. These approaches utilize ontological models to link the extracted triples into a graphical representation [1]. However, these triples are directly sourced from the available knowledge repositories, which are typically incomplete, resulting in incomplete KGs. Consequently, KG completion, also referred to as link prediction, has recently gained prominence as a significant research direction in knowledge representation learning. The primary objective of KG completion is to utilize existing triples to infer new unobserved triples and seamlessly integrate them into the KGs. This process maximizes the available knowledge within the KGs and enhances the performance of learning models across various downstream tasks.

Among existing solutions, embedding methods are widely recognized as state-of-the-art approaches for link prediction in KGs. These embedding methods consist of two key components: a mapping function and a scoring function. The first component maps entities and relations to low-dimensional vectors. The latter component utilizes compositional operators to compute composite vectors for entities, and matches them to compute the probabilities of triples as valid facts [44].

Embedding models can be broadly classified into two main categories: shallow models [31,46,30,42,23,17,7,22] and deep models [12,6,39,11,27,48,4]. Shallow models generate a single feature per embedding parameter and employ simple compositional operators, such as elementwise multiplication and addition to embeddings to model the interactions among entities. These methods require fewer parameters, which enables them to scale effectively to large KGs. However, their limited expressive power hinders their capability of capturing rich semantic relationships among entities, subsequently restricting their predictive performance in link prediction tasks [11,33]. To enhance the expressiveness of shallow models, a common approach is to increase the dimensionality of the embeddings. This increment increase provides a greater capacity to capture rich semantic interactions. However, this also increases the memory and computational demands, leading to scalability challenges when dealing with large KGs [42,11,9].

To address the limitations of shallow models, deep models [11,4,38] employ more expressive compositional operators such as concatenation, convolution, and projection on embeddings. These operators are employed to generate multiple feature maps for learning more expressive embeddings. However, these deep models are more complex in terms of model parameters, because increasing the number of feature maps also increases the number of parameters, making deep methods less efficient for large KGs [33].

Another substantial problem with many existing methods is the generation of similar representations for entities in both the subject and object positions. This limitation hampers their ability to distinguish between valid and invalid triples, particularly in the case of asymmetric relations, which constitute a major proportion of real-world KGs. In semantic modelling tasks, it is crucial to represent an entity differently when it appears as a subject than when it appears as an object to effectively discern valid from invalid facts. For instance, in the case of an asymmetric relation such as *won*, the triples (*Martin Scorsese*, *won*, *Academy Award*) and (*Academy Award*, *won*, *Martin Scorsese*) are not semantically equivalent. However, they may receive nearly identical treatment when the same representations are used for entities in both the subject and object positions. To address this problem, several existing deep models [11,38,4] generate a composite vector solely for the subject entity from subject-relation embeddings to ensure that the subject vector remains different from the object vector. However, these models failed to capture the object-relation interactions before matching the object with the subject entity. By contrast, a few shallow models, such as Canonical Polyadic (CP) [14] and SIMPLE [17], address this problem by creating two separate embedding vectors for each entity, one for the subject position and the other for the object position. While this approach is effective in discerning valid from invalid triples, it doubles the number of entity parameters. In addition, both CP and SIMPLE focus solely on the positional information of entities and neglect the contextual aspects of relations. To address this issue, ContE [34] utilizes two distinct relational contexts for learning entity embeddings: entity representations at subject and object positions 1) within the same relation and 2) across different relations. This setting enabled ContE to capture the contextual information of entities in terms of their relationships. However, ContE employs similar compositional operators for both subject-relation and object-relation embeddings, potentially causing subject and object representations to converge closely in the latent space, thereby limiting predictive performance.

Based on the aforementioned observations, the majority of existing methods lack either an effective or efficient mechanism for representing the subject vector differently from the object vector within the scoring function. This distinction is vital to distinguishing valid triples from their corresponding invalid counterparts. Additionally, deep methods tend to prioritize model expressivity, potentially sacrificing network scalability. In contrast, shallow methods tend to prioritize scalability but at the cost of losing model expressivity. Consequently, existing embedding methods lack a balanced trade-off between these two crucial properties for an accurate and efficient link prediction.

To address these issues, this paper proposes a novel link prediction model that adopts design of the shallow models while incorporating an expressive compositional operator: concatenation with projection from deep models. This combination allows the proposed model to learn more expressive embeddings while retaining its scalability. In particular, the compositional operator, concatenation with projection, was utilized in the subject-relation embeddings to capture the rich semantic interactions between them into a composite vector. In addition, a simple compositional operator, addition, was utilized in the object-relation embeddings to create a composite vector for the object entity. Finally, the distance between these composite vectors was computed to measure the similarity score of a given triple. These distinct compositional operators enable the proposed method to distinguish between valid and invalid triples by generating distinct scores.

Table 1
Major notations.

Notation	Explanation
$\mathcal{G}$	knowledge graph
$\mathcal{E}, \mathcal{R}$	set of entities, set of relations
$\mathbf{E}, \mathbf{R}$	entity embedding matrix, relation embedding matrix
$\mathbb{R}$	Euclidean space
$\mathbb{B}$	hyperbolic space
$\mathbb{T}$	tangent space
$\mathbf{e}_s, \mathbf{e}_o, \mathbf{r}$	subject, object, and relation embedding vectors (Euclidean)
$\mathbf{h}_s, \mathbf{h}_o, \mathbf{q}$	subject, object, and relation embedding vectors (Poincaré)
$\mathbf{W}$	projection matrix
k	curvature
σ	nonlinear activation
$\circ$	compositional operator
$\star$	concatenation
$*$	convolution
$\odot$	Multiplication; elementwise (ew), matrix-vector (mv)
ϕ	scoring function
$\oplus$	Möbius addition
$\otimes$	Möbius matrix-vector product

In addition, KGs are typically composed of many rich hierarchical structures, that is, sets of entities and relations that often include multiple hierarchies, each of which requires different geometric patterns for representation in the embedding space [9,3]. Recent research on hierarchical data, such as social networks [28], block-chain financial networks [24], protein-protein interaction networks [10], and multi-relational graphs [9,3], suggests that hyperbolic space can represent hierarchical structures more effectively than regular Euclidean space. In particular, the hyperbolic space is considered a continuous counterpart of discrete trees, making it naturally suitable for representing tree-like hierarchical structures [28]. In the case of tree-like structures, Euclidean space typically exhibits linear growth with respect to the radius of the circle, whereas hyperbolic space exhibits exponential growth [3,28]. This property makes hyperbolic space a natural fit for modelling tree-like structures, as standard trees grow exponentially. Motivated by these works, the proposed model was extended to hyperbolic space by conducting composition operations using the Poincaré ball model of hyperbolic geometry. This extension aims to capture hierarchical relations in KGs and learn more expressive KG embeddings.

Our specific contributions are summarized as follows:

- A novel Euclidean link prediction model called CoEE is proposed that incorporates an expressive compositional operator, concatenation with projection, into the design of shallow models. This integration aimed to merge the expressive capacity of deep models with the scalability of shallow models into a unified scoring function. Specifically, compositional operator concatenation with projection was defined on the subject-relation embeddings, and addition was employed on the object-relation embeddings. This approach constructs distinct composite vectors for an entity at both subject and object positions.

- An analogue of the proposed Euclidean model in the hyperbolic space, called CoPE, is introduced. CoPE performs composition operations using the Poincaré ball model of hyperbolic geometry to enrich the embeddings with hierarchical patterns present in the KGs.

- A detailed comparison with closely related models was performed in terms of compositional operators and parameter efficiency.

- Multiple experiments are conducted on three challenging datasets to validate the effectiveness of the proposed method.

The remainder of this paper is structured as follows: Section 2 discusses the background and preliminary concepts required to define the proposed model. Section 3 presents the proposed Euclidean and hyperbolic scoring functions and a detailed comparison with closely related methods. Section 4 discusses the experimental settings and presents the results of the link prediction task. Finally, Section 5 summarizes the study, provides concluding remarks, and outlines future directions.

2. Related background and preliminary concepts

This section provides a brief background on link prediction and existing compositional vector space models. Subsequently, it reviews the key concepts of hyperbolic geometry. The major notations used are listed in Table 1.

2.1. Link prediction

Let $\mathcal{G} = (\mathcal{E}, \mathcal{R})$ represents a KG, where $\mathcal{E}$ denotes the set of entities/nodes, and $\mathcal{R}$ denotes the set of relations/edges. In KGs, entities and relations are linked in the form of relationship triples $(e_s, r, e_o) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where each triple depicts a knowledge

fact using a relation $r \in \mathcal{R}$ between the subject and object entities $e_s, e_o \in \mathcal{E}$. The primary goal is to map the sets of entities $\mathcal{E}$ and relations $\mathcal{R}$ into a low-dimensional space, with $\mathbf{E} \in \mathbb{R}^{|\mathcal{E}| \times d}$ and $\mathbf{R} \in \mathbb{R}^{|\mathcal{R}| \times d}$ representing the entity and relation embedding matrices, respectively. In these matrices, each row corresponds to a d-dimensional embedding vector. Link prediction is a KG completion task that aims to predict missing subjects $(?, r, e_o)$ or objects $(e_s, r, ?)$ to infer missing triples. To achieve this, embedding models conduct a lookup operation upon the embedding matrices $\mathbf{E} \in \mathbb{R}^{|\mathcal{E}| \times d}$ and $\mathbf{R} \in \mathbb{R}^{|\mathcal{R}| \times d}$ to retrieve the entity and relation embeddings $\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^d$ and $\mathbf{r} \in \mathbb{R}^d$, respectively. Subsequently, a scoring function ϕ is defined over these embeddings to score each triple, indicating the probability that a triple represents a valid fact.

