# TERRF: A Temporal Knowledge Graph Embedding Model via Rodrigues' Rotation Formula

**Anonymous ACL submission**


## Abstract

Temporal knowledge graph completion (TKGC) aims to predict missing facts at different timestamps. A promising solution for this task is learning temporal knowledge graph representations in vector space, focusing on modeling important relation patterns inherent in temporality. However, existing methods often involve complex spatial transformations, such as expanding into complicated spaces, which might sacrifice computational efficiency. Additionally, relying solely on individual geometric operation also limits representational ability, thereby hindering the predictive performance. To address these challenges, this study introduces a **T**emporal knowledge graph **E**mbedding model via **R**odrigues' **R**otation **F**ormula (TERRF) for TKGC. TERRF treats link prediction as a rigid body transformation in three-dimensional space, comprising two key operations: a Normalized Scaling operation and an Efficient Rotation operation. The Normalized Scaling operation sets an initial position for entities, allowing for more flexible rotations, while the Efficient Rotation operation uses Rodrigues' Rotation Formula, requiring only an axis and angle representation. Experimental results show that our proposed TERRF model significantly outperforms competitive baseline models and achieves state-of-the-art results on three popular benchmark datasets.


## 1 Introduction

Knowledge graphs (KGs) form the core of diverse real-world applications, including but not limited to question answering (Zhang et al., 2024), information retrieval (Ziems et al., 2024), and recommender systems (Wasi, 2024). KGs store structured facts in the form of triples $(s, r, o)$, where $s$, $r$, and $o$ denote the subject, relation, and object, respectively. Despite containing millions of entities and billions of facts, large-scale KGs, such as YAGO (Fabian et al., 2007), Freebase (Bollacker et al., 2008), and Wikidata (Erxleben et al., 2014) remain particularly incomplete. As such, knowledge graph completion (also known as link prediction) has gained widespread attention in recent years.

However, real-world KGs continue growing inherently tied to specific timestamps. For example, the triple (*Barack Obama, Make a visit, South Korea*) is valid on *2014-04-18*. Therefore, temporal knowledge graphs (TKGs) are introduced as quadruples $(s, r, o, \tau)$ to describe facts that evolve over time $\tau$. Given the incompleteness of TKGs, we focus on temporal knowledge graph completion (TKGC), aiming to infer missing temporal links from a TKG. One prominent solution for this task is knowledge graph embedding (KGE), which learns low-dimensional representations for each elements in static/temporal KGs, and then compute plausibility scores for all possible facts.

This has led to the development of a wide range of TKGC work, building on static KGE models (Leblay and Chekol, 2018; García-Durán et al., 2018; Dasgupta et al., 2018). As a promising approach used in static KGE, learning rotation transformation parametrized by relation has gained popularity, where the idea is to rotate the subject entity to fall near its corresponding object entity (Sun et al., 2019; Zhang et al., 2019). When it comes to TKGs, there are two major categories of rotation-based temporal KGE methods: The first category involves learning embeddings in complex or hypercomplex spaces by adding imaginary dimensions (Xu et al., 2020; Chen et al., 2022; Li et al., 2023). The second category, based on (Saxe et al., 2014), aims to rotate entities via orthogonal transformations in the real number system $\mathbb{R}^n$, treating each relation as an orthogonal matrix $\in \mathbb{R}^{n\times n}$. Nevertheless, there are still two limitations:

- For one thing, extending to hypercomplex spaces or defining each relation as an orthogonal matrix for high-dimensional rotation might yield an reduction in computational efficiency, and the space cost is also highly related to the number of entities and relations.

- For another, single type of operation for KGE is insufficient, since each operator may have modeling limitations for different relational patterns (Ying et al., 2024). This indicates that flexible transformations in space are crucial for temporal KGE.

