applied sciences

MDPI

*Article*

# Dual Quaternion Embeddings for Link Prediction

**Liming Gao [1,†], Huiling Zhu [2,†], Hankz Hankui Zhuo [1,*] and Jin Xu [3]**

[1] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510000, China; gaolm3@mail2.sysu.edu.cn
[2] College of Information Science and Technology, Jinan University, Guangzhou 510000, China; zhuhl02@gmail.com
[3] Data Quality Team, WeChat, Tencent Inc., Guangzhou 510000, China; jinxxu@tencent.com
* Correspondence: zhuohank@mail.sysu.edu.cn
† These authors contributed equally to this work.

**Abstract:** The applications of knowledge graph have received much attention in the field of artificial intelligence. The quality of knowledge graphs is, however, often influenced by missing facts. To predict the missing facts, various solid transformation based models have been proposed by mapping knowledge graphs into low dimensional spaces. However, most of the existing transformation based approaches ignore that there are multiple relations between two entities, which is common in the real world. In order to address this challenge, we propose a novel approach called `DualQuatE` that maps entities and relations into a dual quaternion space. Specifically, entities are represented by pure quaternions and relations are modeled based on the combination of rotation and translation from head to tail entities. After that we utilize interactions of different translations and rotations to distinguish various relations between head and tail entities. Experimental results exhibit that the performance of `DualQuatE` is competitive compared to the existing state-of-the-art models.

**Keywords:** knowledge graph embedding; link prediction; artificial intelligence



**Citation:** Gao, L.; Zhu, H.; Zhuo, H.H.; Xu, J. Dual Quaternion Embeddings for Link Prediction. *Appl. Sci.* **2021**, *11*, 5572. https://doi.org/10.3390/app11125572

Academic Editor: Rafael Valencia-Garcia

Received: 13 May 2021
Accepted: 10 June 2021
Published: 16 June 2021



## 1. Introduction

Knowledge graphs, which represent knowledge from real world applications, contain abundant facts. In knowledge graphs, each fact is represented by a triple $(h, r, t)$ which indicates that the relation $r$ between the head entity $h$ and tail entity $t$. Knowledge graphs have been applied to various tasks such as explainable recommendation system [1], question answering [2] and prediction of future research collaborations [3].

Predicting missing facts (i.e., link prediction) is a fundamental task in knowledge graph research. Various models aiming at embedding entities and relations into low-dimension spaces have been proposed. For example, TransE [4] learned the embeddings of entities and relations by transforming head entity to tail entity according to the relation; RotatE [5] and QuatE [6] learned the embeddings of entities and relations by considering relations as rotations from head entities to tail entities. However, existing transformation based models fail to capture multiple relations between head and tail entities. For example, as shown in Figure 1, *David Lynch* is the director, the creator and an actor in the film *Mulholland Drive*, i.e., there are three relations: directed, created and actedIn between *David Lynch* and *Mulholland Drive*. These relations between head entity *David Lynch* and tail entity *Mulholland Drive* have no semantic connections with each other, which should be represented by spatially dispersed embeddings. Most existing transformation based models, however, assume that there is only one relation between each pair of head and tail entities. For instance, for each triple $(h, r, t)$, their corresponding embeddings are assumed to be satisfied with $h + r \approx t$ in TransE, which indicates, for $(h, r_1, t)$, $(h, r_2, t)$, and $(h, r_3, t)$, the embeddings of $r_1$, $r_2$, $r_3$ are similar, as shown in Figure 2c (i.e., $\mathbf{r}_1 \approx \mathbf{r}_2 \approx \mathbf{r}_3$). To overcome this challenge, we propose a novel approach that considers multiple relations between head and tail entities in knowledge graph.

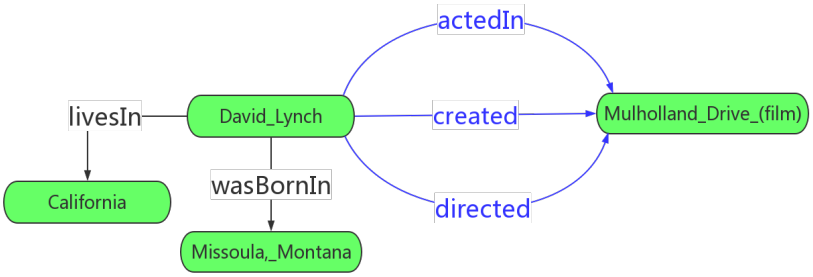
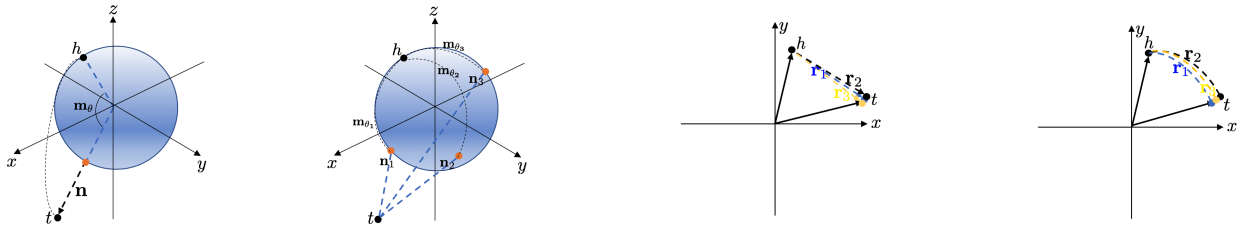



**Figure 1.** Visualization of partial knowledge graph in YAGO3-10. The blue relations indicate that there can be multiple relations between two entities.



(**a**) `DualQuatE` (**b**) relations (**c**) TransE (**d**) RotatE

**Figure 2.** Geometrical significance of `DualQuatE` and multiple relations between the entities. (**a**) denotes the interaction of rotation and translation from the head entity $h$ to the tail entity $t$. $\mathbf{m}_\theta$ represents the rotation of the relation, $\mathbf{n}$ denotes the translation of the relation. (**b**) shows how to express multiple relations between the head entity $h$ and the tail entity $t$ for `DualQuatE`. (**c**,**d**) demonstrate that TransE and RotatE fail to model multiple relations.

In this paper, we propose a model called `DualQuatE` which utilizes the various combinations of distinct rotations and translations to represent multiple relations between head and tail entities. Based on this, easily to think of RotatE combined with TransE in complex space and real space. However, it is hard to find a uniform mathematical expression to convey their combination. Therefore, we propose `DualQuatE` which embeds entities and relations into dual quaternion space to combine rotation and translation. The dual quaternion consists of real part and dual part. More concretely, we embed entities with pure quaternions vectors in three-dimensional space to represent entity embeddings. To distinguish various relations between head entity $h$ and tail entity $t$, we design a score function to utilize dual quaternion Hamilton product to model relations as interaction of rotation and translation. We utilize distinct interactions of rotations and translations to represent various relations between head and tail entities. Compared with RotatE and TransE in two-dimensional space, the dual quaternions space is eight-dimensional with six real degrees of freedom, three for translation and three for rotation; we can explore the interaction of rotation and translation with more free degrees in higher dimensions. Summarized in Table 1, our model has rich expression abilities of relations (i.e., relation patterns and multiple relations).

To conclude, the contributions of our proposed model are listed as follows:

- We introduce dual quaternions to knowledge graph embeddings.
- We propose a novel transformation based model DualQuatE to overcome the challenge of multiple relations between two entities.
- Our experiments denote that DualQuatE is effective compared to the existing state-of-the-art models.

