APSIPA Transactions on Signal and Information Processing, 2024, 13, e

Original Paper

Knowledge Graph Embedding with 3D Compound Geometric Transformations

Xiou Ge[1*], Yun Cheng Wang[1], Bin Wang[2] and C.-C. Jay Kuo[1]

[1] *University of Southern California, Los Angeles, CA, USA*
[2] *Institute for Infocomm Research (I2R), A*STAR, Singapore*

ABSTRACT

The cascade of 2D geometric transformations were exploited to model relations between entities in a knowledge graph (KG), leading to an effective KG embedding (KGE) model, CompoundE. Inspired by the recent trend in KGE designs that leverage multiple transformation from $SO(3)$ instead of $SE(2)$, we leverage 3D compound geometric transformations, including translation, rotation, scaling, reflection, and shear and propose a family of KGE models, named CompoundE3D, in this work. CompoundE3D allows multiple design variants to match rich underlying characteristics of a KG. We propose a beam search-based algorithm to locate the near-optimal embedding scoring function designs for different datasets in the vast search space resulted from different combinations of operator components. Since each variant has its own advantages on a subset of relations, an ensemble of multiple variants can yield superior performance. The effectiveness and flexibility of CompoundE3D are experimentally verified on four popular link prediction datasets.

Keywords: Knowledge graph embedding, link prediction, geometric transformation.

*Corresponding author: Xiou Ge, xiouge@usc.edu.

Received 22 November 2023; Revised 24 January 2024
ISSN 2161-1823; DOI 10.1561/116.00000177

1 Introduction

Knowledge graphs (KGs) find rich applications in knowledge management and discovery [74, 45, 75], recommendation systems [56, 81], fraud detection [76, 82], chatbots [27, 1], etc. KGs are directed relational graphs. They are formed by a collection of triples in form of (h, r, t), where h, r, and t denote head, relation, and tail, respectively. Heads and tails are called entities and represented by nodes while relations are links in KGs. KGs are often incomplete. One critical task in knowledge graph (KG) management is "missing link prediction". Knowledge graph embedding (KGE) methods have received a lot of attention in recent years due to their effectiveness in missing link prediction. According to [23], many KGs such as DBpedia [2], YAGO [50], Freebase [4], NELL [8], Wikidata [55], and ConceptNet [49] have been created and made publicly available for KGE model development and evaluation.

One family of KGE models builds a high-dimensional embedding space, where each entity is a vector. The relation is modeled by a certain geometric manipulation such as translation and rotation. To evaluate the likelihood of a candidate triple, the geometric manipulation associated with the relation is applied to the head entity and then the distance between the manipulated head and the tail is measured. The shorter the distance, the higher likelihood of the triple. To this end, these KGE models are called distance-based KGEs. Examples of distance-based KGEs include TransE [5], RotatE [51], and PairRE [10]. Each of them uses a single geometric transformation to represent relations between entities. Specifically, translation, rotation, and scaling operations are adopted by TransE, RotatE, and PairRE, respectively.

The above-mentioned KGE models achieve reasonably good performance in link prediction with only a single geometric transformation. The cascade of multiple 2D geometric transformations offers a powerful tool in image manipulation [43]. This idea was exploited to develop a new KGE model, called CompoundE, in [20]. TransE, RotatE and PairRE are all degenerate cases of CompoundE. Thus, CompoundE outperforms them in link prediction performance. CompoundE unifies translation, rotation, and scaling operations under one common framework. It has several mathematically provable properties that facilitate the modeling of different complex relation types in KGE. The effectiveness of these composite operators has been successfully demonstrated through extensive experiments and applications in downstream tasks such as entity typing and multihop query answering in [20]. Furthermore, a few recent KGEs [19, 70, 40] leverage $SO(3)$ rotations rather than $SO(2)$ rotations. $SO(3)$ rotations enable the KGEs to better encode non-commutative relations than RotatE which uses $SO(2)$ rotations, achieving more effective parameterization and endowing a model with greater modeling power.

Inspired by prior success, we wonder whether it would be beneficial to look for compound geometric transformations in the 3D space in the KGE model design. Here, we extend the CompoundE work in [20] along three directions. First, we include more affine operations beyond translation, rotation, and scaling such as reflection and shear. Second, we extend these geometric transformations from the 2D space to the 3D space and propose a family of KGE models, CompoundE3D. Third, CompoundE3D allows multiple design variants to match rich underlying characteristics of a KG. Since each variant has its own advantages on a subset of relations, an ensemble of multiple variants can yield superior performance. The effectiveness of CompoundE3D is experimentally verified on four popular link prediction datasets.

It is worthwhile to emphasize that we enhance CompoundE by addressing two critical issues. First, compound operations lead to numerous model variants, and it is unclear how to determine a scoring function that performs the best for a given dataset. Here, we propose an adapted beam search algorithm that builds more complex scoring functions from simple but effective ones gradually. Second, although ensemble learning is a popular strategy, it remains under-explored when it comes to building KGE models. In this work, we explore two ensemble strategies that potentially boost link prediction performance and allow different CompoundE3D variants to work together and complement each other. First, we implement a weighted sum of different scoring functions for link prediction. Second, we apply unsupervised rank aggregation functions to unify rank predictions from individual model variants. Both strategies help boost the ranking of valid candidate entities and reduce the impact of outliers.

The major contributions of this work are summarized below.

- We examine affine operations in the 3D space, instead of the 2D space, to allow more versatile relation representations. Besides translation, rotation, and scaling used in CompoundE, we include reflection and shear transformations which allow an even larger design space.
- We propose an adapted beam search algorithm to discover better model variants. Such a procedure avoids unnecessary exploration of poor variants but zooms into more effective ones to strike a good balance between model complexity and prediction performance.
- We analyze the properties of each operation and its advantage in modeling different relations. Our analysis is backed by empirical results on four datasets.
- To reduce errors of an individual model variant and boost the overall link prediction performance, we aggregate decisions from different variants with two approaches; namely, the sum of weighted distances and rank fusion.

The rest of this paper is organized as follows. A brief review of related work is provided in Section 2. The design methodology of CompoundE3D is explained and the decision ensemble of multiple model variants is elaborated in Section 3. Experimental results and performance benchmarking with previous work are presented in Section 4. Finally, concluding remarks and future research directions are given in Section 5.

2 Related Work

2.1 Knowledge Graph Embedding (KGE) Models

A large number of distance-based KGE models are derived by treating relations as certain transformations. They are briefly reviewed below.

2.1.1 2D Geometric Transformations

Quite a few KGE models are inspired by 2D geometric transformations such as translation, rotation, and scaling in the 2D plane. TransE [5] models the relation as a translation between head and tail entities. This simple model is not able to model symmetric relations effectively. RotatE [51] treats relations as certain rotations in the complex space, which works well for symmetric relations. Furthermore, RotatE introduces a self-adversarial negative sampling loss that improves distance-based KGE model performance significantly. PairRE [10] models relations with the scaling operation to allow variable margins. This is helpful in encoding complex relations. The unitary constraint on entity embedding in PairRE is also effective in practice. CompoundE [20] adopts compound geometric transformations, including translation, rotation, and scaling, to model different relations. It offers a superior KGE model without increasing the overall complexity much.

2.1.2 High-dimensional Transformations

NagE [70] introduces generic group theory to the design of KGE models and gives a generic recipe for their construction. QuatE [78] extends the KGE design to the Quaternion space which enables more compact interactions between entities and relations while introducing more degree of freedom. To model non-commutativeness in relation composition more effectively, both RotatE3D [19] and DensE [40] leverage quaternion rotations but in different forms. ROTH [9] adopts the hyperbolic curvature to capture the hierarchical structure in KGs. On the other hand, it is questioned in [57] whether the introduction of hyperbolic geometry in KGE is necessary.

2.2 Classification-based Models

Another family of models is built by classifying an unseen triple into "valid" (or positive) and "invalid" (or negative) two classes and then using the soft decision to measure the likelihood of the triple.

2.2.1 Neural Models

A multilayer perceptron (MLP) network [18] is used to measure the likelihood of unseen triples for link prediction. The neural tensor network (NTN) [47] adopts a bilinear tensor neural layer to model interactions between entities and relations of triples. ConvE [15] stacks head entities and relations, reshapes them to 2D arrays, and uses the convolutional neural network (CNN) to extract the information from them. The resulting feature map interacts with tail entities through dot products. R-GCN [46] uses the graph convolutional network (GCN) with relation-specific weights to obtain entity representations, which are subsequently fed to DistMult [68] for link prediction. Despite its potential of handling the inductive setting, its performance is not on par with the embedding based approach.

2.2.2 Advanced Neural Networks

KG-BERT [72] uses the pretrained language model, BERT [16], to obtain the entity representation from textual descriptions (rather than from KG links). However, its inference time is much longer compared to embedding-based models. SimKGC [58] improves transformer-based classification methods by constructing contrastive pairs. It uses BERT to estimate the semantic similarity and treats triples of higher similarity score as positive sample pairs, and vice versa. However, its performance is sensitive to the language model quality, and its required computational resource is high. Assuming entities textual descriptions, KEPLER [60] uses pretrained transformers to extract textual embeddings as initialization for entity embeddings. It then uses TransE embedding as a decoder to perform link prediction. Other more advanced KGE can be potentially applied in KEPLER to improve the KG completion performance.

2.2.3 Lightweight Classification Model

KGBoost [63] proposes a novel negative sampling scheme, and uses the XG-Boost [11] classifier for link prediction. Inspired by the Discriminant Feature Learning (DFT) [71, 34] that extracts most discriminative features from trained embeddings, GreenKGC [62] is a lightweight and modularized classification method that trains a binary classifier to classify unseen triples.