Towards these problems, we propose TERRF, a **T**emporal knowledge graph **E**mbedding model via **R**odrigues' **R**otation **F**ormula (Wang and Dai, 2023). Generally, TERRF can be regarded as a three-dimensional (3D) rigid body transformation that includes both rotation and scaling. We give a simple way to define any such transformation by a scaling factor to set the initial position, a direction of the rotation axis, and an angle of rotation. Specifically, a Normalized Scaling operation parameterized by relation and timestamp is initially applied to determine a start position for the subject entity. Then, we introduce a more efficient method for implementing rotation based on Rodrigues' Rotation Formula, which requires only a unit vector and a scalar value to represent the axis and angle of rotation, respectively. Subsequently, the subject entity is rotated, and the plausibility of the quadruple is assessed by computing the embedding similarity between the subject and the object entities. Unlike previous methods that treat relations and timestamps solely as rotations, TERRF also integrates entity features into the rotation construction to achieve a more flexible transformation. Notably, TERRF executes 3D rotation operations in a more straightforward way, and also enhances the flexibility in modeling complex relation patterns.

Experimental results on three challenging benchmarks show that our proposed TERRF outperforms various baseline models, demonstrating the effectiveness of our approach. Additionally, we validate in our experiments that TERRF has more flexible transformation capabilities. From a mathematical perspective, we prove that TERRF can simultaneously model multiple important temporal relation patterns, which is shown in Appendix A.

## 2 Related Work

### 2.1 Static Knowledge Graph Embedding

Traditional static knowledge graph embedding is popularized by distance-based models, such as TransE (Bordes et al., 2013), which capture the relationship between entities by using the semantic distance. To address complex relationships, various extensions, such as TransH (Wang et al., 2014), TransR (Lin et al., 2015), and TransD (Ji et al., 2015) have been proposed following TransE. To enable KGE models to represent more relation patterns, including symmetry, antisymmetry, inversion, and composition, RotatE (Sun et al., 2019) teat each relation as 2D rotation in the complex space $\mathbb{C}$. Beyond complex representations, QuatE (Zhang et al., 2019) explores 3D rotations in hypercomplex space $\mathbb{H}$. To mitigate the issue of space cost, RotateCT (Dong et al., 2022) combines translation and 2D rotation operations in the complex space. DCNE (Dong et al., 2024) represents relations as rotations in 2D space using dual complex number multiplication.

### 2.2 Temporal Knowledge Graph Embedding

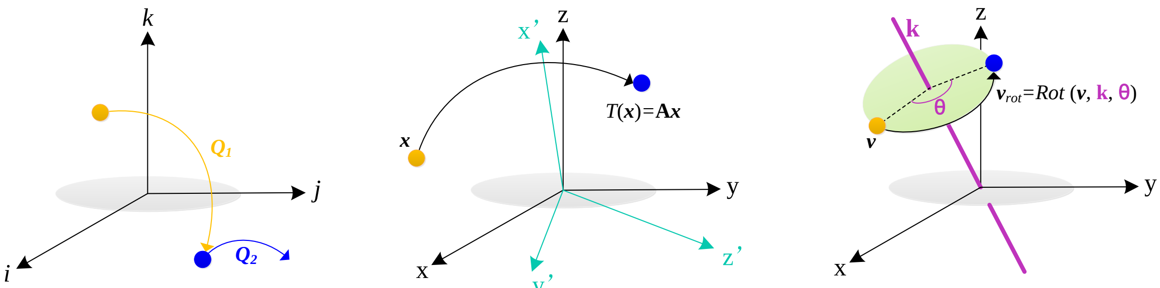
Most temporal KGE methods extend static KG embedding techniques to TKGs. TTransE (Leblay and Chekol, 2018) builds upon TransE (Bordes et al., 2013) by incorporating timestamp information as an additional element that controls the translation. TA-DistMult (García-Durán et al., 2018) is proposed on the basis of the DistMult (Yang et al., 2014) method. HyTE (Dasgupta et al., 2018) maps the factual triples associated with the start time onto hyperplanes. Hibrid-TE (Wang and Li, 2019) integrates both of TransD (Ji et al., 2015) and HyTE by projecting entities and relations onto hyperplanes constructed from time spans. DE-SimplE (Goel et al., 2020) utilizes a diachronic entity embedding function which captures the characteristics of entities at a specific time. TeRo (Xu et al., 2020) defines the temporal evolution of entities as rotations in the complex vector space. TeAST (Li et al., 2023) models TKGs in the complex space, representing the relations on Archimedean spiral timelines. CompoudE (Ge et al., 2022) and TCompoundE (Ying et al., 2024) represent relations and timestamps as different geometric operations.