2.2. Link prediction models

This section reviews various shallow and deep link prediction models in terms of composite representations. The compositional operator is denoted as $\circ$. For example, the composite vector for the subject entity obtained from the subject-relation embeddings is denoted by $\mathbf{e}_s \circ \mathbf{r}$. This composite vector is then matched with the object embedding vector $\mathbf{e}_o$ using either a product or distance metric to measure the plausibility of a given triple [30]. Based on the compositional operators and similarity metrics, the existing models are categorized into product-, distance-, and neural network-based models. A summary of these methods is provided in Table 2.

2.2.1. Neural network-based models

Neural network-based models primarily compute subject-relation composite vectors through concatenation, convolution, and projection. Let $\star$ denotes the concatenation and * denotes the convolution. The subject-relation composite vector is then computed as $\mathbf{e}_s \circ \mathbf{r} = vec((\mathbf{e}_s \star \mathbf{r}) * \psi)\mathbf{W}$, where vec represents matrix-to-vector reshaping, ψ is the convolutional kernel, and $\mathbf{W}$ is a projection matrix. The projection matrix is adaptively learned in conjunction with entity and relation embeddings. ConvE [11] is the first state-of-the-art model based on this composition, and defines the scoring function as follows:

$$\phi(e_s, r, e_o) = \sigma(vec(\sigma((\underline{\mathbf{e}_s} \star \underline{\mathbf{r}}) * \psi))\mathbf{W})\,\mathbf{e}_o, \tag{1}$$

where σ denotes a nonlinear activation function. ConvE first reshapes the 1D subject and relation embeddings $\mathbf{e}_s$ and $\mathbf{r}$, into 2D-shaped embeddings, $\underline{\mathbf{e}_s}$ and $\underline{\mathbf{r}}$, respectively, and subsequently concatenates them. It then applies convolutional filters ψ to the concatenated embeddings, which results in multiple feature maps. The feature maps were reshaped into a vector and passed through the projection matrix $\mathbf{W}$ to capture further subject-relation interactions. The resulting composite representation was then matched with the object embedding vector $\mathbf{e}_o$ to compute the similarity score for a given triple (see Fig. 1).

Despite its solid predictive performance, the use of 2D convolution in ConvE seems nonintuitive, given that word embeddings are inherently structured in 1D [4]. Following this general idea, Conv-TransE [38] and ConvKB [27] extended ConvE by employing 1D convolution and enforcing the translational property of TransE in embeddings. ConvRot [19] extended ConvE with a complex space rotation to ensure that the model captures various relation patterns in the KGs. HypER [4] constructs 1D convolutional filters from relation embeddings, applies them to subject embeddings, and uses a projection matrix on the resulting embeddings to create a composite subject-relation vector. By contrast, CNN-ECFA [15] enhances the expressive capacity of deep networks such as ConvE by incorporating entity-specific standard features. Although these deep approaches are effective at capturing rich entity-relation interactions, the generation of multiple feature maps restricts the scalability of deep models to small KGs.

2.2.2. Distance-based models

Distance-based models create a subject-relation composite vector using the addition operator $\mathbf{e}_s \circ \mathbf{r} = \mathbf{e}_s + \mathbf{r}$, and subsequently match this composite vector with the object embedding vector $\mathbf{e}_s$ using a distance function $\text{dist}_{\mathbb{R}}$ to quantify the similarity between them. In these methods, the relation embedding vector $\mathbf{r}$ serves as a translation vector, shifting the subject embedding vector $\mathbf{e}_s$ towards object embedding vector $\mathbf{e}_o$. Many distance-based models have been built based on this general idea of vector space translation, including state-of-the-art models such as TransE [7], TransM [13], TransR [22], and TransD [16]. TransE defines a subject-relation composition-based scoring function as follows:

$$\phi(e_s, r, e_o) = -\text{dist}_{\mathbb{R}}(\mathbf{e}_s + \mathbf{r}, \mathbf{e}_o). \tag{2}$$

TransE is renowned for its parameter efficiency, which makes it suitable for large KGs. However, it is limited to modelling only 1-1 relations. TransM [13] extends TransE by assigning relation-specific weights to each triple. These weights enable TransM to model 1-N, N-1, and N-N relations by reducing the weight values to push $\mathbf{e}_o$ away from $\mathbf{e}_s + \mathbf{r}$. To further extend TransE's capability to handle 1-N, N-1, and N-N relations, TransR introduces relation-specific transformations for subject and object entities by employing a relation-specific transformation matrix $\mathbf{M}_r \in \mathbb{R}^{d \times d}$. Following these transformations, a regular translational composition was performed to compute the score for a given triple.

TransD [16] defines the scoring function in a manner similar to TransR. However, it introduces two projection matrices, $\mathbf{M}_{r_1}$ and $\mathbf{M}_{r_2}$, and decomposes each into a product of two vectors to create relation-specific projections for both the subject and object entities. These existing translational models demonstrate promising performances. However, they fall short of capturing the complex patterns present in KGs. To this end, RotatE [40] replaces the translation operation in TransE with a complex space rotation via an elementwise product between the subject and relation embeddings. This rotation operation enables RotatE to model various complex patterns, including asymmetric relationships. RotatE4D [20] extends RotatE by performing 4D rotations in quaternion space to model

Table 2
Summary of existing work.

Category	Model	Architecture	Embedding space	Compositional operator
Neural network-based models	ConvE [11]	Deep	Real	Concatenation, convolution, and projection
	Conv-TransE [38]	Deep	Real	Concatenation, convolution, and projection
	ConvKB [27]	Deep	Real	Concatenation, convolution, and projection
	HypER [4]	Deep	Real	Convolution and projection
	ConvRot [19]	Deep	Complex	Convolution and Complex rotation
	CNN-ECFA [15]	Deep	Real	Concatenation, convolution, and projection
Distance-based models	TransE [7]	Shallow	Real	Addition
	TransM [13]	Shallow	Real	Addition
	TransR [22]	Shallow	Real	Addition
	TransD [16]	Shallow	Real	Addition
	RotatE [40]	Shallow	Complex	Elementwise product
	RotatE4D [20]	Shallow	Quaternion	Rotation with quaternion
	MuRP [3]	Shallow	Hyperbolic	Elementwise product and Möbius addition
Product-based models	RESCAL [31]	Shallow	Real	Matrix-vector product
	DistMult [46]	Shallow	Real	Elementwise product
	Hole [30]	Shallow	Real	Circular correlation
	ComplEx [42]	Shallow	Complex	Hermitian product
	QuatE [50]	Shallow	Quaternion	Hamilton product
	QIQE-KGC [21]	Shallow	Quaternion	Hamilton product
	ANALOGY [23]	Shallow	Real	Matrix-vector product
	CP [8]	Shallow	Real	Elementwise product
	SIMPLE [17]	Shallow	Real	Elementwise product
	CKRL [36]	Shallow	Real	Concatenation, projection, and correlation

Note. The compositional operators listed in the table are used to create composite vectors for either subject-relation or object-relation embeddings, except for Hole, ConvKB, and CapsE. Hole employs circular correlation on subject and object embeddings, followed by computing the dot product with relation embeddings. ConvKB and CapsE combine both subject and object entities with relations in the convolution process.

highly complex and hierarchical relations. By contrast, MuRP [3] adopts a distinct approach by extending the translation operation to the hyperbolic space, enabling it to effectively model hierarchical patterns within KGs.

2.2.3. Product-based models

For a given triple, product-based models capture semantic interactions between the features of the subject and the relation by creating a composite vector through an elementwise/matrix-vector product[1] denoted by $\mathbf{e}_s \circ \mathbf{r} = \mathbf{e}_s \odot \mathbf{r}$. Subsequently, this composite vector is matched with the object embedding vector $\mathbf{e}_o$ using a product metric $\text{prod}_{\mathbb{R}}$ such as the generalized dot product[2] to compute the similarity score for each triple. RESCAL [31] is a representative model of product composition. It embeds subject and object entities into vectors $\mathbf{e}_s \in \mathbb{R}^d$ and $\mathbf{e}_o \in \mathbb{R}^d$, the relation into a matrix $\mathbf{M}_r \in \mathbb{R}^{d \times d}$, and computes the subject-relation composition using the matrix-vector (mv) product:

$$\phi(e_s, r, e_o) = \text{prod}_{\mathbb{R}}(\mathbf{e}_s \odot^{mv} \mathbf{M}_r, \mathbf{e}_o). \tag{3}$$

Although RESCAL is one of the most expressive methods with a shallow architecture, its main drawback is that the relation embedding matrix requires $\mathcal{O}(d^2)$ parameters per relation. This large number of parameters demands increased memory and computational power and may lead to overfitting, which limits predictive performance [46]. ANALOGY [23] implemented a scoring function similar to RESCAL but introduced normality and commutativity constraints on the relation matrices. These constraints are deployed to model analogical structures in KGs.