2.3 Advanced Relation Modeling

Special techniques have been developed to model complex relations. For example, to model relations such as 1-to-N, N-to-1, and N-to-N effectively, TransH [64] projects the embedded entity space into relation-specific hyper-planes. TransR [38] learns a relation-specific projection that maps entity vectors to a certain relation space. TransD [29] derives dynamic mapping based on relation and entity projection vectors. TranSparse [30] enforces the relation projection matrix to be sparse. Recently, many KGE models including X+AT [69], SFBR [37], and STaR [35] apply translation and scaling operations to both distance-based and semantic-matching-based [59] models to improve the performance gain. The inclusion of translation is proven to be effective in improving KGEs in the Quaternion space such as DualE [7], BiQUE [25]. ReflectE [77] models each relation as a normal vector of a hyper-plane that reflects entity vectors. It can be used to model symmetric and inverse relations well. So far, the cascade of various affine operations is a natural yet unexplored idea to pursue.

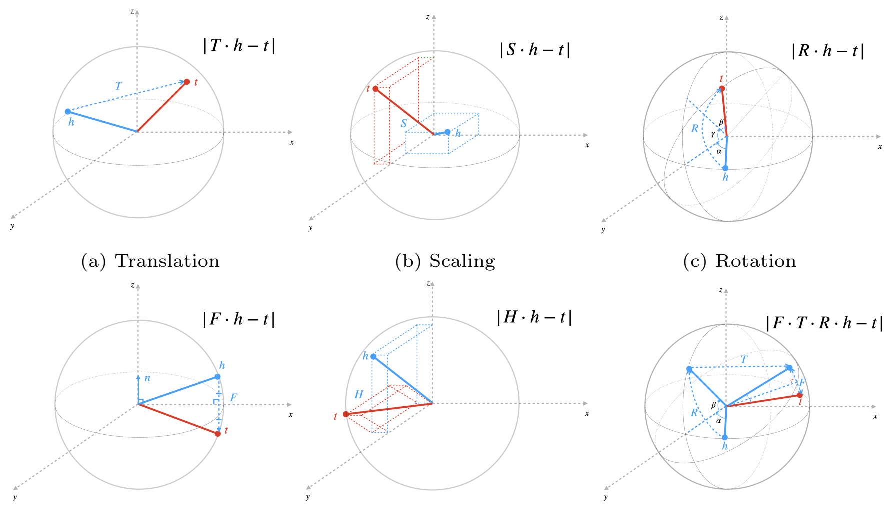
2.4 Model Ensembles

Although ensemble learning is a prevailing strategy in machine learning, it remains under-explored for knowledge graph completion. Link prediction evaluation is essentially a ranking problem. It is desired to optimize an ensemble decision so that valid triples get ranked higher than invalid ones among all candidates. Rank aggregation is a classical problem in information retrieval. Both supervised methods [6, 11] and unsupervised methods [33, 14] have been studied. Since the ground truth ranking in KG's link prediction is not available (except the top-1 triple), the unsupervised setting is more relevant. Yet, the use of rank aggregation to boost link prediction performance has received limited attention. Several examples are given below. KEnS [12] performs ensemble inference to combine predictions from multiple language-based KGEs for multilingual knowledge graph completion. AutoSF [79] develops an algorithm to search for the best scoring functions from multiple semantic matching models. The ensemble of multiple identical low-dimensional KGE models is adopted in [66] to boost the link prediction performance. Recently, DuEL [31] treats link prediction as a classification problem and aggregates binary decisions from several different classifiers using unsupervised techniques.

3 Proposed Method

3.1 CompoundE3D

In this work, we use 3D affine transformations, including Translation, Scaling, Rotation, Reflection, and Shear as illustrated in Figure 1, to model different relations in KGs. This large set of transformation operators offer immense flexibility in the KGE design against different characteristics of KG datasets. Below, we formally define each of the 3D affine operators in homogeneous coordinates.



(a) Translation (b) Scaling (c) Rotation

(d) Reflection (e) Shear (f) Compound

Figure 1: Composing different geometric operations in the 3D subspace.

3.1.1 Translation

Component $\mathbf{T} \in \mathbf{SE}(3)$, illustrated by Figure 1a, is defined as

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & v_y \\ 0 & 0 & 1 & v_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

3.1.2 Scaling

Component $\mathbf{S} \in \mathbf{Aff}(3)$, illustrated by Figure 1b, is defined as

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2}$$

3.1.3 Rotation

Component $\mathbf{R} \in \mathbf{SO}(3)$, illustrated by Figure 1c, is defined as

$$\mathbf{R} = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma) = \begin{bmatrix} a & b & c & 0 \\ d & e & f & 0 \\ g & h & i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

where

$$\begin{aligned} a &= \cos(\alpha)\cos(\beta), \\ b &= \cos(\alpha)\sin(\beta)\sin(\gamma) - \sin(\alpha)\cos(\gamma), \\ c &= \cos(\alpha)\sin(\beta)\cos(\gamma) + \sin(\alpha)\sin(\gamma), \\ d &= \sin(\alpha)\cos(\beta), \\ e &= \sin(\alpha)\sin(\beta)\sin(\gamma) + \cos(\alpha)\cos(\gamma), \\ f &= \sin(\alpha)\sin(\beta)\cos(\gamma) - \cos(\alpha)\sin(\gamma), \\ g &= -\sin(\beta), \\ h &= \cos(\beta)\sin(\gamma), \\ i &= \cos(\beta)\cos(\gamma). \end{aligned} \tag{4}$$

This general 3D rotation operator is the result of compounding yaw, pitch, and roll rotations. They are, respectively, defined as

- Yaw rotation component:

$$\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

- Pitch rotation component:

$$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos(\beta) & 0 & -\sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

- Roll rotation component:

$$\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) & 0 \\ 0 & \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{7}$$

3.1.4 Reflection

Component $\mathbf{F} \in \mathbf{SO}(3)$, illustrated by Figure 1d, is defined as

$$\mathbf{F} = \begin{bmatrix} 1-2n_x^2 & -2n_xn_y & -2n_xn_z & 0 \\ -2n_xn_y & 1-2n_y^2 & -2n_yn_z & 0 \\ -2n_xn_z & -2n_yn_z & 1-2n_z^2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{8}$$

The above expression is derive from the Householder reflection, $\mathbf{F} = \mathbf{I} - \mathbf{2nn^T}$. In the 3D space, $\mathbf{n}$ is a 3-D unit vector that is perpendicular to the reflecting hyper-plane, $\mathbf{n} = [n_x, n_y, n_z]$.

3.1.5 Shear

Component $\mathbf{H} \in \mathbf{Aff}(3)$, illustrated by Figure 1e, is defined as

$$\mathbf{H} = \mathbf{H}_{yz}\mathbf{H}_{xz}\mathbf{H}_{xy} = \begin{bmatrix} 1 & \mathrm{Sh}_x^y & \mathrm{Sh}_x^z & 0 \\ \mathrm{Sh}_y^x & 1 & \mathrm{Sh}_y^z & 0 \\ \mathrm{Sh}_z^x & \mathrm{Sh}_z^y & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{9}$$

The shear operator is the result of compounding 3 operators: $\mathbf{H}_{yz}$, $\mathbf{H}_{xz}$, and $\mathbf{H}_{xy}$ They are mathematically defined as

$$\mathbf{H}_{yz} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \mathrm{Sh}_y^x & 1 & 0 & 0 \\ \mathrm{Sh}_z^x & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{10}$$

$$\mathbf{H}_{xz} = \begin{bmatrix} 1 & \mathrm{Sh}_x^y & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & \mathrm{Sh}_z^y & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{11}$$

$$\mathbf{H}_{xy} = \begin{bmatrix} 1 & 0 & \mathrm{Sh}_x^z & 0 \\ 0 & 1 & \mathrm{Sh}_y^z & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{12}$$

Matrix $\mathbf{H}_{yz}$ has a physical meaning - the shear transformation that shifts the y- and z- components by a factor of the x component. Similar physical interpretations are applied to $\mathbf{H}_{xz}$ and $\mathbf{H}_{xy}$.

The above transformations can be cascaded to yield a compound operator; e.g.,

$$\mathbf{O} = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{R} \cdot \mathbf{F} \cdot \mathbf{H}, \tag{13}$$

In the actual implementation, we use the operator's representation in regular Cartesian coordinate instead of the homogeneous coordinate. Furthermore, a high-dimensional relation operator can be represented as a block diagonal matrix in the form of

$$\mathbf{M_r} = \mathbf{diag}(\mathbf{O_{r,1}}, \mathbf{O_{r,2}}, \ldots, \mathbf{O_{r,n}}), \tag{14}$$

where $\mathbf{O_{r,i}}$ is the compound operator at the i-th stage.

We can define the following three scoring functions for CompoundE3D:

- CompoundE3D-Head

$$f_r^{(h)}(h,t) = \|\mathbf{M_r} \cdot \mathbf{h} - \mathbf{t}\|, \tag{15}$$

- CompoundE3D-Tail

$$f_r^{(t)}(h,t) = \|\mathbf{h} - \hat{\mathbf{M}}_\mathbf{r} \cdot \mathbf{t}\|, \tag{16}$$

- CompoundE3D-Complete

$$f_r^{(h,t)}(h,t) = \|\mathbf{M_r} \cdot \mathbf{h} - \hat{\mathbf{M}}_\mathbf{r} \cdot \mathbf{t}\|, \tag{17}$$

where $\mathbf{h}$ and $\mathbf{t}$ denote head and tail entity embeddings, and $\mathbf{M_r}$ and $\hat{\mathbf{M}}_\mathbf{r}$ denote the relation-specific operators that operate on head and tail entities, respectively.