**Table 1.** The ability of expressing relation patterns and multiple relations between head and tail entities.

| Model | Symmetry | Antisymmetry | Inversion | Composition | Multiple Relations |
|---|---|---|---|---|---|
| TransE | ✓ | ✓ | ✓ | ✓ | X |
| RotatE | ✓ | ✓ | ✓ | ✓ | X |
| HAKE [7] | ✓ | ✓ | ✓ | ✓ | X |
| DistMult [8] | ✓ | X | X | X | ✓ |
| ComplEx [9] | ✓ | ✓ | ✓ | X | ✓ |
| QuatE | ✓ | ✓ | ✓ | X | ✓ |
| `DualQuatE` | ✓ | ✓ | ✓ | ✓ | ✓ |

The rest paper is organized as follows. In Section 2, we introduce the related work. Section 3 presents prerequisite knowledge about dual quaternions. In Section 4, we describe our model. We present the results of experiments and make analysis and discussions in Section 5. In Section 6, we introduce the conclusion of this paper and future work.

## 2. Related Work

To gain high-quality knowledge graphs, approaches which utilize knowledge graph embedding to predict missing facts have been proposed recently. These methods fall into two broad categories in [10]: transformation based models and semantic matching models. Specifically, transformation based models transform head entity to tail entity by relations, while semantic matching models match entities and relations semantics in latent spaces. Compared to transformation based models, semantic matching models suffer from poor interpretability.

Transformation based models usually embed entities and relations into vector space and model the relation as a transformation from head entity embeddings to tail entity embeddings. One of the most representative is TransE which mapped entities and relations to the same space $\mathbb{R}^k$. For each triple $(h, r, t)$, entity embeddings $\mathbf{h}$, $\mathbf{t}$ and relation embedding $\mathbf{r}$ hold $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. Then a series of extensions based on TransE are presented to improve accuracy and interpretability. For instance, TransR [11] introduced relations-specific spaces. TransR modeled relations and entities into different spaces following the idea that TransE can only express 1-to-1 relations. RotatE mapped embeddings into complex space which focused on expressing relation patterns. HAKE [7] utilized the polar coordinate system to capture semantic hierarchies in the knowledge graphs.

Semantic matching models that match latent semantics of entities and relations can be divided into two categories: bilinear models and neural network based models. Bilinear models include DistMult [8], HolE [12], SimplE [9], ComplEx [13] and QuatE and DihEdral [14]. DistMult represented each entity as a vector and each relation as a diagonal matrix. HolE matched latent semantics of entities by circular correlation operation and then the compositional vector interacted with relations latter. ComplEx, mapping knowledge graph embedding into complex space, leveraged Hermitian product to capture latent semantics of entities and relations which could express antisymmetry relation pattern. QuatE, extending knowledge graph embedding from complex space to quaternion space, modeled each relation as rotation in four-dimensional space with more degree of freedom. Compared with ComplEx, QuatE could express the main relation patterns except composition. For each entity, SimplE proposed two embeddings and each of them learned latent semantics dependently. DihEdral mapped relations into dihedral group to capture composition relations. Neural network based models including ConvE [15], R-GCNs [16] and InteractE [17] are proposed recently. ConvE, R-GCNs introduced convolutional network and graph convolutional networks to knowledge graph embedding respectively. Compared with ConvE, InteractE introduced the feature permutation, "checkered" feature reshaping and circular convolution to increase interaction.

Recently, some models introduced hyperbolic space to knowledge graph embeddings. MuRP [18] represented knowledge graph in Poincaré ball of hyperbolic space.

Chami et al. [19] attempted to capture hierarchical and logical patterns in hyperbolic space. Compared to hyperbolic space based models which focused on semantic hierarchies in knowledge graphs, `DualQuatE` try to overcome overcome the challenge of multiple relations between two entities and relation patterns.

Both `DualQuatE` and QuatE use quaternion to embed knowledge graphs. However, those are three different models. The main differences between `DualQuatE` and QuatE are as follows:

- `DualQuatE`, a transformation based model, measures score of triples by the distance between two entities. QuatE which is a semantic matching model measured the latent matching semantics of entities and relations.
- The purpose of the model is different. `DualQuatE` aims to address the challenge of having multiple relations between two entities. QuatE aims to utilize quaterion Hamilton product to encourage a more compact interaction between entities and relations.
- The geometric meaning is different. QuatE embeds entities and relations with quaterions to model relations as rotations. Our model firstly attempts to represent entities with pure quaternions and models relations as interaction of translation and rotation.

## 3. Preliminaries

In this part, we introduce several concepts used in this paper.

- **Quaternion**: Quaternion [20], is a number system that extends complex numbers to four-dimensional numbers. Generally, a quaternion is a number of the form $\mathrm{q} = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$, where $a, b, c, d$ are real numbers and $\mathbf{i}, \mathbf{j}, \mathbf{k}$ satisfy that $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$.
- **Quaternion conjugate**: The definition of conjugate to a quaternion q is $\mathrm{q}^* = \mathrm{a} - \mathrm{b}\mathbf{i} - \mathrm{c}\mathbf{j} - \mathrm{d}\mathbf{k}$.
- **Quaternion Multiplication:** Multiplication of two quaternions $\mathrm{p} = p_0 + p_1\mathbf{i} + p_2\mathbf{j} + p_3\mathbf{k}$ and $\mathrm{q} = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$ is defined by:

$$\begin{aligned} \mathrm{pq} = &(p_0q_0 - p_1q_1 - p_2q_2 - p_3q_3) \\ &+ (p_0q_1 + p_1q_0 + p_2q_3 - p_3q_2)\mathbf{i} \\ &+ (p_0q_2 - p_1q_3 + p_2q_0 + p_3q_1)\mathbf{j} \\ &+ (p_0q_3 + p_1q_2 - p_2q_1 + p_3q_0)\mathbf{k} \end{aligned} \tag{1}$$

- **Rotation with quaternions in three-dimensional space**: The point $\mathrm{v}'$ is rotated by the point v along the unit vector u (i.e., rotation axis), which can use quaternion multiplication to represent. We define v and $\mathrm{v}'$ as pure quaternion, i.e., quaternions with real part being zero, $\mathrm{m} = \cos\frac{\theta}{2} + \mathrm{u}\sin\frac{\theta}{2}$ is a unit quaternion, then

$$\mathrm{v}' = \mathrm{mvm}^* \tag{2}$$

- **Dual quaternion**: Dual quaterion [21] is an eight-dimensional real algebra to combine with quaternions. Formally, a dual quaternion $\delta$ can be represented by $\delta = \mathrm{p} + \epsilon\mathrm{q}$, where $\epsilon$ is a dual unit with $\epsilon^2 = 0$, both the real part p and the dual part q are quaternions. Therefore, a dual quaternion $\delta$ is of the form $\delta = p_0 + p_1\mathbf{i} + p_2\mathbf{j} + p_3\mathbf{k} + \epsilon(q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k})$.
- **Dual quaternion conjugate**: The conjugate of the dual quaternion $\delta = \mathrm{p} + \epsilon\mathrm{q}$ is defined as: $\delta^\diamond = \mathrm{p}^* - \epsilon\mathrm{q}^*$, which can be represented by an 8-tuple: $\delta^\diamond = (p_0, -p_1, -p_2, -p_3, -q_0, q_1, q_2, q_3)$.
- **Dual Quaternion Multiplication**: Dual Quaternion Hamilton product between $\delta_1 = \mathrm{p}_1 + \epsilon\mathrm{q}_1$ and $\delta_2 = \mathrm{p}_2 + \epsilon\mathrm{q}_2$ is defined as follows:

$$\delta_1 \otimes \delta_2 = \mathrm{p}_1\mathrm{p}_2 + \epsilon(\mathrm{p}_1\mathrm{q}_2 + \mathrm{q}_1\mathrm{p}_2) \tag{3}$$