DistMult [46] extended the general idea of product composition in RESCAL and ANALOGY by replacing the relation embedding matrix with a relation embedding vector $\mathbf{r}$, thereby reducing the number of relation parameters to $\mathcal{O}(d)$. However, this reduction in the number of relation parameters imposes symmetry on embeddings, limiting DistMult's ability to asymmetric relations [42]. To model asymmetric relations, CP [17,14] extends DistMult by defining two embedding vectors for each node: one for the subject position and the other for the object position. SIMPLE [17] further extends the CP by including the reciprocal of its scoring function to generate triple scores. ComplEx [42] proposed an alternative approach for modelling asymmetric relations by extending DistMult into a complex space, where the subject and object embedding vectors are considered complex conjugates of each other. QuatE [50] extended ComplEx to the quaternion space, enabling more expressive entities and relation embeddings by allowing more interactions between them. QIQE-KGC [21] combines quantum embeddings with quaternion embeddings to enforce logical inferences for modelling deep dependencies between entities and relations. CKRL [36] adopts a sophisticated linear approach to embed the relationship strengths between entities using three levels of latent spaces: entity, relation, and canonical spaces. It is worth noting that these models are also referred to as bilinear, as they can capture entity-relation interactions in both directions.

[1] The output of the elementwise/matrix-vector product is a vector.
[2] The output of the dot product is a scalar value.

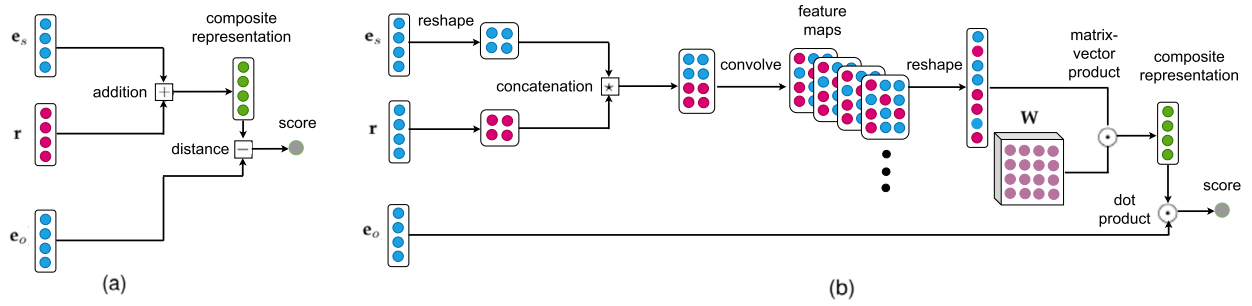

Fig. 1. Illustration of TransE (a) and ConvE (b).

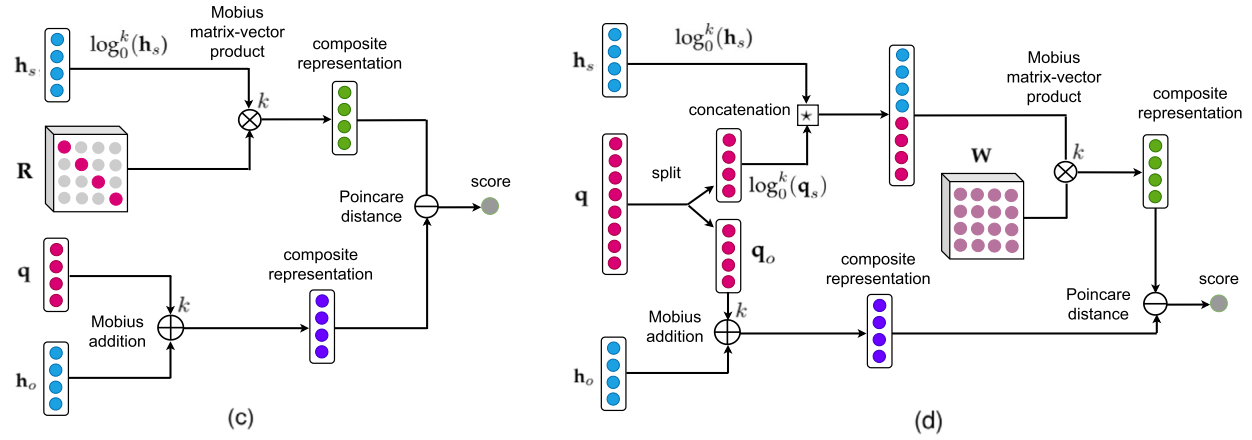

Fig. 2. Illustration of MuRP (c) and proposed CoPE (d).

In addition to the main categories mentioned above, other studies have conducted link prediction in KGs using different approaches. For example, SAttLE [2] utilizes a self-attention-based transformer as an encoder to generate context-aware latent entities and relation embeddings and then utilizes a regular KG embedding model as a decoder to predict missing links. SimRE [49] introduced a contrastive learning approach to integrate the logical and textual features of entities to improve the model expressiveness.

2.3. Poincaré ball model of hyperbolic geometry

Hyperbolic space is a negatively curved space with a constant negative curvature. There are five hyperbolic geometry models, including the Poincaré ball model [8], hyperboloid model [35], and the Klein disk model [32]. In this work, we utilized the Poincaré ball model, which exhibits all the properties of hyperbolic space and is more suitable for gradient-based optimization than other models of hyperbolic geometry [3,28]. The d-dimensional Poincaré ball $\mathbb{B}^{d,k}$ with a radius of $(\frac{1}{\sqrt{k}}, k>0)$ is defined by the manifold $\mathbb{B}^{d,k} = \{\mathrm{x} \in \mathbb{R}^d : ||\mathrm{x}||^2 < \frac{1}{k}\}$. Some basic mathematical operations in the Poincaré ball model are defined as follows:

- **Möbius addition:** Möbius addition [43] is the hyperbolic counterpart of standard Euclidean addition and is defined for two hyperbolic vectors $\mathbf{x}, \mathbf{y} \in \mathbb{B}^{d,k}$ as follows:

$$\mathbf{x} \oplus^k \mathbf{y} = \frac{(1 + 2k(\mathbf{x} \odot^{ew} \mathbf{y}) + k||\mathbf{y}||^2)\mathbf{x} + (1 - k||\mathbf{x}||^2)\mathbf{y}}{1 + 2k(\mathbf{x} \odot^{ew} \mathbf{y}) + k^2||\mathbf{x}||^2||\mathbf{y}||^2}, \tag{4}$$

  where $||.||$ represents the Euclidean norm and $\odot^{ew}$ represents elementwise multiplication in the Euclidean space.
- **Poincaré distance:** The Poincaré distance between two hyperbolic vectors $\mathbf{x} \in \mathbb{B}^{d,k}$ and $\mathbf{y} \in \mathbb{B}^{d,k}$ is commonly defined using Möbius addition:

$$\mathrm{dist}_{\mathbb{B}}(\mathbf{x}, \mathbf{y}) = \frac{2}{\sqrt{k}} \tanh^{-1}\left(\sqrt{k}||-\mathbf{x} \oplus^k \mathbf{y}||\right), \tag{5}$$

  where $\mathrm{dist}_{\mathbb{B}}$ represents Poincaré distance.
- **Logarithmic and exponential maps:** Some fundamental mathematical operations, such as elementwise and matrix-vector products, are not well defined in hyperbolic geometry [3]. In hyperbolic geometry, these operations are performed in the

tangent space $\mathbb{T}_{\mathbf{x}}^k$, which is a d-dimensional Euclidean space associated with each hyperbolic model. The mapping between the hyperbolic and tangent spaces is achieved using logarithmic and exponential maps. In particular, the logarithmic map projects a point from hyperbolic space to the tangent space at 0, denoted as $\log_0^k : \mathbb{B}^{d,k} \to \mathbb{T}_{\mathbf{x}}^k$, while the exponential map projects a point from the tangent space back to hyperbolic space at 0, denoted as $\exp_0^k : \mathbb{T}_{\mathbf{x}}^k \to \mathbb{B}^{d,k}$. The mappings between the Poincaré ball and tangent space for vector $\mathbf{x}$ are defined as follows:

$$\log_0^k(\mathbf{x}) = \tanh^{-1}\left(\sqrt{k}||\mathbf{x}||\right)\frac{\mathbf{x}}{\sqrt{k}||\mathbf{x}||}, \tag{6}$$

$$\exp_0^k(\mathbf{x}) = \tanh\left(\sqrt{k}||\mathbf{x}||\right)\frac{\mathbf{x}}{\sqrt{k}||\mathbf{x}||}. \tag{7}$$

- **Möbius matrix-vector product:** Möbius matrix-vector multiplication [24,10,43] between a vector $\mathbf{x}$ and a matrix $\mathbf{M}$ is commonly defined as follows:

$$\mathbf{M} \otimes^k \mathbf{x} = \exp_0^k(\mathbf{M}(\log_0^k(\mathbf{x})), \tag{8}$$

where the matrix $\mathbf{M}$ is defined in the tangent space. This process involves the following steps. The hyperbolic vector $\mathbf{x} \in \mathbb{B}^{d,k}$ is projected onto the tangent space by using the $\log_0^k$ map. Next, matrix-vector multiplication is performed between the projected vector and the tangent matrix within the tangent space. Finally, the resulting vector is projected back onto the Poincaré ball using the $\exp_0^k$ map.