Generally speaking, we have five different affine operations available to use, i.e. translation, scaling, rotation, reflection, and shear. Each operator can be applied to 1) head entity, 2) tail entity, or 3) both head and tail. Hence, we have in total 15 different ways of applying operators at each stage. All these possible choices are called CompoundE3D variants. For a given KG dataset, there is a huge search space in finding the optimal CompoundE3D variant. It is essential to develop a simple yet effective mechanism to find a variant that gives the best performance under a certain complexity constraint.

3.2 Beam Search for Best CompoundE3D Variant

Since each affine operator can be viewed as a component in designing the new KGE, different order of combination of operators can lead to large design space of factorial complexity. There is a need to find the highly performant embedding design quickly without resorting to exhaustive search. In this subsection, we present a beam search algorithm to locate the close to optimal CompoundE3D variant. For the i-th stage, the set of all operator pairs that can be applied at a certain step is

$$\begin{aligned}\mathbf{P} \in \{&(\mathbf{T},\mathbf{I}),(\mathbf{S},\mathbf{I}),(\mathbf{R},\mathbf{I}),(\mathbf{F},\mathbf{I}),(\mathbf{H},\mathbf{I}),\\ &(\mathbf{I},\hat{\mathbf{T}}),(\mathbf{I},\hat{\mathbf{S}}),(\mathbf{I},\hat{\mathbf{R}}),(\mathbf{I},\hat{\mathbf{F}}),(\mathbf{I},\hat{\mathbf{H}}),\\ &(\mathbf{T},\hat{\mathbf{T}}),(\mathbf{S},\hat{\mathbf{S}}),(\mathbf{R},\hat{\mathbf{R}}),(\mathbf{F},\hat{\mathbf{F}}),(\mathbf{H},\hat{\mathbf{H}})\},\end{aligned} \tag{18}$$

where $\mathbf{I}$ is the identity operator. First, we apply all operator pairs in $\mathbf{P}$ and calculate scoring functions for all intermediate variants. Each variant is optimized with l iterations using the training set and its performance is evaluated on the validation dataset. Then, we choose the top-k best-performing variants as starting points for further exploration in the next step. The same process is repeated until one of terminating conditions is triggered. Afterward, we proceed to the $(i+1)$-th stage. The whole search is completed after the final stage is reached. The total number of stages is a user selected hyper-parameter.

The beam search process in building more complex KGE models from simpler ones is described in Algorithm 1. Additional comments are given below.

- We initialize the algorithm by setting up a loop to iterate over the set, $\mathbf{P}$, of all possible operator combinations to train and evaluate them and find the top-k variants as starting points.

- In the next loop, we have two stopping criteria to terminate the beam search: 1) # operators $> \lambda$, meaning that we stop the search when the number of operators exceeds the upper bound λ; 2) $\frac{\Delta \text{MRR}}{\Delta \text{Param}} < \gamma$, meaning that the ratio of increase in MRR versus the increase in free parameters fall below the threshold γ, and it is no longer worthwhile to increase the model complexity for the marginal gain in model performance.

- $\mathbf{P} \times \mathbf{W}$ denotes the Cartesian product between the operator pairs set $\mathbf{P}$ and top-k variants set $\mathbf{W}$ from the last step while $\dot{f}_r^{i-1}(h,t) \lhd (\mathbf{M}^i, \hat{\mathbf{M}}^i)$ denotes applying the operator pair $(\mathbf{M}^i, \hat{\mathbf{M}}^i)$ to previous optimal scoring function $\dot{f}_r^{i-1}(h,t)$.

- For example, if $\dot{f}_r^{i-1}(h,t) = \|\mathbf{R}\cdot\mathbf{h}-\mathbf{t}\|$ and $(\mathbf{M}^i, \hat{\mathbf{M}}^i) = (\mathbf{S}, \hat{\mathbf{S}})$, then $\tilde{f}_r^{i}(h,t) = \|\mathbf{S}\cdot\mathbf{R}\cdot\mathbf{h}-\hat{\mathbf{S}}\cdot\mathbf{t}\|$.

- After the loop terminates due to any terminating condition is triggered, we select the top-1 performing variant from the explored variants set, $\mathbf{W}$, as the best choice.

Algorithm 1 Beam Search for Best CompoundE3D Variant

```
initialize $i \leftarrow 1, \mathbf{U} \leftarrow \{\}$
for $(\mathbf{M}^i, \hat{\mathbf{M}}^i) \in \mathbf{P}$ do
    $\tilde{f}_r^i(h,t) \leftarrow \|\mathbf{M}^i \cdot \mathbf{h} - \hat{\mathbf{M}}^i \cdot \mathbf{t}\|$;
    train $\tilde{f}_r(h,t)$ for $l$ iterations;
    MRR $\leftarrow$ evaluate $\tilde{f}_r^i(h,t)$ with valid set;
    $\mathbf{U}$.insert({MRR, $\tilde{f}_r^i(h,t)$});
end for
$\mathbf{W} \leftarrow$ top-$k$ variants from $\mathbf{U}$
$i \leftarrow i+1$
$\Delta$MRR $\leftarrow \gamma$, $\Delta$Param $\leftarrow 1$
while # operators $\leq \lambda$ and max $\frac{\Delta \text{MRR}}{\Delta \text{Param}} \geq \gamma$ do
    initialize $\mathbf{V} \leftarrow \{\}$
    for $\{(\mathbf{M}^i, \hat{\mathbf{M}}^i), \dot{f}_r^{i-1}(h,t)\} \in \mathbf{P} \times \mathbf{W}$ do
        $\tilde{f}_r^i(h,t) \leftarrow \dot{f}_r^{i-1}(h,t) \lhd (\mathbf{M}^i, \hat{\mathbf{M}}^i)$;
        train $\tilde{f}_r^i(h,t)$ for $l$ iterations;
        evaluate $\tilde{f}_r^i(h,t)$ with valid set;
        $\Delta$MRR $\leftarrow \tilde{f}_r^i(h,t)$ MRR$-\dot{f}_r^{i-1}(h,t)$ MRR;
        $\Delta$Param $\leftarrow \tilde{f}_r^i(h,t)$ Param$-\dot{f}_r^{i-1}(h,t)$ Param;
        $\mathbf{V}$.insert(MRR, $\Delta$MRR, $\Delta$Param, $\tilde{f}_r^i(h,t)$);
    end for
    $\mathbf{W} \leftarrow$ top-$k$ variants from $\mathbf{V}$;
end while
$f_r^*(h,t) \leftarrow$ best variant from $\mathbf{W}$;
```

3.3 Model Ensembles

Figure 2 illustrates two main KGE ensembles methods that we experiment with CompoundE3D, namely rank fusion and weight distance sum.

3.3.1 Weighted-Distances-Sum (WDS) Strategy

We choose the top-k performing CompoundE3D variants and conduct a weighted average of their predicted scores. The following three weighting schemes are considered.

- **Uniform Weights.** This scheme takes an equal weight of selected k variants as

$$\hat{f}_r(h,t) = \frac{1}{k}\sum_{i=1}^{k} f_r^i(h,t), \tag{19}$$

  where $f_r^i(h,t)$ is the scoring function for the i-th variant.

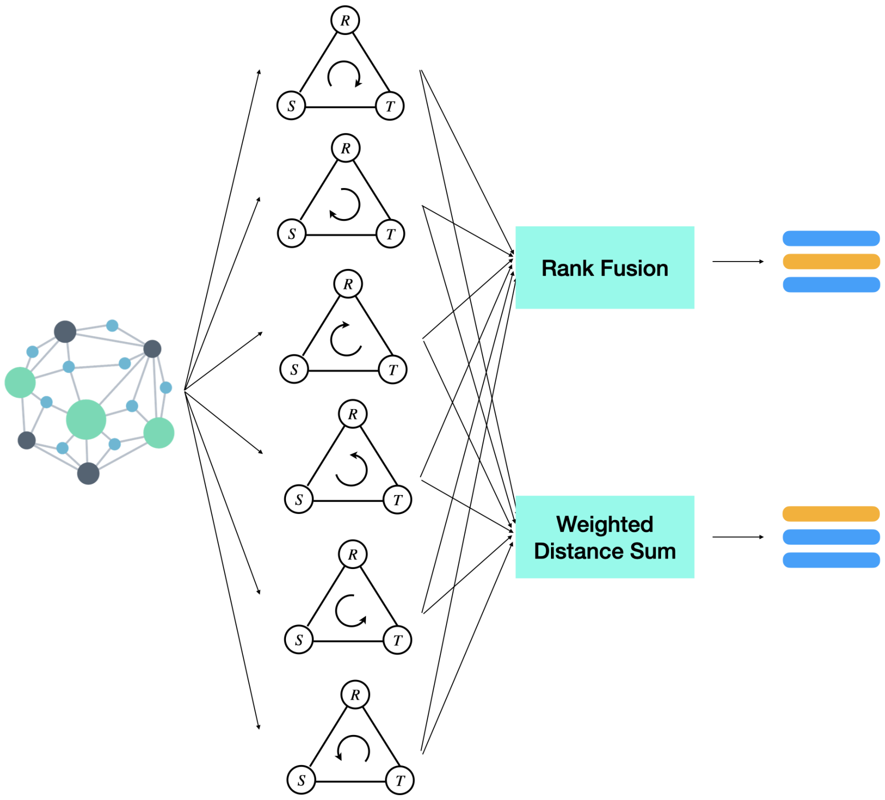


Figure 2: The ensemble of multiple CompoundE3D variants.