- **Unit Dual Quaternion**: A dual quaternion $\delta = \mathrm{p} + \epsilon\mathrm{q}$ is a unit dual quaternion, if $\delta \otimes \delta^* = 1$, namely, $\delta$ satisfies the following conditions:

$$\begin{aligned} p_0^2 + p_1^2 + p_2^2 + p_3^2 = 1, \\ p_0 q_0 + p_1 q_1 + p_2 q_2 + p_3 q_3 = 0 \end{aligned} \tag{4}$$

  where $\delta^* = \mathrm{p}^* + \epsilon\mathrm{q}^*$. In order to simplify the calculation process, we use another effective form to represent unit dual quaternion which defines as follows:

$$\delta = \mathrm{m} + \frac{\epsilon}{2}\mathrm{mn} \tag{5}$$

  where $\mathrm{m} = \cos\frac{\theta}{2} + \mathrm{u}\sin\frac{\theta}{2}$ and n is a pure quaterion. We prove that $\delta$ is a unit dual quaternion:

$$\begin{aligned} \delta \otimes \delta^* &= \left(\mathrm{m} + \frac{\epsilon}{2}\mathrm{nm}\right) \otimes \left(\mathrm{m}^* + \frac{\epsilon}{2}(\mathrm{nm})^*\right) \\ &= \mathrm{mm}^* + \frac{\epsilon}{2}(\mathrm{mm}^*\mathrm{n}^* + \mathrm{nmm}^*) \\ &= 1 + \frac{\epsilon}{2}(\mathrm{n}^* + \mathrm{n}) \\ &= 1 + \frac{\epsilon}{2}(-\mathrm{n} + \mathrm{n}) \\ &= 1 \end{aligned} \tag{6}$$

  We can easily verify $\mathrm{mm}^* = 1$, as shown below:

$$\begin{aligned} \mathrm{mm}^* &= (\cos\frac{\theta}{2} + \mathrm{u}\sin\frac{\theta}{2})(\cos\frac{\theta}{2} - \mathrm{u}\sin\frac{\theta}{2}) \\ &= (\cos\frac{\theta}{2} + u_i\sin\frac{\theta}{2}\mathbf{i} + u_j\sin\frac{\theta}{2}\mathbf{j} + u_k\sin\frac{\theta}{2}\mathbf{k})(\cos\frac{\theta}{2} - u_i\sin\frac{\theta}{2}\mathbf{i} - u_j\sin\frac{\theta}{2}\mathbf{j} - u_k\sin\frac{\theta}{2}\mathbf{k}) \\ &= 1 \end{aligned} \tag{7}$$

- **Combination of Rotation and Translation**: We define a point in the three-dimensional space as a pure quaterion v and let n be the translation. The point v under the rotation $\theta$ followed by the translation n becomes the point $\mathrm{v}'$. It is straightforward to utilize unit dual quaternion multiplication to represent the transformation from v to $\mathrm{v}'$, as shown below:

$$\begin{aligned} \delta \otimes (1 + \epsilon\mathrm{v}) \otimes \delta^{\diamond} &= \left(\mathrm{m} + \frac{\epsilon}{2}\mathrm{nm}\right) \otimes (1 + \epsilon\mathrm{v}) \otimes \left(\mathrm{m}^* - \frac{\epsilon}{2}(\mathrm{nm})^*\right) \\ &= \left(\mathrm{m} + \frac{\epsilon}{2}\mathrm{nm} + \epsilon\mathrm{mv}\right) \otimes \left(\mathrm{m}^* - \frac{\epsilon}{2}\mathrm{m}^*\mathrm{n}^*\right) \\ &= \mathrm{mm}^* + \epsilon\left(\frac{1}{2}(\mathrm{nmm}^* - \mathrm{mm}^*\mathrm{n}^*) + \mathrm{mvm}^*\right) \\ &= 1 + \epsilon(\mathrm{mvm}^* + \mathrm{n}) \\ &= (1 + \epsilon\mathrm{v}') \end{aligned} \tag{8}$$

## 4. Our `DualQuatE` Model

In this section, we introduce our model `DualQuatE` which maps entities and relations to dual quaternion space, and two variations of `DualQuatE`, namely `DualQuatE-1` and `DualQuatE-2`.

We denote a knowledge graph by $\mathcal{G}$, a set of entities by $\mathcal{E}$ and a set of relations by $\mathcal{R}$. A knowledge graph $\mathcal{G}$ is composed of a set of facts, each of which can be represented by $(h, r, t)$, where $h \in \mathcal{E}$ is a head entity, $t \in \mathcal{E}$ is a tail entity, and $r \in \mathcal{R}$ is a relation between $h$ and $t$. We denote a set of facts that are **true** by $\Omega^+$, and a set of facts that are **false** by $\Omega^-$. Given a knowledge graph $\mathcal{G}$, we aim to predict missing facts (i.e., link prediction) in $\mathcal{G}$.

### *4.1. Multiple Relations between the Entities*

To address the challenge of having multiple relations between head and tail entities, we embed knowledge graph into dual quaternion space. $\mathbf{h}, \mathbf{r}, \mathbf{t}$ denote vector of entity embeddings and relation embeddings, each element of entity embeddings $h_i$ or $t_i$ is pure quaternion and every dimension of relation embeddings $r_i$ is unit dual quaternion. We expect to model relation embeddings $\mathbf{r}$ as **interaction** of **rotation** and **translation** from head entity embeddings $\mathbf{h}$ to tail entity embeddings $\mathbf{t}$ as shown in Figure 2a. Specifically, each true triple $(h, r, t)$ satisfies:

$$\mathbf{r} \otimes (1 + \epsilon\mathbf{h}) \otimes \mathbf{r}^{\diamond} = (1 + \epsilon\mathbf{t}) \tag{9}$$

where each dimension of $\mathbf{r}$ is a unit dual quaterinon satisfying Formula (4). We define a quaternion $\mathbf{m} = \cos\frac{\theta}{2} + \mathbf{u}\sin\frac{\theta}{2}$ to represent a rotation about pure unit quaternion $\mathbf{u}$ through $\theta$ and a pure quaternion $\mathbf{n} = n_1\mathbf{i} + n_2\mathbf{j} + n_3\mathbf{k}$. Furthermore, we define a unit dual quaternion by:

$$\mathbf{r} = \mathbf{m} + \frac{\epsilon}{2}\mathbf{nm} \tag{10}$$

With Formula (10), we can deduce the transformation of `DualQuatE` in Formula (9):

$$\begin{aligned} \mathbf{r} \otimes (1 + \epsilon\mathbf{h}) \otimes \mathbf{r}^{\diamond} &= 1 + \epsilon(\mathbf{mhm}^{*} + \mathbf{n}) \\ &= (1 + \epsilon\mathbf{t}) \end{aligned} \tag{11}$$

where the geometric meaning of $\mathbf{mhm}^{*}$ is shown in Formula (2). As shown above, `DualQuatE` transforms head entity $h$ to tail entity $t$ by relation $r$ which combines **rotation** (i.e., $\mathbf{m}$) and **translation** (i.e., $\mathbf{n}$). Unlike previous models learned similar representations of relations $r_1, r_2, r_3$ shown in Figure 2c,d, our model learns combinations of different translations and rotations to represent various relations between head and tail entities.

We define score function by:

$$f_r(h, t) = -||\mathbf{r} \otimes (1 + \epsilon\mathbf{h}) \otimes \mathbf{r}^{\diamond} - (1 + \epsilon\mathbf{t})|| \tag{12}$$

where $||\cdot||$ represent $L_2$ norm of a vector. With the score function we want head entity to be as close to tail entity as possible after the transformation of the relation.

### *4.2. Loss Function*

We employ self-adversarial negative sampling [5] method to generate corrupt samples. We define the probability distribution of negative samples by:

$$p(h'_j, r, t'_j | \{(h'_i, r_i, t'_i)\}) = \frac{\exp \alpha f_r(h'_j, t'_j)}{\sum_i \exp \alpha f_r(h'_i, t'_i)} \tag{13}$$

where $\alpha$ is sampling temperature. Combining with self-adversarial negative sampling, we define loss function by:

$$\begin{aligned} L = &- \log \sigma(\gamma + f_r(h, t)) \\ &- \sum_{i=1}^{n} p(h_i', r, t_i') \log \sigma(-f_r(h'_i, t'_i) - \gamma) \end{aligned} \tag{14}$$

where $\gamma$ is fixed margin. We define our algorithm as shown in Algorithm 1.