3. Composition-based Poincaré embeddings

Having discussed the existing compositional vector space models and preliminaries, this section presents the proposed composition-based Poincaré embeddings (CoPE). This section begins by introducing the proposed method in the context of Euclidean space, called CoEE. It then describes how the CoEE is extended to the Poincaré manifold. Subsequently, a detailed discussion of the training and optimization of the CoPE is provided. Finally, CoPE and closely related methods were compared to highlight their relationships.

3.1. Composite representations on Euclidean manifold

In the previous section, we described various compositional operators that create composite vectors for entity-relation embedding. For example, a matrix-vector product is an effective compositional operator that captures the semantic similarity between the entity and relation embeddings in RESCAL. Subsequently, the composite vector is matched with the object embedding vector via a dot product to compute the similarity score of a given triple. This study aims to extend entity-relation compositions to hyperbolic space. However, the hyperbolic space lacks a clear correspondence with the Euclidean dot product [3]. Moreover, performing 2D convolution in hyperbolic space is a challenging task that we defer to in future work.

In particular, this study leverages concatenation with projection and addition compositions to capture the interactions between subjects and relations and objects and relations, respectively. As mentioned previously, most existing models, such as TransE, ConvE, Conv-TransE, and HypER, construct only composite vectors for the subject entities. This approach is counterintuitive, because both the subject and object entities in a given triple belong to the same entity space and are associated with the same relation. Therefore, it is more intuitive to create composite vectors for both entities to capture their interactions before measuring their similarities.

3.1.1. Euclidean scoring function

The proposed hyperbolic model, CoPE, was built based on the existing Euclidean compositions. Hence, it was initially constructed in Euclidean space and is referred to as CoEE. Specifically, for a given triple (e_s, r, e_o), the CoEE is constructed in the form of distance models [3], and its scoring function is defined as follows:

$$\begin{aligned}\phi(e_s, r, e_o) &= -\text{dist}_{\mathbb{R}}(\mathbf{e}_s^W, \mathbf{e}_o^r)^2 + b_s + b_o, \\ &= -\text{dist}_{\mathbb{R}}(\sigma(\mathbf{W}(\mathbf{e}_s \star \mathbf{r}_s)), \mathbf{e}_o + \mathbf{r}_o)^2 + b_s + b_o,\end{aligned} \tag{9}$$

where $\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^d$ are the Euclidean embedding vectors and $b_s, b_o \in \mathbb{R}$ are scalar biases for the subject and object entities. $\mathbf{r}_s \in \mathbb{R}^d$ and $\mathbf{r}_o \in \mathbb{R}^d$ denote the relation embeddings to be composed with the embedding vectors of the subject and object entities. These vectors are obtained by dividing the Euclidean relation vector $\mathbf{r} \in \mathbb{R}^c$ into two parts: the first $c/2$ dimensions are assigned to $\mathbf{r}_s$ and the remaining $c/2$ dimensions are assigned to $\mathbf{r}_o$, where $c = 2d$. $\text{dist}_{\mathbb{R}}$ denotes the Euclidean distance function, $\star$ denotes the concatenation operator, and $\mathbf{W} \in \mathbb{R}^{2d \times d}$ denotes the projection matrix. $\mathbf{e}_o^r$ denotes the composite vector of the object entity obtained by adding $\mathbf{e}_o$ to $\mathbf{r}_o$. $\mathbf{e}_s^W$ denotes the composite vector of the subject entity. It is formed by concatenating $\mathbf{e}_s$ with $\mathbf{r}_s$ and subsequently applying a matrix-vector product with the projection matrix $\mathbf{W} \in \mathbb{R}^{2d \times d}$, which adaptively learns the interactions between the subject and the relation.

To the best of our knowledge, learning subject-relation interactions using a projection matrix $\mathbf{W}$ in the design of shallow models is the novel contribution of this work. This innovation combined the expressiveness of deep models with the scalability of shallow

models using a unified scoring function. Furthermore, employing distinct compositional operators for subject- and object-relation embeddings allowed the proposed model to distinguish between valid and invalid triples. In Section 4.8, it is validated that utilizing the same compositional operators for both subject- and object-relation embeddings restricts the predictive performance.

3.2. Composite representations on Poincaré manifold

Although CoEE can capture rich entity-relation interactions, it is Euclidean with limited representation capabilities and cannot effectively capture the hierarchical relationships between KG nodes. In the case of hierarchical structures, the number of nodes grows exponentially with respect to the levels. However, in hyperbolic space, such structures can be easily modelled as the hyperbolic distance between two points increases exponentially when moving towards the boundary of the hypersphere [28,10]. In contrast, in Euclidean space, this growth is linear, making it unable to effectively represent increasingly complex hierarchies [3,28]. To handle the hierarchies between the KG nodes, the proposed Euclidean model, CoEE, defined in Eq. (9), is transformed into the Poincaré ball model of hyperbolic geometry and is called CoPE. This transformation involves three major steps: 1) mapping input embeddings from Euclidean to Poincaré's ball, 2) creating a composite vector for the subject entity by concatenation and projection operation on the Poincaré's manifold, and 3) creating a composite vector for the object entity using Möbius addition. Finally, the Poincaré distance between the composite vectors was computed, and the corresponding scalar biases were added to measure the similarity score for a given triple.

3.2.1. Euclidean to Poincaré ball mapping

Input embeddings are typically initialized in Euclidean space. Therefore, the input embedding vectors of the subject entity, object entity, and relation are first mapped from the Euclidean space to the Poincaré manifold using the exponential map as follows:

$$\mathbf{h}_s, \mathbf{h}_o, \mathbf{q} = \exp_0^k(\mathbf{e}_s), \exp_0^k(\mathbf{e}_o), \exp_0^k(\mathbf{r}), \tag{10}$$

where $\mathbf{h}_s, \mathbf{h}_o \in \mathbb{B}^{d,k}$ and $\mathbf{q} \in \mathbb{B}^{c,k}$ denote the corresponding Poincaré embeddings of the input Euclidean embedding vectors $\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^d$ and $\mathbf{r} \in \mathbb{R}^c$, respectively. Similar to $\mathbf{r}_s$ and $\mathbf{r}_o$ described in the previous section, $\mathbf{q}_s \in \mathbb{B}^{d,k}$ and $\mathbf{q}_o \in \mathbb{B}^{d,k}$ are created by splitting $\mathbf{q} \in \mathbb{B}^{c,k}$ into two halves, with the first $c/2$ dimensions assigned to $\mathbf{q}_s$ and the remaining $c/2$ dimensions allocated to $\mathbf{q}_o$. This division of the relation $\mathbf{q}$ allows the proposed method to create composite vectors for both the subject and object entities. This process is illustrated in Fig. 2.

3.2.2. Concatenation and projection on Poincaré manifold

For the subject-relation composite vector $\mathbf{e}_s^W$, defined in Eq. (9), the process initially involves concatenating the subject entity and the relation embedding vectors. This concatenated set of embeddings then conducts matrix-vector multiplication with the projection matrix $\mathbf{W}$. Such a matrix-vector product on the Poincaré manifold is obtained using the Möbius matrix-vector product, which is the hyperbolic counterpart of the Euclidean matrix-vector product and is specifically performed within the tangent space $\mathbb{T}_x^k$ associated with the Poincaré ball. To conduct Möbius matrix-vector multiplication, the projection matrix $\mathbf{W} \in \mathbb{R}^{2d \times d}$ is defined within the tangent space. The logarithmic map $\log_0^k$ shown in Eq. (6) is applied to both the subject $\mathbf{h}_s \in \mathbb{B}^{d,k}$ and relation $\mathbf{q}_s \in \mathbb{B}^{d,k}$ embeddings to map them from the Poincaré ball to the associated tangent space. Subsequently, the subject and relation embedding vectors are concatenated within the tangent space with a subsequent projection by matrix-vector multiplication with $\mathbf{W} \in \mathbb{R}^{2d \times d}$. Subsequently, a nonlinear activation function σ is applied to the resulting embeddings. Finally, these embeddings are mapped back to the Poincaré ball using the exponential map $\exp_0^k$ given as follows:

$$\begin{aligned} \mathbf{h}_s^W &= \mathbf{W} \otimes^k (\mathbf{h}_s \star \mathbf{q}_s), \\ &= \exp_0^k(\sigma(\mathbf{W}(\log_0^k(\mathbf{h}_s) \star \log_0^k(\mathbf{q}_s)))). \end{aligned} \tag{11}$$