Our proposed TERRF leverages scaling and 3D rotation in real number system to offer a comprehensive and expressive representation of TKGs.

## 3 Background

### 3.1 Hamilton's Quaternions

Quaternion belongs to hypercomplex number system $\mathbb{H}$, extending traditional complex number sys-




(A) Visualization of $Q_2 \otimes Q_1$ in quaternion space ($\otimes$: Hamilton product)

(B) Transform $\boldsymbol{v}$ via an orthogonal matrix $\mathbf{A} \in \mathbb{R}^{3\times 3}$

(C) **This work:** given an axis $\mathbf{k} \in \mathbb{R}^3$ and an angle $\theta \in \mathbb{R}^1$, rotate $\boldsymbol{v}$ via Rodrigues' Rotation Formula

Figure 1: Illustrations of the rotation via Hamilton's Quaternions, Orthogonal Transformation, and Rodrigues' Rotation Formula, respectively.

tem $\mathbb{C}$ to 4D space. A quaternion can represent a rotation in 3D space with the expression: $Q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$, where the coefficients $a$, $b$, $c$, $d$ are real numbers and $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ are imaginary units pointing along the three spatial axes. Given quaternions $Q_1$ and $Q_2$, a composite spatial rotation can be modeled with quaternions Hamilton product: $Q_2 \otimes Q_1$, which is shown in Figure 1(A).

Based on this, static KGE model QuatE (Zhang et al., 2019) and temporal KGE model TLT-KGE (Zhang et al., 2022) have modeled link prediction as rotation in quaternion space. Nevertheless, expanding to complicated spaces such as quaternion leads to more dimensions than real number, which increases the space cost and greatly reduces the computational efficiency (Dong et al., 2024).

### 3.2 Orthogonal Transformation

Orthogonal transformation (Saxe et al., 2014) is a linear transformation $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ in a real inner product space. Specifically, $\forall \mathbf{x} \in \mathbb{R}^n$, it holds that $||T(\mathbf{x})|| = ||\mathbf{x}||$, and $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, $\langle \mathbf{x}, \mathbf{y} \rangle = \langle T(\mathbf{x}), T(\mathbf{y}) \rangle$. This means that orthogonal transformations rotate the coordinate system without changing the length of each vector and the angles between vectors (as shown in Figure 1(B)). In mathematics, an orthogonal transformation can be represented by an orthogonal matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, *i.e.*, $T(\mathbf{x}) = \mathbf{A}\mathbf{x}$.

Previous KGE method (Tang et al., 2020) have represented each relation as a orthogonal matrix through Gram-Schmidt process for rotation transformation. However, this approach increases the number of parameters required to express relations and incurs computational overhead due to the matrix orthogonalization process.

### 3.3 Rodrigues' Rotation Formula

Given the limitations of Hamilton's Quaternions and orthogonal transformation, a more concise and effective method for link prediction is highly desirable.

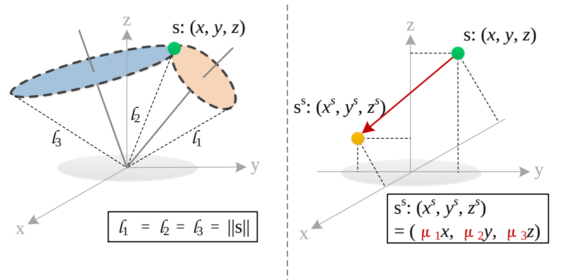
In mathematics, Rodrigues' Rotation Formula (Wang and Dai, 2023) constitutes an efficient algorithm for rotating a vector around a specified axis by a given angle in 3D real number system (as shown in Figure 1(C)). $\forall \mathbf{v} \in \mathbb{R}^3$, a rotated vector $\mathbf{v}_{rot}$ is expressed as:

$$\mathbf{v}_{rot} = Rot(\mathbf{v}, \mathbf{k}, \theta)$$
$$= \cos(\theta)\mathbf{v} + \sin(\theta)\mathbf{k} \times \mathbf{v} + (1 - \cos\theta)(\mathbf{k} \cdot \mathbf{v})\mathbf{k}, \quad (1)$$

where $\mathbf{k} \in \mathbb{R}^3$ stands for a unit vector defining an axis of rotation; $\theta \in \mathbb{R}^1$ represents the angle by which $\mathbf{v}$ is rotated about the axis $\mathbf{k}$ according to the right-hand rule; $\times$ and $\cdot$ denote cross product and dot product, respectively. As such, any rotation operation in 3D space can be accomplished using a single unit vector and an angle.