**Algorithm 1** `DualQuatE`.

**Input**: Entity embeddings $\mathcal{E}$ and relation embeddings $\mathcal{R}$. hyperparameters including margin $\gamma$, martrix dim $k$, negative sample size $n$.

$\mathbf{h}, \mathbf{t} \leftarrow$ uniform $(-\frac{\gamma+2.0}{k}, \frac{\gamma+2.0}{k})$ for each $\mathbf{h}, \mathbf{t} \in \mathcal{E}$
$\mathbf{r} \leftarrow$ uniform $(-\frac{\gamma+2.0}{k}, \frac{\gamma+2.0}{k})$ for each $\mathbf{r} \in \mathcal{R}$
**repeat**
  $T_{pos} \leftarrow$ uniform random sampling $(h, r, t)$
  $(h', r, t') \leftarrow$ generate $n$ negative samples for $(h, r, t)$
  $T = T_{pos} \cup \{(h', r, t')\}$
  compute each $(h', r, t')$ weight: $p(h'_j, r, t'_j | \{(h'_i, r_i, t'_i)\})$
  update relation embeddings $\mathbf{r}$ and entity embeddings $\mathbf{h}, \mathbf{t}$:

$$\mathbf{h}, \mathbf{r}, \mathbf{t} = \mathbf{h}, \mathbf{r}, \mathbf{t} - \nabla_{\theta_r}[-\log \sigma(\gamma + f_r(h, t)) - \sum_{i=1}^{n} p(h_i{}', r, t_i{}') \log \sigma(-f_r(h'_i, t'_i) - \gamma)]$$

**until**

### *4.3. Properties of* `DualQuatE`

In this part we describe the relation patterns and introduce how `DualQuatE` expresses those patterns. Recently, learning relation patterns including **symmetry/antisymmetry**, **inversion** and **composition** have been realized to the key of link prediction task. Our model `DualQuatE` can easily explain the relation patterns of the learned relation embeddings and proof of relation patterns can be found in the Appendix A.

**Inversion**: If a relation $r' \in \mathcal{R}$ is the inverse to a relation $r \in \mathcal{R}$, then we can infer $(h, r, t) \in \Omega^+ \Leftrightarrow (t, r', h) \in \Omega^+$. For example, the relation *has_part* is inverse to the relation $part\_of$. To $r$ and $r'$, we infer that $(\mathbf{m}'\mathbf{m})\mathbf{h}(\mathbf{m}'\mathbf{m})^* + \mathbf{m}'\mathbf{n}\mathbf{m}' + \mathbf{n}' = \mathbf{h}$, which denotes the composition of component $\mathbf{m}$ and $\mathbf{m}'$ have no rotation (i.e., $(\mathbf{m}'\mathbf{m})\mathbf{h}(\mathbf{m}'\mathbf{m})^* = \mathbf{h}$) and the translation $\mathbf{n}$ which is rotated by $\mathbf{m}'$ is the opposite number of the translation $\mathbf{n}'$ (i.e., $\mathbf{m}'\mathbf{n}\mathbf{m}' + \mathbf{n}' = \mathbf{0}$).

**Symmetry**: A relation $r \in \mathcal{R}$ is symmetric, if $(h, r, t) \in \Omega^+ \Leftrightarrow (t, r, h) \in \Omega^+$ holds. For instance, relations *similar_to* and *verb_group* from the dataset WN18 are symmetric. If a relation is symmetric, we reason that $(\mathbf{m}\mathbf{m})\mathbf{h}(\mathbf{m}\mathbf{m})^* + \mathbf{m}\mathbf{n}\mathbf{m}^* + \mathbf{n} = \mathbf{h}$, which means no rotation of the self-composition of component $\mathbf{m}$ (i.e., $(\mathbf{m}\mathbf{m})\mathbf{h}(\mathbf{m}\mathbf{m})^* = \mathbf{h}$) and no translation of component $\mathbf{n}$ (i.e., $\mathbf{m}\mathbf{n}\mathbf{m}^* + \mathbf{n} = \mathbf{0}$).

**Antisymmetry**: A relation $r \in \mathcal{R}$ is antisymmetric, if $(h, r, t) \in \Omega^+ \Rightarrow (h, r, t) \in \Omega^-$, which satisfies $(\mathbf{m}\mathbf{m})\mathbf{h}(\mathbf{m}\mathbf{m})^* + \mathbf{m}\mathbf{n}\mathbf{m}^* + \mathbf{n} \neq \mathbf{h}$. For example, the relation $part\_of$.

**Composition**: A relation $r_3$ is composed by the relation $r_1$ and $r_2$, which can be denoted by $r_3 = r_1 \oplus r_2$ if $(h, r, s) \in \Omega^+ \wedge (s, r, t) \Rightarrow (t, r, h) \in \Omega^+$. For example, relation uncle_of can be composited by brother_of and father_of such as if $(Alva, brother\_of, Aaron)$, $(Aaron, father\_of, Abel)$ are true triples, we can reason $(Alva, uncle\_of, Abel)$ is a true fact in the real world. Relation $r_3$ can be composited by relation $r_1$ and $r_2$; they can be represented by $(\mathbf{m}_2\mathbf{m}_1)\mathbf{h}(\mathbf{m}_2\mathbf{m}_1)^* + \mathbf{m}_2\mathbf{n}_1\mathbf{m}_2^* + \mathbf{n}_2 = \mathbf{m}_3\mathbf{h}\mathbf{m}_3^* + \mathbf{n}_3$, which deduces that $\mathbf{n}_3$ is equal to the sum of translation $\mathbf{n}_2$ and translation $\mathbf{n}_1$ which is rotated by the rotation $\mathbf{m}_2$ (i.e., $\mathbf{m}_2\mathbf{n}_1\mathbf{m}_2^* + \mathbf{n}_2 = \mathbf{n}_3$).

### *4.4. Variations*

We introduce extensions of `DualQuatE`. `DualQuatE` is a transformation based model, which combines rotation and translation. To compare the effects of interaction of rotation and translation, we compare `DualQuatE` with `DualQuatE-1` which models relations as rotation in three-dimensional space. Furthermore, we propose `DualQuatE-2` to explore the role of scaling in the rotation.

`DualQuatE-1`: We devise `DualQuatE-1` which embeds entities and relations to quaternion space. Specifically, we represent entity embeddings $\mathbf{h}, \mathbf{t}$ with pure quaternions and relation embeddings $\mathbf{r}$ with quaternions. We design a score function as follows

$f_r(h,t) = -||\mathbf{rhr}^* - \mathbf{t}||$ to model the relation as rotation in three-dimensional space. Namely, for each fact satisfies: $\mathbf{rhr}^* = \mathbf{t}$.

`DualQuatE-2`: To explore the effect of scaling in knowledge graph embeddings, we present `DualQuatE-2` to introduce scaling. `DualQuatE-2` maps knowledge graph embeddings to four-dimensional space. Especially, we represent entities and relations with quaternions where relation embeddings are not unit quaternions. We define score function $f_r(h,t) = -||\mathbf{hr} - \mathbf{t}||$ meaning relation transform head entity to tail entity combining rotation and scaling.

### *4.5. Connection to TransE and RotatE*

**Compared with RotatE**: RotatE embedded entity embeddings $\mathbf{h}$, $\mathbf{t}$ and relation embeddings $\mathbf{r}$ into the complex space. RotatE utilized score function $-||\mathbf{h} \circ \mathbf{r} - \mathbf{t}||$ to calculate the probability of each triple, where $r_i$ is unit complex $\cos\theta + \mathbf{i}\sin\theta$. `DualQuatE` can be transformed to RotatE by fixing rotation plane and removing translation variables. For instance, we can construct relation embeddings by Formula (10) in $xoy$ plane, where $\mathbf{u} = \mathbf{k}$ and $\mathbf{n} = \mathbf{0}$ (i.e., $\mathbf{r} = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}\mathbf{i}$) and embed entities with corresponding forms: $\mathbf{h}$ or $\mathbf{t} = a\mathbf{i} + b\mathbf{j}$.