- **Geometric Weights.** This scheme sorts the variants based on their MRR performance on the validation dataset in a descending order and assign weight λ^k, $0 < \lambda < 1$, to the k-th variant. That is,

$$\hat{f}_r(h,t) = \frac{1}{\sum_{i=1}^{k} \lambda^k} \sum_{i=1}^{k} \lambda^k f_r^i(h,t). \tag{20}$$

  Since $\lambda^k > \lambda^{k+1}$, we assign a higher weight to a better performer in computing the aggregated distance.

- **Learnable Weights.** This scheme trains a set of learnable weights, $w_i > 0$, based on the training dataset to minimize the following weighted score:

$$\hat{f}_r(h,t) = \frac{1}{\sum_{i=1}^{k} w_i} \sum_{i=1}^{k} w_i f_r^i(h,t). \tag{21}$$

The learnable weights are implemented as parameters in the optimization process under the same learning rate and the optimizer in finding the best variants.

For each relation, we compare the three weight schemes and choose the one that offers the best performance.

3.3.2 Rank Fusion Strategy

Link prediction is a list ranking problem. Rank fusion can be exploited to boost the performance. A few simple rank fusion methods can be applied to score-based KGE methods. For example, we can take the maximum, minimum, median, sum, and L2 distance of candidates' ranks. They are denoted by CombMAX, CombMIN, CombMEDIAN, CombSUM, and Euclidean in Table 1, respectively. Three advanced rank fusion methods are also considered and included in the table. They are: Borda Count [53], Reciprocal Rank Fusion (RRF) [14], and RBC (Rank Biased Centroid) [3]. Borda Count awards points to candidates based on their positions in an individual preference list, where the top candidate gets the most points and the last candidate gets the least points. RRF aggregates the reciprocal rank to discount the importance of lower-ranked candidates. The factor k in the table mitigates the impact of high rankings by outliers. RBC discounts the weights of lower-ranked candidates using a geometric distribution. The mathematical formulas of all rank fusion functions are given in the second column of Table 1, where R_i is the rank of the i-th base model (or variant), $1 \leq i \leq n$, $e \in E$ represents an entity in the entity set, and k and ϕ are hyper-parameters.

3.4 Optimization

By following RotatE's negative sampling loss and the self-adversarial training strategy, we choose the following loss function of CompoundE3D

$$\begin{aligned} L_{\text{KGE}} = & -\log \sigma(\zeta_1 - f_r(h,t)) \\ & -\sum_{i=1}^{n} p(h'_i, r, t'_i) \log \sigma(f_r(h'_i, t'_i) - \zeta_1), \end{aligned} \tag{22}$$

where σ is the sigmoid function, ζ_1 is a preset margin hyper-parameter, (h'_i, r, t'_i) is the i-th negative triple, and $p(h'_i, r, t'_i)$ is the probability of drawing negative triple (h'_i, r, t'_i). Given a positive triple, (h_i, r, t_i), the negative sampling distribution is

$$p(h'_j, r, t'_j | \{(h_i, r, t_i)\}) = \frac{\exp \alpha_1 f_r(h'_j, t'_j)}{\sum_i \exp \alpha_1 f_r(h'_i, t'_i)}, \tag{23}$$

where α_1 is the temperature in the softmax function.

Table 1: A list of rank fusion functions under consideration.

Name	Function
CombMAX	$\max\{R_1(e), \cdots, R_n(e)\}$
CombMIN	$\min\{R_1(e), \cdots, R_n(e)\}$
CombMEDIAN	$\text{median}\{R_1(e), \cdots, R_n(e)\}$
CombSUM	$\sum_{i=1}^{n} R_i(e)$
Euclidean	$\sqrt{R_1(e)^2 + \cdots + R_n(e)^2}$
Borda Count	$\sum_{i=1}^{n} \frac{\|E\| - R_i(e) + 1}{\|E\|}$
RRF [14]	$\sum_{i=1}^{n} \frac{1}{k + R_i(e)}$
RBC [3]	$\sum_{i=1}^{n} (1-\phi)\phi^{R_i(e)-1}$

4 Experiments

4.1 Experimental Setup

4.1.1 Datasets

We evaluate the link prediction performance of CompoundE3D and compare it with several benchmarking methods on the following four KG datasets.

- **DB100K [17].** It is a subset of the DBpedia KG. The dataset contains information related to music content such as genre, band, and musical artisits. It is a relatively dense KG since each entity appears in at least 20 different relations.
- **YAGO3-10 [41].** It is a subset of YAGO3, which describes citizenship, gender, and profession of people. YGGO3-10 contains entities associated with at least 10 different relations.
- **WN18RR [5, 15].** It is a subset of the WordNet lexical database. The inverse relation is removed from WN18RR to avoid test leakage.
- **Ogbl-Wikikg2 [28].** It is extracted from Wikipedia. It contains 2.5M entities and is the largest one among the four selected datasets.

The statistics of the four KG datasets are given in Table 2.

Table 2: Statistics of four link prediction datasets.

Dataset	#Entities	#Relations	#Training	#Validation	#Test	Ave. Degree
DB100K	99,604	470	597,572	50,000	50,000	12
ogbl-wikikg2	2,500,604	535	16,109,182	429,456	598,543	12.2
YAGO3-10	123,182	37	1,079,040	5,000	5,000	9.6
WN18RR	40,943	11	86,835	3,034	3,134	2.19

4.1.2 Evaluation Protocol

The commonly used evaluation protocol for the link prediction task is explained below. For every triple (h, r, t) in the test set, we corrupt either the head entity h or tail entity t to generate test examples $(?, r, t)$ and $(h, r, ?)$. Then, for every head candidate that forms triple $(\hat{h}, r, t)$ and tail candidate that forms triple $(h, r, \hat{t})$, we compute distance-based scoring functions $f_r(\hat{h}, t)$ and $f_r(h, \hat{t})$, respectively. The lower score value indicates that the generated triple is more likely to be true. Then, we sort scores of all candidate triples in ascending order and locate the rank of the ground truth triple. Furthermore, we evaluate the link prediction performance under the filtered rank setting [5] that gives salience to unseen triple predictions since embedding models tend to give observed triples better ranks. We adopt the Hits@k and the mean reciprocal rank (MRR) as evaluation metrics to compare the quality of KGE models.

4.1.3 Hyper-parameter Search

We perform an extensive search on six hyper-parameters of CompoundE3D with respect to different KG datasets. They are: 1) the dimension of the embedding space (**Dim**), 2) the learning rate ($\boldsymbol{lr}$), 3) the batch size (B), 4) the negative sample size (N), 5) the margin hyper-parameter ($\boldsymbol{\zeta}$), and 6) the sampling temperature ($\boldsymbol{\alpha}$). Their search values are listed in Table 3.

In the search process, we first compute scoring functions with a certain hyper-parameter setting that allows a few variants to have decent performance, where the number of training iterations for each variant is set to $l = 30000$. After locating the optimal variant, we finetune hyper-parameters under the optimal variant. The optimal configurations are shown in Table 4. The Adam optimizer [32] is employed for all parameter tuning. For ensemble experiments, we adopt the same optimal configuration for each base variant model.

4.1.4 Other Implementation Details

We run experiments and perform hyper-parameter tuning on a variety of GPUs, including Nvidia P100 (16G), V100 (32G), A100 (40G) and A40 (48G), depending on the GPU memory requirement of a job. Typically, we

Table 3: The search space of six hyper-parameters.

Dataset	**DB100K**	**ogbl-wikikg2**	**YAGO3-10**	**WN18RR**
Dim	$\{150, 300, 450, 600\}$	$\{90, 150, 180, 240, 300\}$	$\{450, 600, 750, 900\}$	$\{180, 240, 360, 480, 600\}$
$\boldsymbol{lr}$	$\{2, 3, 4, 5, 6, 7, 8, 9\} \times 10^{-5}$	$\{0.0005, 0.001, 0.005, 0.01\}$	$\{3, 4, 5, 6, 7\} \times 10^{-4}$	$\{4, 5, 6, 7, 8\} \times 10^{-4}$
$\boldsymbol{B}$	$\{256, 512, 1024, 2048\}$	$\{2048, 4096, 8192\}$	$\{512, 1024, 2048, 4096\}$	$\{512, 1024, 2048, 4096\}$
$\boldsymbol{N}$	$\{256, 512, 1024, 2048\}$	$\{125, 250, 500\}$	$\{256, 512, 1024, 2048\}$	$\{256, 512, 1024, 2048\}$
$\boldsymbol{\zeta}$	$\{4, 5, 6, 7, 8, 9, 10, 11, 12, 13\}$	$\{5, 6, 7, 8, 9\}$	$\{11, 12, 13, 13.1, 13.3, 13.5\}$	$\{5, 6, 7, 8, 9\}$
$\boldsymbol{\alpha}$	$\{0.5, 0.7, 0.9, 1.0, 1.2\}$	$\{0.5, 1.0\}$	$\{0.8, 0.9, 1.0, 1.1, 1.2\}$	$\{0.5, 0.7, 0.9, 1.0, 1.2\}$

Table 4: Optimal configurations for link prediction tasks, where B and N denote the batch size and the negative sample size, respectively.