## 4 Methodology

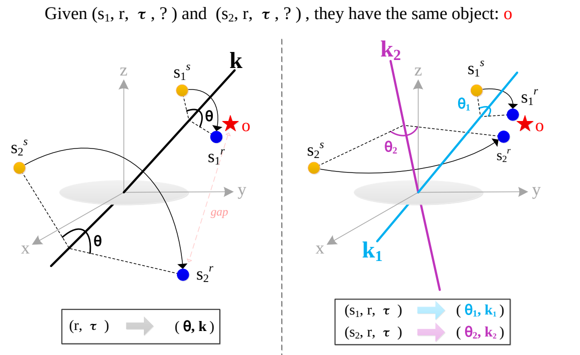
Formally, let $\mathcal{E}$ represent the set of entities, $\mathcal{R}$ refers to the set of relations, and $\mathcal{T}$ stands for the set of timestamps. A temporal knowledge graph $\mathcal{G}$ is defined as a collection of factual quadruples $\{(s, r, o, \tau)\}$, where $s, o \in \mathcal{E}$, $r \in \mathcal{R}$, and $\tau \in \mathcal{T}$. Our goal is to predict the object for $(s, r, ?, \tau)$, and the subject for $(?, r, o, \tau)$. Given all facts occurring at time $\tau$, we denote $\mathbf{e}_s$, $\mathbf{e}_r$, $\mathbf{e}_o$, $\mathbf{e}_\tau$ as the embedding of $s$, $r$, $o$, $\tau$, respectively, each of which $\in \mathbb{R}^n$. To achieve 3D transformations, we uniformly partition each embedding into a combination of local vectors with a magnitude of 3 for processing, rep-




(A) The issue of directly using rotation.

(B) Our solution

Figure 2: Illustration of Normalized Scaling. (A) Given a point $s$, direct rotation operations will confine $s$ to the surface of a 3D hypersphere with radius $||s||$, limiting spatial flexibility. (B) To address this, we perform a dynamical scaling operation on $s$ before rotating. This scales the coordinates of $s$ along each axis to obtain new coordinates $s^s$, which serve as the starting point for subsequent rotations.




(A) The issue of only integrating r, $\tau$ for rotation construction.

(B) Our solution

Figure 3: Illustration of axis and angle constructions, where $s_1^s$, $s_2^s$ are the positions after applying Normalized Scaling to $s_1$ and $s_2$; $s_1^r$ and $s_2^r$ are the rotated positions. (A) Constructing the direction axis $\mathbf{k}$ and rotation angle $\theta$ solely from the relation $r$ and timestamp $t$ might make it difficult to simultaneously position $s_1^r$ and $s_2^r$ close to the object $o$. (B) To address this, we incorporate entity features $s$ into the construction process, allowing the rotation process to adapt to changes in entity information and enhancing spatial flexibility.

resented as:

$$\mathbf{e} = \bigoplus_{i=0}^{\frac{n}{3}-1} \mathbf{e}^{[3i:\ 3i+3]}, \tag{2}$$

where $\mathbf{e}$ denotes $\mathbf{e}_s$, $\mathbf{e}_r$, or $\mathbf{e}_\tau$; $\bigoplus$ denotes the concatenation. Then, we propose a 3D rigid body transformation (TERRF) for temporal KGE, consisting of two operations: *Normalized Scaling* and *Rotation via Rodrigues' Rotation Formula*. Note that the rotation implemented in this paper just involves rotating around an axis from the origin by a certain angle.