**Compared with TransE**: TransE modeled relation as translation that embedded entity embeddings $\mathbf{h}$, $\mathbf{t}$ and relation embeddings $\mathbf{r}$ to vector space. To express TransE, we can set $\theta = 0$ (i.e., $\mathbf{m} = \mathbf{0}$) in relation embeddings to ignore the rotation. In other words, the relation embedddings in `DualQuatE` can be expressed as $\mathbf{r} = 1 + \frac{\epsilon}{2}\mathbf{n}$.

## 5. Experiments

### *5.1. Experiment Settings*

#### 5.1.1. Datasets

We evaluated our approach on widely used datasets: WN18, FB15k, WN18RR, FB15k-237 and YAGO3-10, details of which are shown in Table 2. WN18 [4] is sampled from WordNet (https://wordnet.princeton.edu/ accessed on 11 June 2021), which is a knowledge graph about lexical relations of words. WN18RR [15] is a subset of WN18 with inverse relations removed. FB15k [4] is a large database with structured general human knowledge. FB15k-237 [22] is a subset of FB15k with reverse relations removed. YAGO3-10 [23] is a subset of YAGO3 which extends YAGO (https://io.datascience-paris-saclay.fr/dataset/YAGO accessed on 11 June 2021) in different languages. Tuples in YAGO3-10 mainly come from Wikipedia describing individuals, e.g., who lives in which city.

**Table 2.** Specific information of the experimental datasets. #E and #R denote entities and relations number in datasets, #TR, #V and #TE denote the size of training set, valid set and test set.

| Dataset | #E | #R | #TR | #V | #TE |
|---|---|---|---|---|---|
| FB15k | 14,951 | 1345 | 483,142 | 50,000 | 59,071 |
| FB15k-237 | 14,541 | 237 | 272,115 | 17,535 | 20,466 |
| WN8 | 40,943 | 18 | 141,442 | 5000 | 5000 |
| WN8RR | 40,943 | 11 | 86,835 | 3034 | 3134 |
| YOGA3-10 | 123,182 | 37 | 1,079,040 | 5000 | 5000 |

#### 5.1.2. Evaluation Metric

Similar to [6], we used three metrics to measure our approach, i.e., Mean Rank (MR), Mean Reciprocal Rank (MRR), and Hit@n. To calculate those metrices, we first replace $h$ by all entities $h' \in \mathcal{E}$ for each testing triplet $(h,r,t) \in \mathbb{T}$ (where $\mathbb{T}$ is a set of testing triplets) and compute score $f_r(h',t)$ for each triple $(h',r,t)$. After that we sort $h'$ according to score $f_r(h',t)$ ascendingly and get the rank of the original entity $h$, denoted by $\mathcal{K}(h)$. Note that

$\mathcal{K}(h)$ is the "rank" of $h$ instead of the score of $h$, e.g., if the score of $h$ the smallest, $\mathcal{K}(h)$ is 1. We can calculate MR as shown below:

$$\text{MR} = \frac{\sum_{(h,r,t)\in\mathbb{T}} \mathcal{K}(h)}{|\mathbb{T}|}$$

which means MR is an average of ranks of all the original entities in the testing triplets. Likewise, MRR can be calculated as follows:

$$\text{MRR} = \frac{\sum_{(h,r,t)\in\mathbb{T}} \frac{1}{\mathcal{K}(h)}}{|\mathbb{T}|}$$

which indicates MRR is an average of inverse ranks of all the original entities in the testing triplets. Hit@n suggests the proportion of original entities in the top $n$ entities, which can be calculated by:

$$\text{Hit@n} = \frac{\sum_{(h,r,t)\in\mathbb{T}} one(\mathcal{K}(h) \leq n)}{|\mathbb{T}|}$$

where $one(\mathcal{K}(h) \leq n)$ is 1 if $\mathcal{K}(h) \leq n$, and 0 if $\mathcal{K}(h) > n$. We tested different values $n$ = 1, 3, 10 in the evaluation, similar to the setting used in reference [6].

#### 5.1.3. Baselines

We compared our model with several state-of-the-art baselines. For transformation based models, we compared our model to TransE [4], TorusE [24], RotatE [5] and HAKE [7]; for bilinear models, we compared our model to ComplEx [13], HolE [12], SimplE [9], DihEdral [14] and QuatE [6] (to make the comparison fair, we use the version of QuatE without type constraints on the common link prediction datasets considering the requirement of type constraints is too strong).

#### 5.1.4. Implementation Details

We utilized Pytorch (https://pytorch.org accessed on 11 June 2021) to implement our model (https://github.com/gaoliming123/DualQuatE accessed on 11 June 2021) and its variations `DualQuatE-1` and `DualQuatE-2`. We tuned the hyperparameters as follows:the embedding dimension $k \in \{100, 200, 300, 500\}$, the learning rate $\in \{0.0001, 0.0003, 0.0005, 0.0008\}$, the ratio of negative sample $n \in \{32, 64, 128\}$, the margin $\gamma \in \{3, 6, 9, 12, 15, 18, 24\}$ and the self-adversarial sampling temperature $\alpha \in \{0.5, 1.0\}$. We adopt $k = 100$ for WN18RR and WN18, $k = 200$ for FB15k-237 and YAGO3-10 and $k = 500$ for FB15k.

### *5.2. Results*

Tables 3–5 show the experimental results on four datasets. The performance of `DualQuatE` and its variations represent comparability to state-of-the-art models. For YAGO3-10, the link prediction results are shown in Table 3, from which we can see that `DualQuatE` is competitive compared to most previous knowledge graph embedding models, especially in metric Hit@10. The result of YAGO3-10 tells us that the performance of `DualQuatE` is better than `DualQuatE-1`, which indicates that modeling relations as the interaction of rotation and translation with more degrees of freedom (as done by `DualQuatE`) is indeed better than simply modeling relations as rotation (as done by `DualQuatE-1`). Furthermore, the advanced results of `DualQuatE-2` and `DualQuatE` inspire us to explore the mixed effects of vector operations. Tables 4 and 5 indicate the effects of our models on four common datasets: WN18RR, FB15k-237, WN18 and FB15k. We can find that our models perform better on datasets WN18RR and FB15k-237; for WN18 and FB15k, metrics are almost close to previous models and several metrics surpass the previous.

**Table 3.** Link prediction on datasets YAGO3-10. ♠ represents the results that came from (Toutanova and Chen, 2015) and the others came from the original papers. The results with bold are the best and the underlined ones are the second best results.

| | YAGO3-10 | | | | |
|---|---|---|---|---|---|
| **Model** | **MR** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** |
| DistMult ♠ | 5926 | 0.34 | 0.24 | 0.38 | 0.54 |
| ComplEx ♠ | 6351 | 0.36 | 0.26 | 0.40 | 0.55 |
| ConvE ♠ | 1671 | 0.44 | 0.35 | 0.49 | 0.62 |
| RotatE | 1767 | 0.495 | 0.402 | 0.550 | 0.670 |
| InteractE | 2375 | 0.541 | **0.462** | - | 0.687 |
| HAKE | - | **0.545** | **0.462** | **0.596** | 0.694 |
| `DualQuatE-1` | 1636 | 0.477 | 0.377 | 0.534 | 0.672 |
| `DualQuatE-2` | 1889 | 0.503 | 0.411 | 0.557 | 0.676 |
| `DualQuatE` | **1210** | 0.534 | 0.445 | 0.591 | **0.695** |

**Table 4.** Link prediction on datasets FB15k-237 and WN18RR. ♣ represents the results that came from (Sun et al., 2019); the others are from the original papers. ¶ denotes the results of QuatE without type constraints from the paper. The results in bold are the best and the underlined ones are the second best results.