Dataset	CompoundE3D Variant	#Dim	lr	B	N	ζ	α
DB100K	$\|\mathbf{S}\cdot\mathbf{h}-\hat{\mathbf{T}}\cdot\hat{\mathbf{R}}\cdot\hat{\mathbf{S}}\cdot\mathbf{t}\|$	600	0.00005	1024	512	9	1
ogbl-wikikg2	$\|\mathbf{T}\cdot\mathbf{h}-\hat{\mathbf{H}}\cdot\mathbf{t}\|$	300	0.001	8192	125	8	1
YAGO3-10	$\|\mathbf{T}\cdot\mathbf{S}\cdot\mathbf{R}\cdot\mathbf{h}-\mathbf{t}\|$	600	0.0005	1024	1024	13.3	1.1
WN18RR	$\|\mathbf{R}\cdot\mathbf{S}\cdot\mathbf{T}\cdot\mathbf{h}-\mathbf{t}\|$	480	0.00005	512	256	6	1

request 8 CPU cores with less than 70G RAM for each job. Results of each optimal configuration in Table 4 can be reproduced on one single V100 for all datasets. For the WN18RR dataset, we adopt the rotation implementation from Rotate3D [19].

4.2 *Experimental Results*

4.2.1 *Performance Evalution*

We compare the link prediction performance of a few benchmarking KGE methods with that of CompoundE3D using the optimal configuration given in Table 4. The performance benchmarking results for DB100K, ogbl-wikikg2, and YAGO3-10 datasets are shown, respectively, in Table 5, Table 6, and

Table 5: Comparison of the link prediction performance under the filtered rank setting for DB100k.

Datasets	**DB100K**			
Model	**MRR**	**H@1**	**H@3**	**H@10**
TransE [5]	0.111	0.016	0.164	0.27
DistMult [68]	0.233	0.115	0.301	0.448
HolE [42]	0.26	0.182	0.309	0.411
ComplEx [52]	0.242	0.126	0.312	0.44
Analogy [39]	0.252	0.142	0.323	0.427
RUGE [26]	0.246	0.129	0.325	0.433
ComplEx-NNE+AER [17]	0.306	0.244	0.334	0.418
SEEK [67]	0.338	0.268	0.37	0.467
AcrE (Parallel) [44]	0.413	0.314	0.472	0.588
PairRE [10]	0.412	0.309	0.472	0.600
TransSHER [36]	0.431	0.345	0.476	0.589
CompoundE [20]	0.405	0.306	0.461	0.588
CompoundE3D	0.450	0.373	0.488	0.594
CompoundE3D RRF	<u>0.457</u>	<u>0.376</u>	<u>0.497</u>	<u>0.607</u>
CompoundE3D WDS	**0.462**	**0.378**	**0.506**	**0.616**

Table 6: Comparison of the link prediction performance under the filtered rank setting for ogbl-wikikg2.

Datasets	**ogbl-wikikg2**		
Metrics	**Dim**	**Valid MRR**	**Test MRR**
AutoSF+NodePiece	100	0.5806	0.5703
ComplEx-N3-RP	100	0.6701	0.6481
TransE [5]	500	0.4272	0.4256
DistMult [68]	500	0.3506	0.3729
ComplEx [52]	250	0.3759	0.4027
RotatE [51]	250	0.4353	0.4353
Rotate3D [19]	100	0.5685	0.5568
PairRE [10]	200	0.5423	0.5208
TripleRE [73]	200	0.6045	0.5794
CompoundE [22]	100	0.6704	0.6515
CompoundE3D	90	0.6994	0.6826
	180	<u>0.7146</u>	<u>0.6962</u>
	300	**0.7175**	**0.7006**

Table 7: Comparison of the link prediction performance under the filtered rank setting for YAGO3-10.

Datasets	**YAGO3-10**			
Metrics	**MRR**	**Hit@1**	**Hit@3**	**Hit@10**
DistMult [68]	0.34	0.24	0.38	0.54
ComplEx [52]	0.36	0.26	0.4	0.55
DihEdral [65]	0.472	0.381	0.523	0.643
ConvE [15]	0.44	0.35	0.49	0.62
RotatE [51]	0.495	0.402	0.55	0.67
InteractE [54]	0.541	0.462	-	0.687
HAKE [80]	<u>0.545</u>	0.462	0.596	0.694
DensE [40]	0.541	**0.465**	0.585	0.678
Rot-Pro [48]	0.542	0.443	0.596	0.699
CompoundE [20]	0.477	0.376	0.538	0.664
CompoundE3D	0.542	0.450	0.602	0.701
CompoundE3D RRF	0.541	0.446	<u>0.607</u>	**0.707**
CompoundE3D WDS	**0.551**	<u>0.463</u>	**0.608**	<u>0.703</u>

Table 7. In particular, rows with bold model names are experimental results from this paper and results from other models are obtained from other papers. The best and the second-best results in each column are indicated by the boldface font and with an underline respectively. CompoundE3D has significant

performance improvement over CompoundE and other recent models. We see a clear advantage of CompoundE3D by including more affine operators and extending affine transformations from 2D to 3D in the new framework,

To verify the effectiveness of model ensembles, we examine two different ensemble strategies for DB100K and YAGO3-10 datasets.

- For the DB100K dataset, we select the best two performing variants. They are $\|\mathbf{S} \cdot \mathbf{h} - \hat{\mathbf{R}} \cdot \hat{\mathbf{S}} \cdot \mathbf{t}\|$ and $\|\mathbf{S} \cdot \mathbf{h} - \hat{\mathbf{T}} \cdot \hat{\mathbf{R}} \cdot \hat{\mathbf{S}} \cdot \mathbf{t}\|$
- For the YAGO3-10 dataset, we select the best three performing variants. They are $\|\mathbf{T} \cdot \mathbf{S} \cdot \mathbf{R} \cdot \mathbf{h} - \mathbf{t}\|$, $\|\mathbf{S} \cdot \mathbf{R} \cdot \mathbf{T} \cdot \mathbf{h} - \mathbf{t}\|$, and $\|\mathbf{S} \cdot \mathbf{T} \cdot \mathbf{R} \cdot \mathbf{h} - \mathbf{t}\|$.

4.2.2 Model Ensembles

As discussed in Section 3.3, we have two strategies to conduct model ensembles: weighted-distances-sum (WDS) and rank fusion. Among the three WDS strategies, the learnable weight strategy is the most effective one for DB100K while the uniform weight performs the best for YAGO3-10. We use CompoundE3D WDS to denote the best WDS scheme in Tables 5 and 7 and document the performance of other weighting strategies in Table 8. Among all eight rank fusion strategies, we observe that reciprocal rank fusion (RRF) is the most effective one for both DB100K and YAGO3-10. Thus, we use CompoundE3D RRF to denote the best rank fusion scheme in Tables 5 and 7, and document the performance of other rank fusion strategies in Table 9. Overall, the WDS strategy has a slight advantage over the best performing rank fusion strategy RRF. There are differences between WDS and RRF strategy. WDS aggregates the distance scores from each sub-models in the ensemble and then makes the ranking decision. In the RRF strategy, the individual model first ranks relevant entities based on the distance score, and then reranks the entities by using a rank aggregation function. The WDS strategy requires more memory to be optimized on a single GPU. If without model sharding, the GPU memory can be a bottleneck for including more models to further boosting the ensemble performance. For RRF, although each sub-model can be trained on a single GPU in parallel, the rank aggregation computation has a long runtime.

Table 8: Comparison of different weighted-distances-sum (WDS) strategies for DB100K and YAGO3-10.

Datasets	**DB100K**				**YAGO3-10**			
Strategies	**MRR**	**H@1**	**H@3**	**H@10**	**MRR**	**H@1**	**H@3**	**H@10**
Learnable Weights	**0.462**	**0.378**	**0.506**	**0.616**	0.545	0.451	0.586	0.696
Uniform Weights	0.460	0.376	0.503	0.614	**0.551**	**0.463**	**0.608**	**0.703**
Geometric Weights	0.446	0.348	0.503	0.618	0.531	0.439	0.580	0.691

Table 9: Performance comparison of different rank fusion methods for DB100K and YAGO3-10.

Datasets	**DB100K**				**YAGO3-10**			
Aggregation Function	**MRR**	**Hit@1**	**Hit@3**	**Hit@10**	**MRR**	**Hit@1**	**Hit@3**	**Hit@10**
CombMAX	0.455	0.375	0.496	0.603	0.536	0.440	0.600	0.701
CombMIN	0.452	0.369	0.496	0.606	0.527	0.427	0.597	0.702
CombMEDIAN	0.456	0.376	0.497	0.606	0.541	0.445	0.605	0.705
CombSUM	0.456	0.376	0.497	0.606	0.540	0.446	0.606	0.704
Euclidean	0.455	0.375	0.496	0.605	0.540	0.445	0.603	0.702
Borda	0.456	0.376	0.497	0.606	0.540	0.446	0.606	0.704
RRF	**0.457**	**0.376**	**0.497**	**0.607**	**0.541**	**0.446**	**0.607**	**0.707**
RBC	0.456	0.376	0.497	0.606	0.540	0.445	0.604	0.703

4.2.3 Effectiveness of Beam Search

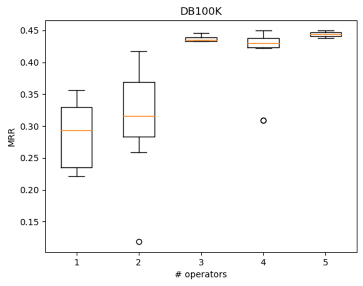
We conduct ablation studies on DB100K and YAGO3-10 datasets to shed light on the effects of different transformation operators on model performance. We begin with the variant of the simplest configuration and add additional operators at each stage. Good simple models that lead to optimal variants and their performance numbers are reported in Tables 10 and 11. Furthermore, we visualize the distribution of the MRR performance as more operators are added with respect to DB100K and YAGO-3 in Figures 3a and 3b, respectively. To interpret box plots, yellow bar represents the median, box represents the interquantile range, two end-bars denote the lower and upper whiskers, and lastly dots are outliers. They both show the effectiveness of the proposed beam search algorithm.