### 4.1 Normalized Scaling

Due to the limitations of direct rotation operations (see Figure 2 (A)), we apply a Normalized Scaling operation before performing rotations (as shown in Figure 2 (B)). Such an operation is employed to determine an initial position for each entity according to different relations and timestamps. Specifically, we design a scaling factor (a unit vector) in 3D space. First, we define the fusion vector $\mathbf{e}_{r,\tau} = \mathbf{e}_r + \mathbf{e}_\tau$. Then, for the $i$-th segment vector of the subject entity, the scaling vector is computed as:

$$\tilde{\mathbf{e}_s}^{[3i:\ 3i+3]} = \frac{h(\mathbf{e}_{r,\tau})^{[3i:\ 3i+3]}}{||h(\mathbf{e}_{r,\tau})^{[3i:\ 3i+3]}||} \circ \mathbf{e}_s^{[3i:\ 3i+3]}, \tag{3}$$

where $h(\cdot)$ denotes a linear layer; all of $\mathbf{e}_s^{[3i:\ 3i+3]}$, $h(\mathbf{e}_{r,\tau})^{[3i:\ 3i+3]}$ and $\tilde{\mathbf{e}_s}^{[3i:\ 3i+3]} \in \mathbb{R}^3$; $\circ$ refers to the Hadamard (or element-wise) product. The reason for normalizing the scaling factors is to prevent overfitting.

### 4.2 Rotation via Rodrigues' Rotation Formula

Subsequently, we employ a more flexible 3D space transformation to achieve the reasoning process for TKGC. Our goal is to use a simpler way to characterize rotations in 3D space. Unlike existing methods that utilize hypercomplex numbers or orthogonal matrices, we construct only an **Axis Vector** and **Angle Value** to perform rotation operations. We naturally consider relation $r$ and timestamp $\tau$ as the necessary factors for representing the geometric transformation. However, as shown in Figure 3 (A), when $r$ and $\tau$ are fixed, the spatial transformation of the subject entity will also be fixed. To prevent the limitations arising from using the same axis and angle for all entities, we incorporate entity features into the construction of the rotation axis and angle (as shown in Figure 3 (B)). As such, we define a new fusion vector $\mathbf{e}_{s,r,\tau} = \mathbf{e}_s + \mathbf{e}_r + \mathbf{e}_\tau$

**Axis Vector.** The axis vector is defined by a normalization process:

$$\mathbf{k}^{[3i:\ 3i+3]} = \frac{\mathcal{F}_{\mathbf{k}}(\mathbf{e}_{s,r,\tau})^{[3i:\ 3i+3]}}{||\mathcal{F}_{\mathbf{k}}(\mathbf{e}_{s,r,\tau})^{[3i:\ 3i+3]}||}, \tag{4}$$

where $\mathcal{F}_{\mathbf{k}}(\cdot)$ is a linear layer that incorporates $\mathbf{e}_s$, $\mathbf{e}_r$, and $\mathbf{e}_\tau$; the segment $\mathbf{k}^{[3i:\ 3i+3]}$ is a unit vector, which reflects the $i$-th direction of the axis around which $\tilde{\mathbf{e}_s}^{[3i:\ 3i+3]}$ will rotate.

**Angle Value.** We obtain the angle representation by converting $\mathbf{e}_{s,r,\tau}$ to

$$\theta_i = 2\pi \circ \sigma(\mathcal{F}(\mathcal{F}_\theta(\alpha \mathbf{e}_{s,r,\tau})^{[3i:\,3i+3]})), \quad (5)$$

where $\theta_i \in \mathbb{R}^1$ is the $i$-th calculated angle; $\sigma(\cdot)$ is the Tanh function to constrain $\theta_i \in (-2\pi, 2\pi)$; $\mathcal{F}_\theta(\cdot)$ denotes a linear layer; $\mathcal{F}(\cdot)$ is employed to further project the segment of the hidden representation from $\mathbb{R}^3$ to $\mathbb{R}^1$; $\alpha$ is the angle weight, which is implemented as a hyperparameter.