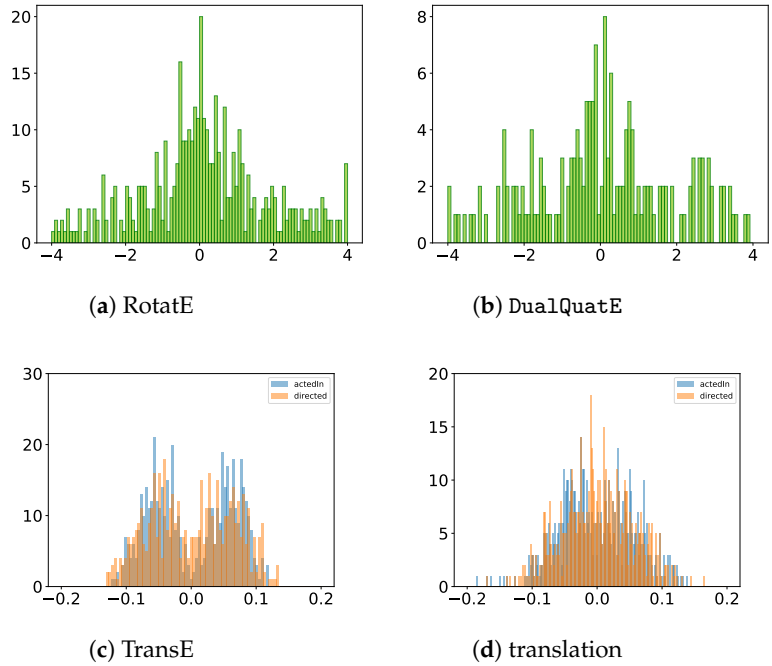
| | FB15k-237 | | | | | WN18RR | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **MR** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** | **MR** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** |
| TransE ♣ | 357 | 0.294 | - | - | 0.465 | 3384 | 0.226 | - | - | 0.501 |
| ComplEx ♣ | 339 | 0.247 | 0.158 | 0.275 | 0.428 | 5261 | 0.44 | 0.41 | 0.46 | 0.51 |
| RotatE ♣ | 177 | 0.338 | 0.241 | 0.375 | 0.533 | 3340 | 0.476 | 0.428 | 0.492 | 0.571 |
| DihEdral | - | 0.32 | 0.23 | 0.353 | 0.502 | - | 0.48 | **0.452** | 0.491 | 0.536 |
| QuatE ¶ | 176 | 0.311 | 0.221 | 0.342 | 0.495 | 3472 | 0.481 | 0.436 | 0.500 | 0.564 |
| InteractE | 172 | **0.354** | **0.263** | - | 0.535 | 5202 | 0.463 | 0.430 | - | 0.528 |
| HAKE | - | 0.346 | 0.250 | 0.381 | 0.542 | - | **0.497** | **0.452** | **0.516** | **0.582** |
| `DualQuatE-1` | 173 | 0.329 | 0.230 | 0.368 | 0.530 | 2989 | 0.463 | 0.408 | 0.484 | 0.571 |
| `DualQuatE-2` | 174 | 0.345 | 0.246 | **0.384** | **0.545** | 3324 | 0.484 | 0.437 | 0.503 | 0.576 |
| `DualQuatE` | **171** | 0.342 | 0.245 | 0.381 | 0.535 | **2755** | 0.470 | 0.415 | 0.493 | **0.582** |

**Table 5.** Link prediction on datasets FB15k and WN18. ♣ represents the results that came from (Sun et al., 2019) and the others came from the original papers. ¶ denotes the results of QuatE without type constraints from the paper. The results with bold are the best and the underlined ones are the second best results.

| | FB15k | | | | | WN18 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **MR** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** | **MR** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** |
| TransE ♣ | - | 0.463 | 0.297 | 0.578 | 0.749 | - | 0.495 | 0.113 | 0.888 | 0.943 |
| ComplEx | - | 0.692 | 0.599 | 0.759 | 0.840 | - | 0.941 | 0.936 | 0.945 | 0.947 |
| HolE | - | 0.524 | 0.402 | 0.613 | 0.739 | - | 0.938 | 0.930 | 0.945 | 0.949 |
| TorusE | - | 0.733 | 0.674 | 0.771 | 0.832 | - | 0.619 | 0.943 | 0.950 | 0.954 |
| SimplE | - | 0.727 | 0.660 | 0.773 | 0.838 | - | 0.942 | 0.939 | 0.944 | 0.947 |
| RotatE ♣ | 40 | **0.797** | **0.746** | **0.830** | **0.884** | 309 | **0.949** | **0.944** | 0.952 | 0.959 |
| DihEdral | - | 0.733 | 0.641 | 0.803 | 0.877 | - | 0.946 | 0.942 | 0.948 | 0.952 |
| QuatE ¶ | 41 | 0.770 | 0.700 | 0.821 | 0.878 | 388 | **0.949** | 0.941 | **0.954** | 0.960 |
| `DualQuatE-1` | **31** | 0.751 | 0.659 | 0.825 | **0.884** | 241 | 0.947 | 0.939 | 0.952 | 0.959 |
| `DualQuatE-2` | 50 | 0.766 | 0.696 | 0.818 | 0.877 | 220 | 0.948 | 0.942 | 0.953 | **0.961** |
| `DualQuatE` | 35 | 0.754 | 0.664 | 0.827 | **0.884** | **183** | **0.949** | 0.943 | 0.952 | 0.960 |

### *5.3. Relation Embeddings*

In this part we analyze the properties of `DualQuatE` learned for relations. `DualQuatE` can distinguish multiple relations between head and tail entities, for example, as shown in Figure 3. We compared our model with RotatE; Figure 3a,b display the difference of the representation of relation actedIn and directed. Figure 3a shows that the relations actedIn and directed are more similar where the gap between the two relations is clustered around zero. For `DualQuatE`, the difference which is shown in Figure 3b is more dispersed. Maybe the learned embeddings of our model are slightly concentrated around zero. We speculate that the reason for this result is due to less relations in YAGO3-10, which causes the diversity of relations between the entities to be more sparse. Figure 3d denotes the histrograms of translation component of relation actedIn and directed. Compared with the relation directed, the distribution of the relation actedIn is more decentralized. The values of translation component of relation directed are concentrated around zero, while the values of relation actedIn are more around $\pm 0.05$. Namely, head entity which is rotated by the rotation component of directed will be closer to tail entity. Figure 3c denotes that the distribution of the embeddings of relations actedIn and directed is very similar.




(**a**) RotatE (**b**) `DualQuatE`

(**c**) TransE (**d**) translation

**Figure 3.** Visualization of the multiple relations between the entities. (**a**) denotes the histrograms of different between the *actedIn* embeddings and *directed* embeddings of RotatE. (**b**) shows the histrograms of the `DualQuatE`. (**c**) denotes the histrograms of *actedIn* and *directed* embeddings of TransE. (**d**) displays the histrograms of translation embeddings of *actedIn* and *directed* of `DualQuatE`.

Limited by the length of the article, we visualize only some relation patterns in this paper. Figure 4a shows that the self composition of rotation $\mathbf{m}$ is close to 0 or $2\pi$. Figure 4b,c show the rotation embeddings to antisymmetry relation has_part from dataset WN18. For inversion relations $r$ and $r'$, Figure 4d denotes the embeddings of rotation elements between $\mathbf{m}$ and $\mathbf{m}'$ which means the composition of $\mathbf{m}$ and $\mathbf{m}'$ is 0 or $2\pi$.