3.2.3. Addition on Poincaré manifold

For the object-relation composite vector $\mathbf{e}_o^r$ in the proposed Euclidean scoring function, the relation embedding vector is added to the object embedding vector. Such an addition between the Poincaré embeddings of the object and relation can be seamlessly performed on the Poincaré manifold using Möbius addition as follows:

$$\begin{aligned} \mathbf{h}_o^q &= \mathbf{h}_o \oplus^k \mathbf{q}_o, \\ &= \frac{(1 + 2\langle \mathbf{h}_o \odot^{ew} \mathbf{q}_o \rangle + ||\mathbf{q}_o||^2)\mathbf{h}_o + (1 - ||\mathbf{h}_o||^2)\mathbf{q}_o}{1 + 2\langle \mathbf{h}_o \odot^{ew} \mathbf{q}_o \rangle + ||\mathbf{h}_o||^2||\mathbf{q}_o||^2}. \end{aligned} \tag{12}$$

3.2.4. Hyperbolic scoring function

Having introduced the input Euclidean to the Poincaré ball mapping in Eq. (10), and representing both the subject and object composite vectors on the Poincaré manifold in Eq. (11) and Eq. (12), we define the scoring function of CoPE. This scoring function is the hyperbolic counterpart of the proposed Euclidean scoring function

$$\begin{aligned} \phi(e_s, r, e_o) &= -\text{dist}_{\mathbb{B}}(\mathbf{h}_s^W, \mathbf{h}_o^q)^2 + b_s + b_o, \\ &= -\text{dist}_{\mathbb{B}}(\mathbf{W} \otimes^k (\mathbf{h}_s \star \mathbf{q}_s), \mathbf{h}_o \oplus^k \mathbf{q}_o)^2 + b_s + b_o, \end{aligned} \tag{13}$$

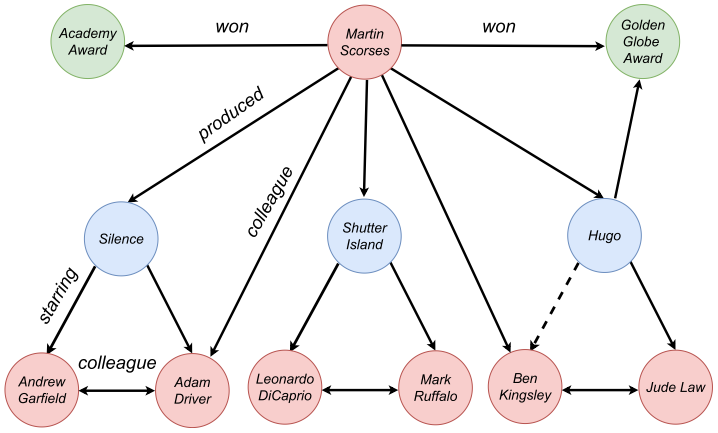

Fig. 3. A toy example of link prediction in KGs. Circles represent nodes, and arrows represent edges between nodes. Solid arrows represent existing triples, and a dashed arrow represents a missing triple. Link prediction aims to derive the missing triple (*Hugo*, *starring*, *Ben Kingsley*) based on existing triples in the given KG.

where $\text{dist}_{\mathbb{B}}$ denotes a hyperbolic distance function that computes the Poincaré distance between the composite vectors of the subject and object entities. Subsequently, the corresponding scalar biases were added to generate the final score ϕ for a given triple (e_s, r, e_o). It is worth noting that the Möbius addition and Möbius matrix-vector product do not add any new parameters. Hence, their parameter complexities were similar to those of their Euclidean counterparts.

3.3. Training and optimization

Link prediction is the task of predicting the missing relationships between entities in a given KG, as illustrated in Fig. 3. For link prediction, CoPE utilizes a standard preprocessing procedure that includes data augmentation [11,3] and negative sampling [7,3]. In the data augmentation, a reciprocal triple $z' = (e_o, r^{-1}, e_s)$ is added to the training set for each triple $z = (e_s, r, e_o)$. In negative sampling, either the object (e_s, r, e'_o) or the subject (e'_s, r, e_o) of each triple is replaced with a randomly chosen entity from the entity set $\mathcal{E}$ to generate negative/invalid triples z^n. This step is important for training a KG embedding model because KGs contain only valid triples.

In the first training step, the entity and relation embedding matrices $\mathbf{E} \in \mathbb{R}^{|\mathcal{E}| \times d}$ and $\mathbf{R}^{|\mathcal{R}| \times c}$ are initialized in the Euclidean space, where each row represents a Euclidean embedding vector. Subsequently, mini-batch training was conducted. For each triple in input batch B, a lookup operation is performed on the embedding matrices to retrieve the initial Euclidean embedding vectors $\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^d$ and $\mathbf{r} \in \mathbb{R}^c$. In the next step, the exponential map $\exp_0^k$ is used to project the initial embeddings from the Euclidean space to the corresponding Poincaré embeddings $\mathbf{h}_s, \mathbf{h}_o \in \mathbb{B}^{d,k}$, and $\mathbf{q} \in \mathbb{B}^{c,k}$. After splitting $\mathbf{q}$ into $\mathbf{q}_s \in \mathbb{B}^{d,k}$ and $\mathbf{q}_o \in \mathbb{B}^{d,k}$, these embeddings were given to the scoring function to produce a similarity score for a given triple. This score represents the probability of a triple being considered valid. Finally, the model was trained by minimizing the Bernoulli negative log-likelihood loss

$$\mathcal{L}(v, u) = -\frac{1}{|\mathcal{Z}|} \sum_{i=1}^{|\mathcal{Z}|} (v_i \log(u_i) + (1 - v_i) \log(1 - u_i)), \tag{14}$$

where $|\mathcal{Z}|$, u, and v denote the number of training triples, predicted probability, and binary label of the given triple, respectively.

As shown in Algorithm 1, the embedding parameters are typically initialized in Euclidean space. For optimization, one can choose between two established methods: tangent space optimization [10] and Riemannian optimization [28]. However, as mentioned in [10], Riemannian optimization is challenging and yields a slightly lower performance than tangent space optimization. Consequently, the CoPE was optimized using the tangent space optimization. In this approach, the embedding parameters are initialized in the tangent space, recovered through the $\exp_0^k$ map for hyperbolic operations, projected back to the tangent space via $\log_0^k$ map, and subsequently optimized using the Euclidean Adam [18].

3.4. Comparison with closely related models

This section compares CoPE with existing models that either learn two embedding vectors for each entity/relation or utilize distinct compositional operators for the subject- and object-relation embeddings. One such model is the Canonical Polyadic (CP) [17,14], which is a Euclidean model that creates two embedding vectors for each entity and defines a scoring function as follows:

$$\phi(e_s, r, e_o) = \text{prod}_{\mathbb{R}}(\mathbf{e}_s^s \odot^{ew} \mathbf{r}, \mathbf{e}_o^o). \tag{15}$$

The scoring function of CP is similar to that of DistMult, with the only difference being the learning of two embedding vectors, $\mathbf{e}_s^s$ and $\mathbf{e}_o^o$, for each entity. For instance, the embedding vector $\mathbf{e}_s^s$ is used when an entity appears as the subject of a triple and $\mathbf{e}_o^o$ is used when the same entity appears as an object in another triple. This construction enables the CP to differentiate between a valid triple

Algorithm 1 Training process of CoPE.

Input: Knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R})$; entity embedding size d; relation embedding size c; batch size (bs); scoring function ϕ; set of training triples $\mathcal{Z} = [z, z', z^n]$, where $z = (e_s, r, e_o)$, $z' = (e_o, r^{-1}, e_s)$, and z^n represents negative/invalid triples.
Output: Embedding matrices $\mathbf{E}, \mathbf{R}$, biases b_s, b_o, and projection matrix $\mathbf{W}$.

Initialization:
  $\mathbf{E} \leftarrow$ Xavier_Normal() ▷ $\forall e_s, e_o \in \mathcal{E}$
  $\mathbf{R} \leftarrow$ Xavier_Normal() ▷ $\forall r \in \mathcal{R}$
  $b_s, b_o \leftarrow$ Zeros() ▷ $\forall e_s, e_o \in \mathcal{E}$
  $\mathbf{W} \leftarrow$ Xavier_Normal() ▷ Projection matrix
for $epoch = 1, ..., n$ **do**
  $B \leftarrow$ Sample_batch($\mathcal{Z}, bs$)
  $loss \leftarrow 0$
  for $z \in B$ **do**
    $\mathbf{e}_s, \mathbf{e}_o, \mathbf{r} \leftarrow \text{Lookup}(\{\mathbf{E}, \mathbf{R}\}, z)$
    $\mathbf{h}_s, \mathbf{h}_o, \mathbf{q} \leftarrow \left(\exp_0^k(\mathbf{e}_s), \exp_0^k(\mathbf{e}_o), \exp_0^k(\mathbf{r})\right)$
    $\mathbf{q}_s, \mathbf{q}_o = \text{split}(\mathbf{q})$
    $u \leftarrow \text{Scoring\_Function}\left(\mathbf{h}_s, \mathbf{q}_s, \mathbf{h}_o, \mathbf{q}_o, \mathbf{W}, b_s, b_o\right)$ ▷ CoPE in Eq. (13)
    $loss \leftarrow \mathcal{L}(u, v)$
  end for
  Optimize and update $\mathbf{E}, \mathbf{R}, \mathbf{W}, b_s, b_o$ using Euclidean Adam.
end for

and its corresponding invalid triple by scoring them differently, particularly for asymmetric relations. It is worth noting that the CP learns both embedding vectors independently.