After getting the axis vector and angle value, we apply the Rodrigues' Rotation Formula to $\tilde{\mathbf{e}_s}$, and obtain the rotated vector $\mathbf{e}_s^\star$:

$$\mathbf{e}_s^{\star[3i:\,3i+3]} = Rot(\tilde{\mathbf{e}_s}^{[3i:\,3i+3]}, \mathbf{k}^{[3i:\,3i+3]}, \theta_i). \quad (6)$$

The score function is defined as the inner product between $\mathbf{e}_s^\star$ and $\mathbf{e}_o$:

$$\phi(s, r, o, \tau) =< \mathbf{e}_s^\star, \mathbf{e}_o > . \quad (7)$$

### 4.3 Training

With the scoring function $\phi(s, r, o, \tau)$, the likelihood of any $o \in \mathcal{E}$ answering the query $(s, r, ?, \tau)$ can be computed as:

$$\mathcal{P}(o|s, r, \tau) = \frac{\exp \phi(s, r, o, \tau)}{\sum_{s' \in \mathcal{E}} \exp \phi(s', r, o, \tau)}, \quad (8)$$

and the reverse inferring likelihood for query $(?, r, o, \tau)$ is similarly defined as $\mathcal{P}(s|o, r^{-1}, \tau)$, where $r^{-1}$ denotes the reverse relation of $r$. As a widely used strategy in TNTComplEx (Lacroix et al., 2020) and TCompoundE (Ying et al., 2024), we employ reciprocal learning for training TERRF. The loss function is defined as:

$$\begin{aligned}\mathcal{L}_u &= -\log \mathcal{P}(o|s, r, \tau) - \log \mathcal{P}(s|o, r^{-1}, \tau) \\ &+ \lambda_u(||\mathbf{e}_s||_3^3 + ||\mathbf{e}_s^\star||_3^3 + ||\mathbf{e}_o||_3^3)\end{aligned} \quad (9)$$

where the N3 regularization is applied to the original entity embeddings $\mathbf{e}_s$, $\mathbf{e}_o$, and the transformed entity embedding $\mathbf{e}_s^\star$; $\lambda_u$ is the weight coefficient. Following TNTComplEx (Lacroix et al., 2020), we employ a smoothing temporal regularizer to ensure that neighboring timestamps have similar representations, which is calculated as:

$$\mathcal{L}_\tau = \frac{1}{N_\tau - 1} \sum_{i=1}^{N_\tau - 1} ||\mathbf{e}_{\tau(i+1)} - \mathbf{e}_{\tau(i)}||_3^3 \quad (10)$$

The total loss function is defined as:

$$\mathcal{L} = \mathcal{L}_u + \lambda_\tau \mathcal{L}_\tau \quad (11)$$

where $\lambda_\tau$ represents the weight coefficient for regularizer.

| | ICEWS14 | ICEWS05-15 | GDELT |
|---|---|---|---|
| $\#\mathcal{E}$ | 7,128 | 10,488 | 500 |
| $\#\mathcal{R}$ | 230 | 251 | 20 |
| $\#\mathcal{T}$ | 365 | 4,017 | 366 |
| #Train | 72,826 | 386,962 | 2,735,685 |
| #Valid | 8,963 | 46,092 | 341,961 |
| #Test | 8,941 | 46,275 | 341,961 |

Table 1: Statistics of TKGC datasets in the experiment.

## 5 Experiments

### 5.1 Datasets

We evaluate the performance on three public TKG benchmark datasets: **ICEWS14**, **ICEWS05-15**, and **GDELT**. Among them, ICEWS14 and ICEWS05-15 are derived from *Integrated Crisis Early Warning System (ICEWS)* dataset (Lautenschlager et al., 2015). ICEWS is an event-based KG that contains political facts starting from 1995, with ICEWS14 focusing on events in 2014 and ICEWS05-15 covering events from 2005 to 2015. GDELT is a subset of the *Global Database of Events, Language, and Tone (GDELT)* (Leetaru and Schrodt, 2013). GDELT integrates information from diverse sources, encompassing factual entries with daily timestamps ranging from April 1, 2015, to March 31, 2016. Notably, GDELT focuses on the coverage to the 500 most common entities and the 20 most frequent relations. The summary of the datasets are listed in Table 1.

### 5.2 Evaluation Protocol

Given a test quadruple $(s, r, o, \tau)$, we predict the missing subject or object entity for the two queries $(s, r, ?, \tau)$ and $(?, r, o, \tau)$. During inference, we employ the time-aware filtered setting (Lacroix et al., 2020; Xu et al., 2020; Chen et al., 2022; Ying et al., 2024) to measure the performance in the TKGC task.