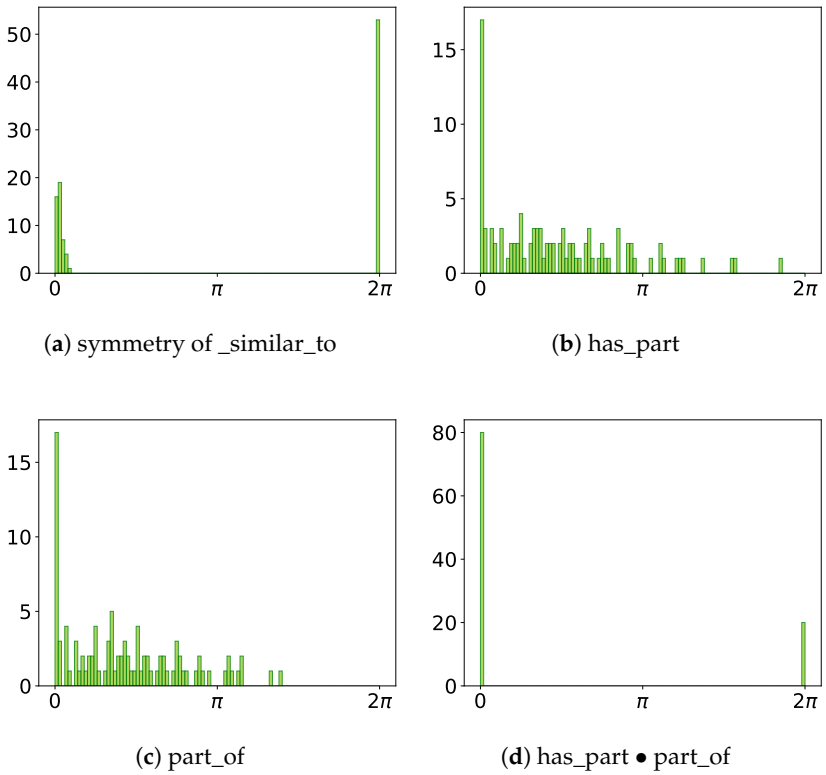



**Figure 4.** Visualization of relation patterns represented by `DualQuatE` in rotation, where (**a**) denotes the symmetry relation "similar_to" in rotation of **mm**, (**b**,**c**) are rotations of "has_part" and "part_of", and (**d**) exhibits the inversion effects, where "has_part" • "part_of" represents **mm**′.

### *5.4. Space and Time Complexity*

In this part, we list space and time complexity of the different transformation based models and bilinear models as shown Table 6. $m$ and $n$ denote number of entities and relations. $d$ is dimensions of entity or relation embeddings.

**Table 6.** Space and Time Complexity.

| Method | Space Complexity | Time Complexity |
|---|---|---|
| TransE | $\mathcal{O}(nd + md)$ | $\mathcal{O}(d)$ |
| RotatE | $\mathcal{O}(nd + md)$ | $\mathcal{O}(d)$ |
| HAKE | $\mathcal{O}(nd + md)$ | $\mathcal{O}(d)$ |
| RESCAL | $\mathcal{O}(nd + md^2)$ | $\mathcal{O}(d^2)$ |
| HolE | $\mathcal{O}(nd + md)$ | $\mathcal{O}(dlogd)$ |
| ComplEx | $\mathcal{O}(nd + md)$ | $\mathcal{O}(d)$ |
| QuatE | $\mathcal{O}(nd + md)$ | $\mathcal{O}(d)$ |
| `DualQuatE` | $\mathcal{O}(nd + md)$ | $\mathcal{O}(d)$ |

## 6. Conclusions

In this paper, we propose a novel model, `DualQuatE`, for knowledge graph embedding, which maps entities and relations to dual quaternion space. We present a new score function to model each relation as interaction of rotation and translation, which addresses the multiple relations between two entities. We demonstrate that our model is able to express main relation patterns and outperforms state-of-the-art baselines. However, `DualQuatE` does not consider temporal information and semantic hierarchies in knowledge graphs. In the feature, we will investigate how to explore temporal information and semantic hierarchies based on our model. It is also interesting to investigate the possibility of

applying our `DualQuatE` model to learning representations of propositions for helping learning action models [25–28] and recognizing plans [29–31] in planning community.

**Author Contributions:** Conceptualization, L.G., H.Z. and H.H.Z.; methodology, L.G. and H.Z.; software, L.G.; validation, L.G., H.Z. and H.H.Z.; formal analysis, L.G. and H.H.Z.; investigation, L.G. and H.Z.; resources, L.G., H.Z. and H.H.Z.; data curation, L.G.; writing—original draft preparation, L.G.; writing—review and editing, H.Z. and H.H.Z.; visualization, L.G.; supervision, H.Z., H.H.Z. and J.X.; project administration, L.G.; funding acquisition, H.Z. and H.H.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the National Natural Science Foundation of China (Grant No. 62076263), the National Natural Science Foundation of China for Young Scientists of China (Grant No. 11701592), the Joint Funds of the National Natural Science Foundation of China (Grant No. U1811263), Guangdong Natural Science Funds for Distinguished Young Scholar (Grant No. 2017A030306028), Guangdong special branch plans young talent with scientific and technological innovation (Grant No. 2017TQ04X866), Pearl River Science and Technology New Star of Guangzhou and Guangdong Province Key Laboratory of Big Data Analysis and Processing.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Publicly available datasets were analyzed in this study. These data can be found here: https://github.com/gaoliming123/DualQuatE, accessed on 11 June 2021.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix A. Proof of Relation Patterns

**symmetry**: Relation $r \in \mathcal{R}$ is symmetry, then both $(h, r, t)$ and $(t, r, h) \in \mathcal{G}$. The following equations will hold:

$$\begin{aligned} \mathbf{r} \otimes (1 + \epsilon\mathbf{h}) \otimes \mathbf{r}^{\diamond} &= (1 + \epsilon\mathbf{t}) \\ \mathbf{r} \otimes (1 + \epsilon\mathbf{t}) \otimes \mathbf{r}^{\diamond} &= (1 + \epsilon\mathbf{h}) \end{aligned} \tag{A1}$$

Then we deduce that:

$$\mathbf{mhm}^{*} + \mathbf{n} = \mathbf{t} \tag{A2}$$

$$\mathbf{mtm}^{*} + \mathbf{n} = \mathbf{h} \tag{A3}$$

Bring Formula (5) into Formula (6), and then we can get:

$$\begin{aligned} \mathbf{m}(\mathbf{mhm}^{*} + \mathbf{n})\mathbf{m}^{*} + \mathbf{n} &= \mathbf{h} \\ (\mathbf{mm})\mathbf{h}(\mathbf{mm})^{*} + \mathbf{mnm}^{*} + \mathbf{n} &= \mathbf{h} \end{aligned} \tag{A4}$$

**Inversion**: Relation $r \in \mathcal{R}$ is inversion, iff another relation $r' \in \mathcal{R}$ exists and satisfies both $(h, r, t)$ and $(t, r', h) \in \mathcal{G}$. Then the following equations will hold:

$$\begin{aligned} \mathbf{r} \otimes (1 + \epsilon\mathbf{h}) \otimes \mathbf{r}^{\diamond} &= (1 + \epsilon\mathbf{t}) \\ \mathbf{r}' \otimes (1 + \epsilon\mathbf{t}) \otimes \mathbf{r}'^{\diamond} &= (1 + \epsilon\mathbf{h}) \end{aligned} \tag{A5}$$

Then we deduce that:

$$\mathbf{mhm}^{*} + \mathbf{n} = \mathbf{t} \tag{A6}$$

$$\mathbf{m}'\mathbf{tm}'^{*} + \mathbf{n} = \mathbf{h} \tag{A7}$$

Bring Formula (10) into Formula (11), and then we can get:

$$\begin{aligned} \mathbf{m}(\mathbf{mhm}^* + \mathbf{n})\mathbf{m}'^* &= \mathbf{h} \\ (\mathbf{m}'\mathbf{m})\mathbf{h}(\mathbf{m}'\mathbf{m})^* + \mathbf{m}'\mathbf{n}\mathbf{m}' + \mathbf{n}' &= \mathbf{h} \end{aligned} \tag{A8}$$

**Composition**: A relation $r_3$ is the composition of relation $r_1$ and $r_2$, which can be denoted by $r_3 = r_1 \oplus r_2$.