SIMPLE [17] is an extension of the CP that introduces an additional embedding vector $\mathbf{r}^{-1}$ as the inverse of the relation embedding vector. SIMPLE defines the scoring function as

$$\phi(e_s, r, e_o) = \frac{1}{2}(\text{prod}_{\mathbb{R}}(\mathbf{e}_s^s \odot^{ew} \mathbf{r}, \mathbf{e}_o^o) + \text{prod}_{\mathbb{R}}(\mathbf{e}_o^s \odot^{ew} \mathbf{r}^{-1}, \mathbf{e}_s^o)). \tag{16}$$

By introducing an inverse relation, SIMPLE learns the average of CP scores for the relation (e_s, r, e_o) and its reciprocal (e_o, r^{-1}, e_s). Despite their strong theoretical foundations, CP and SIMPLE do not outperform existing single-vector models on challenging datasets.

An effective solution to the problem of different representations has been efficiently addressed in the recent hyperbolic MuRP model [3], which creates distinct composite vectors for subject and object entities. The MuRP defines the scoring function as follows:

$$\phi(e_s, r, e_o) = -\text{dist}_{\mathbb{B}}(\mathbf{R} \otimes^k \mathbf{h}_s, \mathbf{h}_o \oplus^k \mathbf{q})^2 + b_s + b_o, \tag{17}$$

where $\mathbf{R} \in \mathbb{R}^{|\mathcal{R}| \times d \times d}$ is a diagonal relation matrix that can be interpreted as a relation embedding vector. MuRP composes $\mathbf{R}$ with the subject embedding vector using elementwise multiplication, similar to methods such as DistMult, CP, and SIMPLE, which capture weaker interactions than the projection matrix $\mathbf{W}$ in the proposed model. A comparison of CoPE and MuRP in terms of their compositional operators is illustrated in Fig. 2.

4. Experiments

This section provides details of the experimental settings, datasets, evaluation metrics, and experimental results. The results of the proposed models are compared with those of several existing state-of-the-art embedding methods. Additionally, several ablation experiments were conducted by introducing variants of the proposed model to demonstrate the effectiveness of each component.

4.1. Datasets details

FB15k-237 [11], WN18RR [41], NELL-995 [45] are classical KG completion datasets extracted from open-source KGs: Freebase [5], WordNet [25], and the Never-Ending Language Learner (NELL) system [26], respectively. Recently, Safavi and Koutra [37] found that existing KG completion datasets contain many generic triples with similar contents that are difficult to interpret. Therefore, they introduced a new benchmark, CoDEx [37], which comprises datasets, including CoDEx-s and CoDEx-m. Both datasets were extracted from Wikipedia with the aim of introducing more diverse and interpretable content to make the KG completion task more realistic and challenging. In these datasets, knowledge facts are represented by triples with nodes representing words and edges denoting the lexical relationships between them. These triples cover diverse domains, including business, geography, literature, media, entertainment, medicine, music, politics, science, visual arts, and sports, making them more diverse than other existing datasets. In this study, CoDEx-s and CoDEx-m, along with a traditional dataset, the Nations,[3] were employed to evaluate the predictive performance of CoPE. The statistics of these datasets are given in Table 3.

[3] https://github.com/ZhenfengLei/KGDatasets.

Table 3
Simulation Setup.

Parameters	CoDEx-s	CoDEx-m	Nations	Software/Hardware
$\|\mathcal{E}\|$	2034	17050	14	
$\|\mathcal{R}\|$	42	51	55	
$\|\mathcal{Z}\|$	32,888	185,584	1592	Python 3.7.6,
$\|\mathcal{V}\|$	1827	10,310	199	PyTorch 1.8.1,
$\|\mathcal{T}\|$	1828	10,310	201	Ubuntu 18.04.5,
bs	128	128	128	CUDA version 11.1,
d	200	400	200	Xeon(R) Silver 4210 CPU,
c	400	800	400	256 GB DDR4 RAM,
l_r	0.0001	0.0003	0.0001	NVIDIA GeForce RTX 2080 Ti GPU.
n	50	50	10	
ep	500	500	500	

Table 4
Link prediction results.

Dataset	CoDEx-m			CoDEx-s			Nations		
Model	MRR	hit@1	hit10	MRR	hit@1	hit@10	MRR	hit@1	hit@10
ConvE [11]	.318	.239	.464	.444	.343	**.635**	.830	.741	**1.0**
Conv-TransE [38]	.307	.235	.443	.427	.327	.617	.832	.743	**1.0**
HypER [4]	.316	.248	.443	.423	.326	.603	.776	.662	.995
CNN-ECFA (ConvE) [15]	.292	.221	.426	.408	.308	.594	.812	.709	.997
TransE [7]	.303	.223	.454	.354	.219	.634	.711	.587	.987
RESCAL [31]	.317	.244	.456	.404	.293	.623	.609	.423	.973
DistMult [46]	.307	.241	.432	.406	.301	.596	.749	.614	.990
ANALOGY [23]	.314	.246	.441	.423	.326	.609	.825	.731	.995
CP [14]	.296	.232	.412	.433	.345	.596	.641	.460	.982
SIMPLE [17]	.255	.189	.376	.379	.276	.574	.755	.634	.995
ATTH [9]	.315	.237	.464	.402	.286	.632	.785	.664	.995
MuRP [3]	.306	.226	.456	.420	.311	.632	.818	.726	**1.0**
CoPE (ours)	**.326**	**.251**	**.466**	**.446**	**.350**	.631	**.835**	**.754**	**1.0**

Note. The results of ConvE, RESCAL, and TransE are taken from [37]. The results of DistMult, ANALOGY, and CP are produced through the publically available code base [4]. For the remaining models, the results are produced using their original source codes.

4.2. Evaluation protocols

The performance of the KG embedding methods was evaluated using the MRR and hit@m metrics, where m $\in \{1, 10\}$. The MRR is computed by averaging the inverse ranks of all test triples, whereas hit@m is determined by computing the percentage of valid triples that appear in the top@m ranked triples. These metrics are computed under two settings: raw and filtered. This study specifically leverages the filtered setting, which is widely recommended because they remove all valid triples from the set of candidate triples. These candidate triples are derived by replacing either the subject or object of each test triple with each entity from the entity set $\mathcal{E}$.

4.3. Simulation setup

Table 3 provides a comprehensive overview of the simulation setup. This includes details on the statistics of the datasets, the software and hardware used for implementation, and the optimal hyperparameters adopted in the proposed model. These hyperparameters are determined through a manual search on the following parameters: entity embedding dimensions (d), relation embedding dimensions (c), batch size (bs), learning rate (lr), the number of invalid triples (n), and the number of training epochs (ep). The proposed model is trained using these hyperparameters on all three datasets, utilizing the default training, validation, and test splits denoted as $\mathcal{Z}$, $\mathcal{V}$, and $\mathcal{T}$, respectively.

4.4. Baselines

The effectiveness of the proposed model was assessed by comparing its experimental results with those of several state-of-the-art baseline methods. These baselines include both shallow models such as TransE [7], RESCAL [31], ANALOGY [23], DistMut [46], CP [14], SIMPLE [17], ATTH [9], and MuRP [3], and deep models such as ConvE [11], Conv-TransE [38], HypER [4], and CNN-ECFA (ConvE) [15].

Table 5
Lower dimensions results on CoDEx-s.

Model	Embedding size	No. of parameters	MRR	hit@1	hit@10
DistMult	60	127.2 K	.351	.234	.577
CP		249.3 K	.389	.285	.590
ConvE		283.3 K	.386	.279	.598
HypER		246.8 K	.374	.268	.587
CoPE		143.4 K	**.407**	**.302**	**.611**
DistMult	40	84.8 K	.319	.210	.526
CP		166.2 K	.364	.256	.573
ConvE		148.7 K	.357	.248	.575
HypER		139.7 K	.342	.242	.543
CoPE		95.4 K	**.380**	**.270**	**.591**
DistMult	20	42.4 K	.237	.152	.401
CP		83.1 K	.314	.211	.509
ConvE		56.4 K	.297	.194	.506
HypER		58.3 K	.275	.182	.466
CoPE		48.9 K	**.351**	**.242**	**.573**

4.5. Link prediction results and discussion

The objective of link prediction is to predict the entities that may be related to other entities within a KG. To achieve this, each test subject-relation pair is combined with each entity from the entity set $\mathcal{E}$ to generate a set of candidate triples. Subsequently, these candidate triples were fed into the trained model to obtain similarity scores. Once the candidate triples were scored, they were ranked based on these scores to compute the MRR and hit@m metrics.