Table 10: Ablation study on CompoundE3D for DB100K.

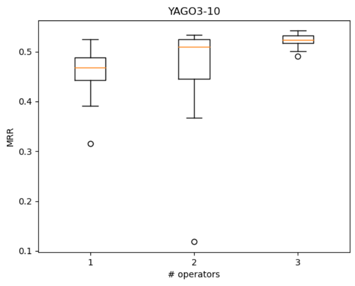
Datasets	**DB100K**			
Variant	**MRR**	**Hit@1**	**Hit@3**	**Hit@10**
$\|\|\mathbf{S}\cdot\mathbf{h}-\hat{\mathbf{S}}\cdot\mathbf{t}\|\|$	0.417	0.323	0.471	0.590
$\|\|\mathbf{S}\cdot\mathbf{h}-\hat{\mathbf{R}}\cdot\hat{\mathbf{S}}\cdot\mathbf{t}\|\|$	0.447	0.364	0.492	0.600
$\|\|\mathbf{S}\cdot\mathbf{h}-\hat{\mathbf{T}}\cdot\hat{\mathbf{R}}\cdot\hat{\mathbf{S}}\cdot\mathbf{t}\|\|$	0.450	0.373	0.488	0.594

Table 11: Ablation study on CompoundE3D for YAGO3-10.

Datasets	**YAGO3-10**			
Metrics	**MRR**	**Hit@1**	**Hit@3**	**Hit@10**
$\|\|\mathbf{R}\cdot\mathbf{h}-\mathbf{t}\|\|$	0.496	0.402	0.547	0.676
$\|\|\mathbf{S}\cdot\mathbf{R}\cdot\mathbf{h}-\mathbf{t}\|\|$	0.501	0.404	0.554	0.680
$\|\|\mathbf{T}\cdot\mathbf{S}\cdot\mathbf{R}\cdot\mathbf{h}-\mathbf{t}\|\|$	0.542	0.450	0.602	0.701



(a) DB100K



(b) YAGO3-10

Figure 3: The distribution of the MRR performance versus the operator number of various model variants for DB100K and YAGO3-10 dataset.

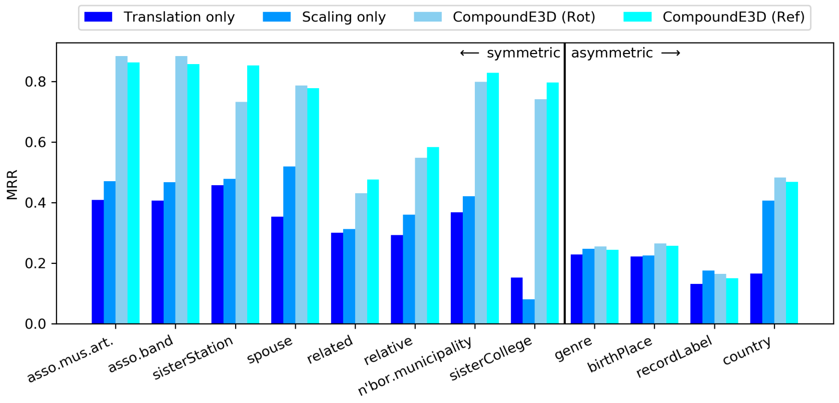


Figure 4: Effects of rotation and reflection operators on symmetric relations.

4.2.4 Modeling of Symmetric Relations

Rotation and reflection are isometric operations. As stated in [51, 77], their 2D versions can handle symmetric relations well in some cases. It is our conjecture that the same property holds for their corresponding 3D operators. To check it, we perform ablation studies and evaluate the base scoring functions of those with only translation and scaling versus those with rotation and reflection as well. The MRR performance numbers of different model variants for symmetric and asymmetric relations in DB100K are compared in Figure 4. In this figure, we choose the most frequently observed relation types for meaningful comparison. As expected, rotation and reflection operators indeed bring more significant performance improvement on symmetric relations than asymmetric relations. This supports our conjecture that rotation and reflection operators are intrinsically advantageous for the modeling of symmetric relations.

4.2.5 Modeling of Multiplicity

Multiplicity is the scenario where multiple relations co-exist between two entities; namely, triples $(h, r_1, t), \ldots, (h, r_n, t)$ hold simultaneously. Generally, it is challenging to model multiplicity in traditional KGE models due to their limited power in relational modeling. In contrast, CompoundE3D is capable of modeling multiplicity relation patterns well since it can use multiple distinct sets of transformations that map from the head to the tail. We present two examples to illustrate CompoundE3D's capability in modeling multiplicity relations in Figure 5. They are taken from the actual link prediction examples in DB100K. There are three different relations held for a fixed (head, tail) pair. The top three tail predictions for each relation in the two examples are shown in the figure. We see that CompoundE3D can handle multiplicity well due to its rich set of variants.

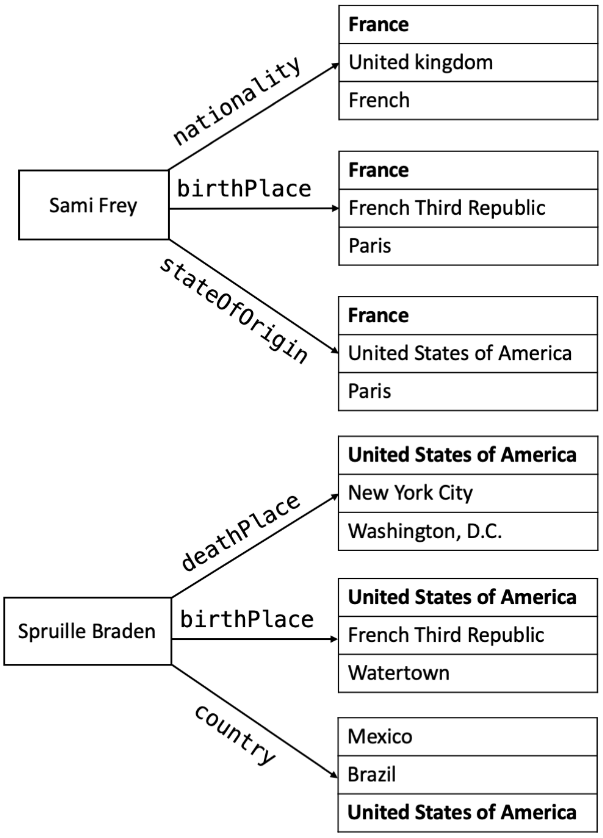


Figure 5: Illustration of CompoundE3D's capability in multiplicity modeling.

4.2.6 Modeling of Hierarchical Relations

We would like to investigate CompoundE3D's capability in modeling hierarchical relations. WN18RR offers a representative dataset containing hierarchical relations. Two metrics can be used to measure the hierarchical behavior of relations [9]: 1) the Krackhardt score denoted by Khs_{G_r}, and 2) the curvature estimate denoted by ξ_{G_r}. If relation r has a high Khs_{G_r} score and a low ξ_{G_r} score, then it has a stronger hierarchical behavior, and vice versa. We compare the filtered MRR performance of different baseline models, such as TransE, RotatE, CompoundE (2D version), and CompoundE3D in Table 12. In the same table, we also list the Khs_{G_r} and ξ_{G_r} values for each relation to see whether it has a stronger hierarchical behavior. We see from the table that CompoundE and CompoundE3D have better performance than TransE and RotatE in almost all relations. Furthermore, CompoundE3D outperforms CompoundE in all hierarchical relations except "member meronym". This result indicates that CompoundE3D can model hierarchical relations more effectively than CompoundE by including more diverse 3D transformations.

Table 12: Comparison of filtered MRR performance on each relation type of WN18RR.

Relations	**Types**	Khs_{G_r}	ξ_{G_r}	**TransE**	**RotatE**	**CompoundE**	**CompoundE3D**
similar to	1-to-1	0.07	-1.00	0.294	**1.000**	**1.000**	**1.000**
verb group	1-to-1	0.07	-0.50	0.363	0.961	**0.974**	0.898
member meronym	1-to-N	1.00	-0.50	0.179	**0.259**	0.254	0.246
has part	1-to-N	1.00	-1.43	0.117	0.200	0.200	**0.202**
member of domain usage	1-to-N	1.00	-0.74	0.113	0.297	0.309	**0.378**
member of domain region	1-to-N	1.00	-0.78	0.114	0.217	0.401	**0.413**
hypernym	N-to-1	1.00	-2.64	0.059	0.156	0.179	**0.182**
instance hypernym	N-to-1	1.00	-0.82	0.289	0.322	0.351	**0.356**
synset domain topic of	N-to-1	0.99	-0.69	0.149	0.339	0.382	**0.396**
also see	N-to-N	0.36	-2.09	0.227	0.625	**0.629**	0.622
derivationally related form	N-to-N	0.07	-3.84	0.440	0.957	0.956	**0.959**