$$\mathbf{m}_1\mathbf{hm}_1^* + \mathbf{n}_1 = \mathbf{s} \tag{A9}$$

$$\mathbf{m}_2\mathbf{sm}_2^* + \mathbf{n}_2 = \mathbf{t} \tag{A10}$$

$$\mathbf{m}_3\mathbf{sm}_3^* + \mathbf{n}_3 = \mathbf{t} \tag{A11}$$

Then we can get:

$$\begin{aligned} \mathbf{m}_2(\mathbf{m}_1\mathbf{hm}_1^*)\mathbf{m}_2^* + \mathbf{n}_2 &= \mathbf{m}_3\mathbf{hm}_3^* + \mathbf{n}_3 \\ (\mathbf{m}_2\mathbf{m}_1)\mathbf{h}(\mathbf{m}_2\mathbf{m}_1)^* + \mathbf{m}_2\mathbf{n}_1\mathbf{m}^* + \mathbf{n}_2 &= \mathbf{m}_3\mathbf{hm}_3^* + \mathbf{n}_3 \end{aligned} \tag{A12}$$

## References


1. Chen, Z.; Wang, X.; Xie, X.; Wu, T.; Bu, G.; Wang, Y.; Chen, E. *Co-Attentive Multi-Task Learning for Explainable Recommendation*; Kraus, S., Ed.; IJCAI: Macao, China, 2019; pp. 2137–2143.
2. Kumar, V.; Hua, Y.; Ramakrishnan, G.; Qi, G.; Gao, L.; Li, Y. Difficulty-Controllable Multi-hop Question Generation from Knowledge Graphs. In Proceedings of the Semantic Web—ISWC 2019—18th International Semantic Web Conference, Auckland, New Zealand, 26–30 October 2019; Ghidini, C., Hartig, O., Maleshkova, M., Svátek, V., Cruz, I.F., Hogan, A., Song, J., Lefrançois, M., Gandon, F., Eds.; Part I; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2019; Volume 11778, pp. 382–398.
3. Kanakaris, N.; Giarelis, N.; Siachos, I.; Karacapilidis, N. Shall I Work with Them? A Knowledge Graph-Based Approach for Predicting Future Research Collaborations. *Entropy* **2021**, *23*, 664. [CrossRef] [PubMed]
4. Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 2787–2795.
5. Sun, Z.; Deng, Z.; Nie, J.; Tang, J. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. *arXiv* **2019**, arXiv:1902.10197.
6. Zhang, S.; Tay, Y.; Yao, L.; Liu, Q. Quaternion Knowledge Graph Embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019*; Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R., Eds.; NeurIPS: Vancouver, BC, Canada, 2019; pp. 2731–2741.
7. Zhang, Z.; Cai, J.; Zhang, Y.; Wang, J. *Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction*; AAAI Press: Palo Alto, CA, USA, 2020; pp. 3065–3072.
8. Yang, B.; Yih, W.; He, X.; Gao, J.; Deng, L. *Embedding Entities and Relations for Learning and Inference in Knowledge Bases*; Bengio, Y., LeCun, Y., Eds.; ICLR: San Diego, CA, USA, 2015.
9. Kazemi, S.M.; Poole, D. *SimplE Embedding for Link Prediction in Knowledge Graphs*; Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R., Eds.; NeurIPS: Montréal, QC, Canada, 2018; pp. 4289–4300.
10. Wang, Q.; Mao, Z.; Wang, B.; Guo, L. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 2724–2743. [CrossRef]
11. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. *Learning Entity and Relation Embeddings for Knowledge Graph Completion*; Bonet, B., Koenig, S., Eds.; AAAI Press: Palo Alto, CA, USA, 2015; pp. 2181–2187.
12. Nickel, M.; Rosasco, L.; Poggio, T.A. *Holographic Embeddings of Knowledge Graphs*; Schuurmans, D., Wellman, M.P., Eds.; AAAI Press: Palo Alto, CA, USA, 2016; pp. 1955–1961.
13. Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; Bouchard, G. *Complex Embeddings for Simple Link Prediction*; ICML: New York, NY, USA, 2016; pp. 2071–2080.
14. Xu, C.; Li, R. *Relation Embedding with Dihedral Group in Knowledge Graph*; ACL: Florence, Italy, 2019; pp. 263–272.
15. Dettmers, T.; Minervini, P.; Stenetorp, P.; Riedel, S. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*; AAAI Press: Palo Alto, CA, USA, 2018; pp. 1811–1818.

16. Schlichtkrull, M.S.; Kipf, T.N.; Bloem, P.; van den Berg, R.; Titov, I.; Welling, M. Modeling Relational Data with Graph Convolutional Networks. In Proceedings of the Semantic Web—15th International Conference, ESWC 2018, Heraklion, Crete, Greece, 3–7 June 2018; Gangemi, A., Navigli, R., Vidal, M., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., Alam, M., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2018; Volume 10843, pp. 593–607.
17. Vashishth, S.; Sanyal, S.; Nitin, V.; Agrawal, N.; Talukdar, P.P. *InteractE: Improving Convolution-Based Knowledge Graph Embeddings by Increasing Feature Interactions*; AAAI Press: Palo Alto, CA, USA, 2020; pp. 3009–3016.
18. Balazevic, I.; Allen, C.; Hospedales, T. Multi-relational poincaré graph embeddings. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 4463–4473.
19. Chami, I.; Wolf, A.; Juan, D.; Sala, F.; Ravi, S.; Ré, C. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020; Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R., Eds.; Association for Computational Linguistics: Beijing, China, 2020; pp. 6901–6914.
20. Hamilton, W.R. LXXVIII. On quaternions; or on a new system of imaginaries in Algebra: To the editors of the Philosophical Magazine and Journal. *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **1844**, *25*, 489–495. [CrossRef]
21. Kotelnikov, A.P. Screw calculus and some applications to geometry and mechanics. *Ann. Imp. Univ. Kazan* **1895**, *24*.
22. Toutanova, K.; Chen, D. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*; Association for Computational Linguistics: Beijing, China, 2015; pp. 57–66.
23. Mahdisoltani, F.; Biega, J.; Suchanek, F.M. YAGO3: A Knowledge Base from Multilingual Wikipedias. In Proceedings of the Seventh Biennial Conference on Innovative Data Systems Research (CIDR 2015), Asilomar, CA, USA, 4–7 January 2015.
24. Ebisu, T.; Ichise, R. TorusE: Knowledge Graph Embedding on a Lie Group. In *Proceedings of the AAAI Conference on Artificial Intelligence*; AAAI Press: Palo Alto, CA, USA, 2018; pp. 1819–1826.
25. Zhuo, H.H.; Muñoz-Avila, H.; Yang, Q. Learning hierarchical task network domains from partially observed plan traces. *Artif. Intell.* **2014**, 212, 134–157. [CrossRef]
26. Zhuo, H.H.; Yang, Q. Action-model acquisition for planning via transfer learning. *Artif. Intell.* **2014**, *212*, 80–103. [CrossRef]
27. Zhuo, H.H.; Zha, Y.; Kambhampati, S. Discovering Underlying Plans Based on Shallow Models. *ACM Trans. Intell. Syst. Technol.* **2020**, *11*, 18:1–18:30. [CrossRef]
28. Zhuo, H.H.; Kambhampati, S. Model-lite planning: Case-based vs. model-based approaches. *Artif. Intell.* **2014**, *246*, 1–21. [CrossRef]
29. Zhuo, H.H. Recognizing Multi-Agent Plans When Action Models and Team Plans Are Both Incomplete. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 30:1–30:24. [CrossRef]
30. Feng, W.; Zhuo, H.H.; Kambhampati, S. Extracting Action Sequences from Texts Based on Deep Reinforcement Learning. In Proceedings of the International Joint Conferences on Artifical Intelligence (IJCAI), Stockholm, Sweden, 13–19 July 2018; pp. 4064–4070.
31. Zhuo, H.H. Human-Aware Plan Recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*; AAAI Press: Palo Alto, CA, USA, 2017; pp. 3686–3693.