Table 4 presents the results of the proposed model and compares them to the results of the reported baselines on the CoDEx-s, CoDEx-m, and nations datasets. The upper rows of the table contain the results of the deep models, while the lower rows include the results of the shallow models. Among the deep models, ConvE has the best results, primarily due to its 2D convolution, which captures more subject-relation interactions compared to the 1D convolution in HypER and Conv-TransE. ConvE outperforms HypER with improvements of 4.96% and 0.63% and Conv-TransE with improvements of 3.98% and 3.58% in MRR on the CoDEx-s and CoDEx-m datasets, respectively. For the shallow methods, the predictive performance of the models varies on both the CoDEx-s and CoDEx-m datasets. For instance, CP performs better on the CoDEx-s dataset, while RESCAL on the CoDEx-m dataset. However, despite their comparable performances, these shallow models substantially lag behind ConvE on both datasets. This comparison reveals that model expressivity is a crucial factor for achieving better performance in link prediction.

By contrast, CoPE combines significant features from both shallow and deep models, enabling it to outperform state-of-the-art methods in both categories. Specifically, CoPE surpassed the best-reported deep model, ConvE, with improvements of 4.8% and 2% in hit@1 for the CoDEx-m and CoDEx-s datasets, respectively. This substantial performance gain of CoPE can be attributed to the effectiveness of using distinct compositional operators along with the utilization of Poincaré embeddings and the expressive compositional operator on subject-relation embeddings. In the following sections, a series of ablation experiments are conducted to assess the contribution of these elements to the predictive performance.

4.6. Ablation on embedding dimensionality and parameter efficiency

An interesting observation regarding KG embedding methods is the trade-off between the embedding size and predictive performance. Euclidean models typically require high-dimensional embeddings to achieve considerable performance. In contrast, hyperbolic space can effectively embed complex relations with a smaller number of dimensions, as illustrated in [3]. In this section, an ablation experiment is conducted to compare the low-dimensional performance and parameter efficiency of CoPE with a few Euclidean models on the CoDEx-s dataset with dimension sizes $d \in \{20, 40, 60\}$. Table 5 shows that CoPE outperforms the Euclidean models by a more substantial margin in lower dimensions than in higher dimensions. Table 5 lists the required number of parameters for the given dimensions. Except for DistMult, CoPE required fewer parameters than the other reported Euclidean models. ConvE and HypER are deep models that produce multiple features per embedding parameter and thus require numerous parameters. CP, on the other hand, creates two embedding vectors for each entity, doubling the number of entity parameters.

4.7. Ablation on embedding space

To demonstrate the effectiveness of Poincaré embeddings, the proposed Euclidean model, CoEE, was evaluated on both the CoDEx-s and CoDEx-m datasets. From Fig. 4, it can be observed that except for hit@10 on CoDEx-s, CoPE outperformed CoEE across all metrics. This ablation experiment validated the capability of Poincaré embeddings to capture hierarchical relations, subsequently enhancing their predictive performance.

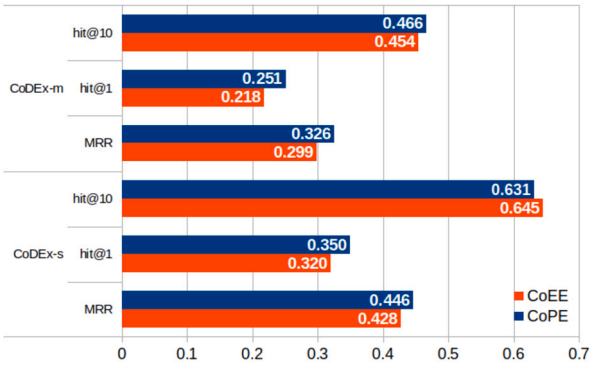

Fig. 4. CoPE vs. CoEE on CoDEx-m and CoDEx-s datasets.

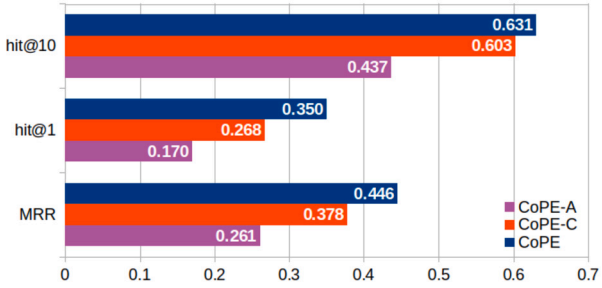

Fig. 5. CoPE vs. CoPE-A vs. CoPE-C on CoDEx-s dataset.

4.8. Ablation on similar compositions

To demonstrate the effectiveness of distinct composite representations, another ablation experiment was conducted in which two variants of CoPE were introduced, each employing similar compositional operators on both subject- and object-relation embeddings. The first variant utilizes the Möbius addition and is named CoPE-A:

$$\phi(e_s, r, e_o) = -\text{dist}_{\mathbb{B}}(\mathbf{h}_s \oplus^k \mathbf{q}_s, \mathbf{h}_o \oplus^k \mathbf{q}_o)^2 + b_s + b_o. \tag{18}$$

The second variant utilizes concatenation with projection and is named CoPE-C:

$$\phi(e_s, r, e_o) = -\text{dist}_{\mathbb{B}}(\mathbf{W} \otimes^k (\mathbf{h}_s \star \mathbf{q}_s), \mathbf{W} \otimes^k (\mathbf{h}_o \star \mathbf{q}_o))^2 + b_s + b_o. \tag{19}$$

Both of these variants are evaluated on the test set of the CoDEx-s dataset. As shown in Fig. 5, both variants experienced a substantial decrease in the MRR, hit@1, and hit@10 values. This ablation experiment provides valuable insights into the importance of employing distinct composite representations for entities in both subject and object positions.

4.9. Ablation on Möbius addition

In the previous section, we demonstrated the significance of generating distinct composite vectors for subject and object entities to enhance link prediction performance. CoPE uses a simple compositional operator to construct composite vectors for object entities. This choice was motivated by multiple compelling factors. First, addition views a relation as a translation operation between the subject and object entities, which has shown promising performance in various downstream tasks, including triple classification, entity alignment, and relation prediction [7,3]. Second, it can be easily implemented in hyperbolic space through its hyperbolic counterpart, the Möbius addition, compared to other compositional operators, such as the elementwise product. The elementwise product is performed in the tangent space and requires mapping between the hyperbolic and tangent spaces. Finally, the addition involves a minimal number of parameter computations, preserving the scalability of the proposed model.

To assess the performance of the addition operator, another ablation experiment was conducted, in which the Möbius addition was replaced with tangent-space elementwise product for object-relation embeddings. This variant was named CoPE-M. CoPE-M was evaluated on the test set of the CoDEx-s dataset, and it resulted in decreases of 7.5%, 8.3%, and 7.7% in the MRR, hit@1, and hit@10 values, respectively, compared with CoPE (see Fig. 6). This ablation experiment demonstrated that the object-relation composition using Möbius addition delivered superior results compared with the elementwise product.

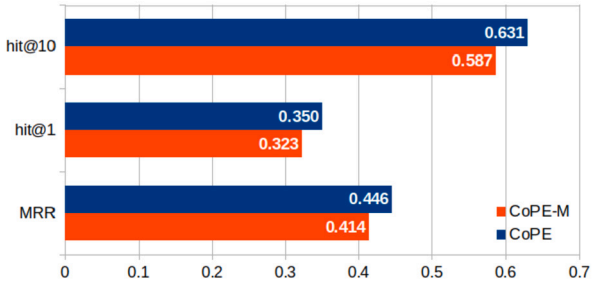

Fig. 6. CoPE vs. CoPE-M on CoDEx-s dataset.

5. Conclusions

This paper proposes CoPE as an effective link predictor for KGs based on hyperbolic geometry. CoPE implements an expressive compositional operator, concatenation with projection, with the design principles of shallow models. This unique approach enables the CoPE to merge the strengths of both shallow and deep models. Specifically, CoPE enhances the model expressivity for capturing complex patterns while maintaining scalability to large KGs. Furthermore, CoPE generates distinct compositional vectors for each entity in both the subject and object positions on the Poincaré manifold. This allows it to effectively handle both asymmetric and hierarchical relations within the KGs. Comprehensive experiments were conducted on three challenging datasets, demonstrating that CoPE outperforms many state-of-the-art shallow and deep methods in both higher and lower dimensions. Notably, CoPE achieves superior performance while requiring fewer model parameters than most existing competitors. Additionally, several ablation experiments were conducted to validate the parameter efficiency and performance contribution of each CoPE component.