4.2.7 Model Efficiency

It is important to investigate the relationship between the model performance and the model dimension. The model dimension reflects memory and computational complexities. To illustrate the advantage of CompoundE3D over prior models across a wide range of embedding dimensions, we plot the MRR performance of link prediction on the Wikikg2 dataset in Figure 6, where the dimension values are set to 12, 24, 48, 102, 150, 198, 252, and 300. We see from the figure that CompoundE3D consistently outperforms all benchmarking models in all dimensions. Furthermore, we analyze the complexity of different KGE models in terms of the number of free parameters. Table 13 compares the number of free parameters of different KGE models for the ogbl-wikikg2 dataset. In addition, we compared the wall clock time for completing the

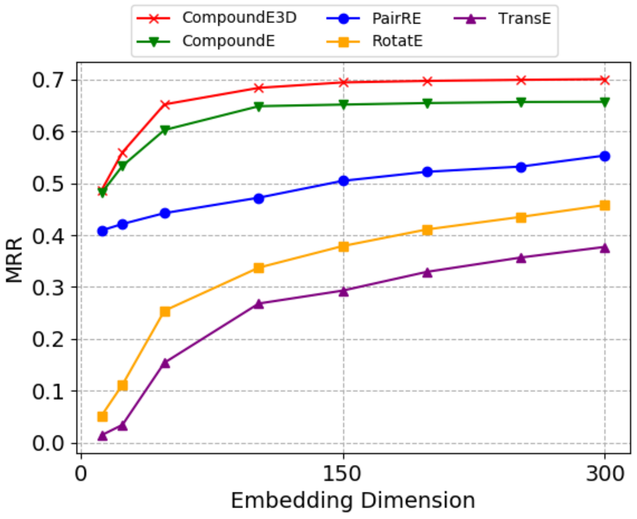


Figure 6: Comparing different model's MRR performance metric across different dimensions.

Table 13: Complexity comparison of KGE models on ogbl-wikikg2 under a similar testing MRR.

Models	**No. of Parameters**	**Wall Clock Time**
TransE [5]	1,251M	-
DistMult [68]	1,251M	-
ComplEx [52]	1,251M	-
ComplEx-RP [13]	250.1M	-
RotatE [51]	500M	-
RotatE3D [19]	750.4M	-
PairRE [10]	500M	-
CompoundE [20]	250.1M	8 hrs 48 mins
CompoundE3D	**225.2M**	7 hrs 4 mins

training and inference experiments under the same hardware setting for CompoundE and CompoundE3D using the optimal configuration. CompoundE3D reduces the total runtime by 19.7%. This runtime reduction is largely due to the simplier formulation of the CompoundE3D scoring function.

We refer to the ogbl-wikikg2 leaderboard when reporting the number of free parameters used by baseline models. The reported number of parameters for CompoundE3D is when the embedding dimension is set to 90. As shown in Figure 6 and Table 13, CompoundE3D offers the best performance among all benchmarking models while having the smallest number of free parameters.

5 Conclusion and Future Work

A novel and effective KGE model based on composite affine transformations in the 3D space, named CompoundE3D, was proposed in this work. A beam search procedure was devised to build a desired KGE from the simplest configuration to more complicated ones. The ensemble of the top-k model variants was also explored to further boost link prediction performance. Extensive experimental results were provided to demonstrate the superior performance of CompoundE3D. We conducted ablation studies to assess the effect of each operator and performed case studies to shed light on the modeling power of CompoundE3D for several relation types such as multiplicity, symmetric relations, and hierarchical relations.

As to future research directions, it will be interesting to explore the effectiveness of CompoundE3D in other important KG problems such as entity typing [24, 61] and entity alignment [21]. Besides, research on performance boosting in low-dimensional embedding space is valuable in practical real-world applications and worth further investigation.

Large Language Models (LLMs) have demonstrated impressive capability in different NLP applications and become the state-of-the-art model in many tasks. We believe the LLM can still benefit from incorporating knowledge graph embedding methods such as retrieval augmented generation (RAG). To achieve that, it is essential to combine KG embedding methods with text representation learning approaches. We are confident that our work can contribute to and be integrated into future research developments in this area. Furthermore, today's LLMs are based on generative pre-trained transformers (GPTs). They are challenged by their hallucination, reliability, huge computational complexity, and lack of incremental learning capabilities. To tackle these deficiencies, an alternative approach could be the decomposition of a generic LLM to several smaller domain-specific mid-size language models (MLM) that have a "multi-modal interface" to handle textual or visual input/output and a "knowledge core" implemented by knowledge graphs (KGs) for knowledge representation, data mining, and incremental learning. Such a modular design could improve the interpretability, reliability, computational complexity, and incremental capability of next-generation MLMs with justifiable and logical reasoning.

6 Acknowledgments

The authors acknowledge the Center for Advanced Research Computing (CARC) at the University of Southern California for providing computing resources that have contributed to the research results reported within this publication. URL: https://carc.usc.edu.

References

[1] A. Ait-Mlouk and L. Jiang, "KBot: A Knowledge graph based chatBot for natural language understanding over linked data", *IEEE Access*, 8, 2020, 149220–30.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a web of open data", in *The Semantic Web*, 2007, 722–35.

[3] P. Bailey, A. Moffat, F. Scholer, and P. Thomas, "Retrieval consistency in the presence of query variations", in *Proc. 40th Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retrieval (SIGIR '17)*, 2017, 395–404.

[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and 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, 1247–50.

[5] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data", *Proceedings of the 26th International Conference on Neural Information Processing Systems*, 2013, 2787–95.

[6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: from pairwise approach to listwise approach", in *Proc. 24th Int. Conf. Mach. Learn. (ICML 2007)*, 2007, 129–36.

[7] Z. Cao, Q. Xu, Z. Yang, X. Cao, and Q. Huang, "Dual quaternion knowledge graph embeddings", in *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021, 6894–902.

[8] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, and T. M. Mitchell, "Toward an architecture for never-ending language learning", in *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010, 1306–13.

[9] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré, "Low-Dimensional Hyperbolic Knowledge Graph Embeddings", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 6901–14.

[10] L. Chao, J. He, T. Wang, and W. Chu, "PairRE: Knowledge Graph Embeddings via Paired Relation Vectors", in *Proc. 59th Annu. Meet. Assoc. Comput. Linguist. (ACL 2021)*, 2021, 4360–9.

[11] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system", in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, 785–94.

[12] X. Chen, M. Chen, C. Fan, A. Uppunda, Y. Sun, and C. Zaniolo, "Multilingual Knowledge Graph Completion via Ensemble Knowledge Transfer", in *Proc. Conf. Empirical Meth. Natural Language Process. Find. (EMNLP'20 Findings)*, 2020, 3227–38.

[13] Y. Chen, P. Minervini, S. Riedel, and P. Stenetorp, "Relation Prediction as an Auxiliary Training Objective for Improving Multi-Relational Graph Representations", in *Proc. 3rd Conf. Autom. Knowl. Base Constr. (AKBC2021)*, 2021.

[14] G. V. Cormack, C. L. Clarke, and S. Buettcher, "Reciprocal rank fusion outperforms condorcet and individual rank learning methods", in *Proc. 32nd Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retrieval (SIGIR '09)*, 2009, 758–9.

[15] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings", in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, 2018, 1811–8.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", *arXiv preprint arXiv:1810.04805*, 2018.

[17] B. Ding, Q. Wang, B. Wang, and L. Guo, "Improving knowledge graph embedding using simple constraints", *arXiv preprint arXiv:1805. 02408*, 2018.

[18] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and 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, 601–10.

[19] C. Gao, C. Sun, L. Shan, L. Lin, and M. Wang, "Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding", in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, 385–94.

[20] X. Ge, Y. C. Wang, B. Wang, and C.-C. J. Kuo, "Compounding Geometric Operations for Knowledge Graph Completion", in *Proc. 61st Annu. Meet. Assoc. Comput. Linguist. (ACL 2023)*, 2023, 6947–65.

[21] X. Ge, Y. C. Wang, B. Wang, C.-C. J. Kuo, *et al.*, "TypeEA: Type-Associated Embedding for Knowledge Graph Entity Alignment", *APSIPA Transactions on Signal and Information Processing*, 12(1),

[22] X. Ge, Y.-C. Wang, B. Wang, and C.-C. J. Kuo, "CompoundE: Knowledge graph embedding with translation, rotation and scaling compound operations", *arXiv preprint arXiv:2207.05324*, 2022.

[23] X. Ge, Y.-C. Wang, B. Wang, and C.-C. J. Kuo, "Knowledge Graph Embedding: An Overview", *arXiv preprint arXiv:2309.12501*, 2023.

[24] X. Ge, Y.-C. Wang, B. Wang, and C. J. Kuo, "CORE: A knowledge graph entity type prediction method via complex space regression and embedding", *Pattern Recognition Letters*, 157, 2022, 97–103.

[25] J. Guo and S. Kok, "BiQUE: Biquaternionic Embeddings of Knowledge Graphs", *arXiv preprint arXiv:2109.14401*, 2021.

[26] S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo, "Knowledge graph embedding with iterative guidance from soft rules", in *Proc. 32nd AAAI Conf. Artif. Intell.* Vol. 32, No. 1, 2018.

[27] H. He, A. Balakrishnan, M. Eric, and P. Liang, "Learning Symmetric Collaborative Dialogue Agents with Dynamic Knowledge Graph Embeddings", in *Proc. 55th Annu. Meet. Assoc. Comput. Linguist. (ACL 2017)*, 2017, 1766–76.

[28] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open Graph Benchmark: Datasets for Machine Learning on Graphs", *arXiv preprint arXiv: 2005.00687*, 2020.

[29] G. Ji, S. He, L. Xu, K. Liu, and 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 (volume 1: Long papers)*, 2015, 687–96.