Although the Poincaré model is well suited for gradient-based optimization and exhibits significant performance gain, it suffers from the problem of numerical instabilities in the embeddings owing to the fraction in the Poincaré distance [29]. In the future, we intend to extend the CoPE from the Poincaré model to the Lorentz model [29] and the hyperboloid model [10,35] of hyperbolic geometry. These models avoid fractions and prevent numerical instabilities in embeddings.

CRediT authorship contribution statement

Adnan Zeb: Writing – original draft, Methodology, Investigation, Conceptualization. **Summaya Saif:** Writing – review & editing. **Junde Chen:** Writing – review & editing. **James Jianqiao Yu:** Writing – review & editing, Resources. **Qingshan Jiang:** Writing – review & editing, Resources. **Defu Zhang:** Writing – review & editing, Supervision, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the author used ChatGPT in order to improve the language and readability of the previously written and reviewed text. After using this tool/service, the author reviewed and edited the content as needed and takes full responsibility for the content of the publication.

Acknowledgement

This work is partially supported by the National Natural Science Foundation of China (Grant no. 61672439).

References

[1] M. Alobaidi, K.M. Malik, M. Hussain, Automated ontology generation framework powered by linked biomedical ontologies for disease-drug domain, Comput. Methods Programs Biomed. 165 (2018) 117–128.

[2] P. Baghershahi, R. Hosseini, H. Moradi, Self-attention presents low-dimensional knowledge graph embeddings for link prediction, Knowl.-Based Syst. 260 (2023) 110124.
[3] I. Balazevic, C. Allen, T. Hospedales, Multi-relational Poincaré graph embeddings, in: Advances in Neural Information Processing Systems, 2019, pp. 4463–4473.
[4] I. Balažević, C. Allen, T.M. Hospedales, Hypernetwork knowledge graph embeddings, in: International Conference on Artificial Neural Networks, Springer, 2019, pp. 553–565.
[5] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, 2008, pp. 1247–1250.
[6] A. Bordes, X. Glorot, J. Weston, Y. Bengio, A semantic matching energy function for learning with multi-relational data, Mach. Learn. 94 (2) (2014) 233–259.
[7] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Advances in Neural Information Processing Systems, 2013, pp. 2787–2795.
[8] J.W. Cannon, W.J. Floyd, R. Kenyon, W.R. Parry, et al., Hyperbolic geometry, Flavors Geom. 31 (2) (1997) 59–115.
[9] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, C. Ré, Low-dimensional hyperbolic knowledge graph embeddings, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 6901–6914.
[10] I. Chami, R. Ying, C. Ré, J. Leskovec, Hyperbolic graph convolutional neural networks, in: Advances in Neural Information Processing Systems, 2019, pp. 4869–4880.
[11] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018, pp. 1811–1818.
[12] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, W. 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, 2014, pp. 601–610.
[13] M. Fan, Q. Zhou, E. Chang, T.F. Zheng, Transition-based knowledge graph embedding with relational mapping properties, in: Pacific Asia Conference on Language, Information and Computation, 2014, pp. 328–337.
[14] F.L. Hitchcock, The expression of a tensor or a polyadic as a sum of products, J. Math. Phys. 6 (1–4) (1927) 164–189.
[15] K. Hu, X. Zhu, H. Liu, Y. Qu, F.L. Wang, T. Hao, Convolutional neural network-based entity-specific common feature aggregation for knowledge graph embedding learning, IEEE Trans. Consum. Electron. (2023) 1–1, https://doi.org/10.1109/TCE.2023.3302297.
[16] G. Ji, S. He, L. Xu, K. Liu, J. 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, 2015, pp. 687–696.
[17] S.M. Kazemi, D. Poole, SimplE embedding for link prediction in knowledge graphs, in: Advances in Neural Information Processing Systems, 2018, pp. 4284–4295.
[18] D.P. Kingma, J. Ba Adam, A method for stochastic optimization, arXiv:1412.6980.
[19] T. Le, N. Le, B. Le, Knowledge graph embedding by relational rotation and complex convolution for link prediction, Expert Syst. Appl. 214 (2023) 119122.
[20] T. Le, H. Tran, B. Le, Knowledge graph embedding with the special orthogonal group in quaternion space for link prediction, Knowl.-Based Syst. 266 (2023) 110400.
[21] L. Li, X. Zhang, Z. Jin, C. Gao, R. Zhu, Y. Liang, Y. Ma, Knowledge graph completion method based on quantum embedding and quaternion interaction enhancement, Inf. Sci. 648 (2023) 119548.
[22] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence, 2015, pp. 2181–2187.
[23] H. Liu, Y. Wu, Y. Yang, Analogical inference for multi-relational embeddings, in: International Conference on Machine Learning, 2017, pp. 2168–2178.
[24] Q. Liu, M. Nickel, D. Kiela, Hyperbolic graph neural networks, in: Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019, pp. 8230–8241.
[25] G.A. Miller, WordNet: a lexical database for English, Commun. ACM 38 (11) (1995) 39–41.
[26] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, et al., Never-ending learning, Commun. ACM 61 (5) (2018) 103–115.
[27] T.D. Nguyen, D.Q. Nguyen, D. Phung, et al., A novel embedding model for knowledge base completion based on convolutional neural network, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2018, pp. 327–333.
[28] M. Nickel, D. Kiela, Poincaré embeddings for learning hierarchical representations, in: Advances in Neural Information Processing Systems, 2017, pp. 6338–6347.
[29] M. Nickel, D. Kiela, Learning continuous hierarchies in the Lorentz model of hyperbolic geometry, in: International Conference on Machine Learning, 2018, pp. 3779–3788.
[30] M. Nickel, L. Rosasco, T. Poggio, Holographic embeddings of knowledge graphs, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence, 2016, pp. 1955–1961.
[31] M. Nickel, V. Tresp, H.-P. Kriegel, A three-way model for collective learning on multi-relational data, in: Proceedings of the 28th International Conference on Machine Learning, 2011, pp. 809–816.
[32] F. Nielsen, R. Nock, Visualizing hyperbolic Voronoi diagrams, in: Proceedings of the 30th Annual Symposium on Computational Geometry, 2014, pp. 90–91.
[33] N. Ostapuk, J. Yang, P. Cudré-Mauroux, Activelink: deep active learning for link prediction in knowledge graphs, in: The World Wide Web Conference, 2019, pp. 1398–1408.
[34] F. Pu, Z. Zhang, Y. Feng, B. Yang, Learning context-based embeddings for knowledge graph completion, J. Data Inf. Sci. 7 (2) (2022) 84–106.
[35] W.F. Reynolds, Hyperbolic geometry on a hyperboloid, Am. Math. Mon. 100 (5) (1993) 442–455.
[36] M. Sabet, M. Pajoohan, M.R. Moosavi, Representation learning of knowledge graphs with correlation-based methods, Inf. Sci. 641 (2023) 119043.
[37] T. Safavi, D. Koutra, CoDEx: a comprehensive knowledge graph completion benchmark, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 8328–8350.
[38] C. Shang, Y. Tang, J. Huang, J. Bi, X. He, B. Zhou, End-to-end structure-aware convolutional networks for knowledge base completion, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 3060–3067.
[39] R. Socher, D. Chen, C.D. Manning, A. Ng, Reasoning with neural tensor networks for knowledge base completion, in: Advances in Neural Information Processing Systems, 2013, pp. 926–934.
[40] Z. Sun, Z.-H. Deng, J.-Y. Nie, J. Tang, RotatE: knowledge graph embedding by relational rotation in complex space, in: International Conference on Learning Representations, 2018.
[41] K. Toutanova, D. 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, 2015, pp. 57–66.
[42] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: Proceedings of the 33rd International Conference on Machine Learning, 2016, pp. 2071–2080.
[43] A.A. Ungar, Hyperbolic trigonometry and its application in the Poincaré ball model of hyperbolic geometry, Comput. Math. Appl. 41 (1–2) (2001) 135–147.
[44] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: a survey of approaches and applications, IEEE Trans. Knowl. Data Eng. 29 (12) (2017) 2724–2743.
[45] W. Xiong, T. Hoang, W.Y. Wang, DeepPath: a reinforcement learning method for knowledge graph reasoning, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 564–573.
[46] B. Yang, W. tau Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, in: International Conference on Learning Representations, 2015.

[47] A. Zeb, S. Saif, J. Chen, A.U. Haq, Z. Gong, D. Zhang, Complex graph convolutional network for link prediction in knowledge graphs, Expert Syst. Appl. 200 (2022) 116796.
[48] A. Zeb, S. Saif, J. Chen, D. Zhang, Learning knowledge graph embeddings by deep relational roto-reflection, Knowl.-Based Syst. 252 (2022) 109451.
[49] D. Zhang, Z. Rong, C. Xue, G. Li, SimRE: simple contrastive learning with soft logical rule for knowledge graph embedding, Inf. Sci. (2024) 120069.
[50] S. Zhang, Y. Tay, L. Yao, Q. Liu, Quaternion knowledge graph embeddings, in: Advances in Neural Information Processing Systems, 2019, pp. 2735–2745.