[30] G. Ji, K. Liu, S. He, and J. Zhao, "Knowledge graph completion with adaptive sparse transfer matrix", in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, 985–91.

[31] U. Joshi and J. Urbani, "Ensemble-Based Fact Classification with Knowledge Graph Embeddings", in *Proc. 19th Extended Semantic Web Conf. (ESWC'22)*, Springer, 2022, 147–64.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980*, 2014.

[33] A. Klementiev, D. Roth, and K. Small, "Unsupervised rank aggregation with distance-based models", in *Proc. 25th Int. Conf. Mach. Learn. (ICML 2008)*, 2008, 472–9.

[34] C.-C. J. Kuo and A. M. Madni, "Green learning: Introduction, examples and outlook", *J. Vis. Commun. Image. Represent.*, 2022, 103685.

[35] J. Li and Y. Yang, "STaR: Knowledge Graph Embedding by Scaling, Translation and Rotation", in *International Conference on AI and Mobile Services*, 2022, 31–45.

[36] Y. Li, W. Fan, C. Liu, C. Lin, and J. Qian, "TranSHER: Translating Knowledge Graph Embedding with Hyper-Ellipsoidal Restriction", in *Proc. 60th Annu. Meet. Assoc. Comput. Linguist. (ACL 2022)*, December 2022, 8517–28.

[37] Z. Liang, J. Yang, H. Liu, and K. Huang, "A semantic filter based on relations for knowledge graph completion", in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, 7920–9.

[38] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion", in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, 2181–7.

[39] H. Liu, Y. Wu, and Y. Yang, "Analogical inference for multi-relational embeddings", in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, 2168–78.

[40] H. Lu, H. Hu, and X. Lin, "DensE: An enhanced non-commutative representation for knowledge graph embedding with adaptive semantic hierarchy", *Neurocomputing*, 476, 2022, 115–25.

[41] F. Mahdisoltani, J. Biega, and F. Suchanek, "Yago3: A knowledge base from multilingual wikipedias", in *Proc. 7th Biennial Conf. Innov. Data Syst. Research*, CIDR Conference, 2014.

[42] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs", in *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30, No. 1, 2016.

[43] W. K. Pratt, *Introduction to digital image processing*, CRC press, 2013.

[44] F. Ren, J. Li, H. Zhang, S. Liu, B. Li, R. Ming, and Y. Bai, "Knowledge Graph Embedding with Atrous Convolution and Residual Learning", in *Proc. 58th Annu. Meet. Assoc. Comput. Linguist. (ACL 2020)*, 2020, 1532–43.

[45] S. Sang, Z. Yang, L. Wang, X. Liu, H. Lin, and J. Wang, "SemaTyP: a knowledge graph based literature mining method for drug discovery", *BMC Bioinform.*, 19, 2018, 1–11.

[46] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks", in *Proceedings of the European Semantic Web Conference*, 2018, 593–607.

[47] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion", *Proceedings of the 26th International Conference on Neural Information Processing Systems*, 26, 2013.

[48] T. Song, J. Luo, and L. Huang, "Rot-pro: Modeling transitivity by projection in knowledge graph embedding", *Adv. Neural Info. Process. Syst. 34 (NeurIPS 2021)*, 34, 2021, 24695–706.

[49] R. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An open multilingual graph of general knowledge", in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017, 4444–51.

[50] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A Core of Semantic Knowledge", in *Proceedings of the 16th International Conference on World Wide Web*, 2007, 697–706.

[51] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, "RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space", in *7th International Conference on Learning Representations (ICLR 2019)*, 2019, 1–18.

[52] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction", in *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, 2016, 2071–80.

[53] M. Van Erp and L. Schomaker, "Variants of the borda count method for combining ranked classifier hypotheses", in *7th International Workshop on frontiers in handwriting recognition*, International Unipen Foundation, 2000, 443–52.

[54] S. Vashishth, S. Sanyal, V. Nitin, N. Agrawal, and P. Talukdar, "Interacte: Improving convolution-based knowledge graph embeddings by increasing feature interactions", in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, Vol. 34, No. 03, 2020, 3009–16.

[55] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledge base", *Communications of the ACM*, 57(10), 2014, 78–85.

[56] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems", in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage. (CIKM'18)*, 2018, 417–26.

[57] K. Wang, Y. Liu, D. Lin, and M. Sheng, "Hyperbolic Geometry is Not Necessary: Lightweight Euclidean-Based Models for Low-Dimensional Knowledge Graph Embeddings", in *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021, 464–74.

[58] L. Wang, W. Zhao, Z. Wei, and J. Liu, "SimKGC: Simple Contrastive Knowledge Graph Completion with Pre-trained Language Models", in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, 4281–94.

[59] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications", *IEEE Transactions on Knowledge and Data Engineering*, 29(12), 2017, 2724–43.

[60] X. Wang, T. Gao, Z. Zhu, Z. Zhang, Z. Liu, J. Li, and J. Tang, "KEPLER: A unified model for knowledge embedding and pre-trained language representation", *Transactions of the Association for Computational Linguistics*, 9, 2021, 176–94.

[61] Y.-C. Wang, X. Ge, B. Wang, and C.-C. J. Kuo, "AsyncET: Asynchronous Learning for Knowledge Graph Entity Typing with Auxiliary Relations", *arXiv preprint arXiv:2308.16055*, 2023.

[62] Y.-C. Wang, X. Ge, B. Wang, and C.-C. J. Kuo, "Greenkgc: A lightweight knowledge graph completion method", *arXiv preprint arXiv:2208.09137*, 2022.

[63] Y.-C. Wang, X. Ge, B. Wang, and C.-C. J. Kuo, “KGBoost: A classification-based knowledge base completion method with negative sampling”, *Pattern Recognition Letters*, 157, 2022, 104–11.

[64] Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph embedding by translating on hyperplanes”, in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014, 1112–9.

[65] C. Xu and R. Li, “Relation Embedding with Dihedral Group in Knowledge Graph”, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, 263–72.

[66] C. Xu, M. Nayyeri, S. Vahdati, and J. Lehmann, “Multiple run ensemble learning with low-dimensional knowledge graph embeddings”, in *Proc. 2021 Int. Joint Conf. Neural Netw. (IJCNN)*, IEEE, 2021, 1–8.

[67] W. Xu, S. Zheng, L. He, B. Shao, J. Yin, and T.-Y. Liu, “SEEK: Segmented Embedding of Knowledge Graphs”, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 3888–97.

[68] B. Yang, S. W.-t. Yih, X. He, J. Gao, and L. Deng, “Embedding Entities and Relations for Learning and Inference in Knowledge Bases”, in *Proceedings of the International Conference on Learning Representations (ICLR) 2015*, 2015.

[69] J. Yang, Y. Shi, X. Tong, R. Wang, T. Chen, and X. Ying, “Improving knowledge graph embedding using affine transformations of entities corresponding to each relation”, in *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021, 508–17.

[70] T. Yang, L. Sha, and P. Hong, “NagE: Non-Abelian group embedding for knowledge graphs”, in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, 1735–42.

[71] Y. Yang, W. Wang, H. Fu, C.-C. J. Kuo, *et al.*, “On supervised feature selection from high dimensional feature spaces”, *APSIPA Transactions on Signal and Information Processing*, 11(1), 2022.

[72] L. Yao, C. Mao, and Y. Luo, “KG-BERT: BERT for knowledge graph completion”, *arXiv preprint arXiv:1909.03193*, 2019.

[73] L. Yu, Z. Luo, H. Liu, D. Lin, H. Li, and Y. Deng, “Triplere: Knowledge graph embeddings via tripled relation vectors”, *arXiv preprint arXiv:2209.08271*, 2022.

[74] T. Yu, J. Li, Q. Yu, Y. Tian, X. Shun, L. Xu, L. Zhu, and H. Gao, “Knowledge graph for TCM health preservation: Design, construction, and applications”, *Artif. Intell. Med.*, 77, 2017, 48–52.

[75] X. Zeng, X. Tu, Y. Liu, X. Fu, and Y. Su, “Toward better drug discovery with knowledge graph”, *Curr. Opin. Struct. Biol.*, 72, 2022, 114–26.

[76] Q. Zhan and H. Yin, “A loan application fraud detection method based on knowledge graph and neural network”, in *Proc. 2nd Int. Conf. Innov. Artif. Intell.* 2018, 111–5.

[77] Q. Zhang, R. Wang, J. Yang, and L. Xue, “Knowledge graph embedding by reflection transformation”, *Knowledge-Based Systems*, 238, 2022, 107861.

[78] S. Zhang, Y. Tay, L. Yao, and Q. Liu, “Quaternion knowledge graph embeddings”, in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, 2735–45.

[79] Y. Zhang, Q. Yao, W. Dai, and L. Chen, “AutoSF: Searching scoring functions for knowledge graph embedding”, in *Proceedings of the 2020 IEEE 36th International Conference on Data Engineering (ICDE)*, IEEE, 2020, 433–44.

[80] Z. Zhang, J. Cai, Y. Zhang, and J. Wang, “Learning hierarchy-aware knowledge graph embeddings for link prediction”, in *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, No. 03, 2020, 3065–72.

[81] K. Zhou, W. X. Zhao, S. Bian, Y. Zhou, J.-R. Wen, and J. Yu, “Improving conversational recommender systems via knowledge graph based semantic fusion”, in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD ’20)*, 2020, 1006–14.

[82] X. Zhu, X. Ao, Z. Qin, Y. Chang, Y. Liu, Q. He, and J. Li, “Intelligent financial fraud detection practices in post-pandemic era”, *The Innov.*, 2(4), 2021, 100176.