We adopt the widely used evaluation metrics, including Mean Reciprocal Rank (MRR) and Hits@N (Qiao and Hu, 2020). MRR is the average of the reciprocals of the ranks for all correct triples. The Hits@N is the proportion of the top N of all correct triples rankings. Notably, the higher values of MRR and Hits@N indicate better performances. We set N=1, 3, 10 for Hits@N in this experiment. For convenience, we denote Hits@N as H@N (N$\in \{1, 3, 10\}$) throughout this paper.

### 5.3 Experimental Setup

We use Python 3.8 and Pytorch framework to implement our model. All computations are done on a single NVIDIA GeForce RTX 2080 Ti GPU for the sake of fairness. We train our model using the Adagrad (Duchi et al., 2011) optimizer and choose the optimal hyperparameters using a grid search method according MRR on the validation set.

Finally, the optimal configurations of our model are as follows. The embedding dimensionality is set to 6000 for all the datasets. For the ICEWS14 dataset, the learning rate, batch size, and the max epoch are set to 0.01, 4000, and 400. We set the learning rate, batch size, and max epoch to 0.08, 6000, 100 in ICEWS05-15. Regarding GDELT, the learning rate, batch size, and the max epoch are set to 0.05, 2000, and 50. Other important optimal hyperparameters for TERRF are shown as follows:

- **ICEWS14**: $\lambda_u = 0.01$, $\lambda_\tau = 1$, $\alpha = 1$;
- **ICEWS05-15**: $\lambda_u = 0.05$, $\lambda_\tau = 1$, $\alpha = 0.1$;
- **GDELT**: $\lambda_u = 0.001$, $\lambda_\tau = 0.001$, $\alpha = 0.1$.

### 5.4 Baselines

We compare our model TERRF with traditional static KGE models and representative temporal KGE models with interpolation setting. The static KGE models include TransE (Bordes et al., 2013), DistMult (Yang et al., 2014), ComplEx (Trouillon et al., 2016), and SimplE (Kazemi and Poole, 2018). The temporal KGE models include TTransE (Leblay and Chekol, 2018), DE-SimplE (Goel et al., 2020), TA-DisMult (García-Durán et al., 2018), HyTE (Dasgupta et al., 2018), ChronoR (Sadeghian et al., 2021), TComplEx (Lacroix et al., 2020), TNTComplEx (Lacroix et al., 2020), TGAP (Jung et al., 2020), TeLM (Xu et al., 2021), BoxTE (Messner et al., 2022), TLT-KGE (Zhang et al., 2022), RotateQVS (Chen et al., 2022), TARGAT (Xie et al., 2023), TeAST (Li et al., 2023), and TCompoundE (Ying et al., 2024).

Notably, the TCompoundE model, which integrates translation and scaling operations is highly related to our work. Other recent temporal KGE models that represent TKGs in complex or hypercomplex space (*e.g.*, TeLM, RotateQVS, TLT-KGE, TeAST) are also associated with our TERRF, since the rotation learning is involved. Additionally, we also compare our approach with the models utilizing graph neighborhood information (*i.e.*, T-GAP, and TARGAT).

### 5.5 Main Results

Table 2 presents the link prediction results of our proposed TERRF and various baselines on three benchmarks. Generally, TERRF outperforms all the baselines on ICEWS14, ICEWS05-15, and GDELT across all the metrics. These results demonstrate the superiority of the TERRF model. Compared to traditional static KGE models, our model TERRF significantly surpasses all these baselines, which shows the usefulness of time information. Compared to the most related models, such as RotateQVS (Chen et al., 2022), TeAST (Li et al., 2023) and TCompoundE (Ying et al., 2024), TERRF obtains significant improvement gains. The reason is that our method achieves a more flexible spatial transformation, thereby modeling complex relational and temporal information in TKGs. Additionally, TERRF also obtains substantially better results than the methods integrating graph neighborhood information, such as T-GAP (Jung et al., 2020), and TARGET (Xie et al., 2023). It indicates that the rigid transformation of our method is indeed useful. Notably, we observe that TERRF outperforms the other baselines by large margins in GDELT dataset. Particularly, TERRF achieves relative improvements of 3.6%, 5.0%, 2.7%, 0.8% for MRR, H@1, H@3, and H@10 compared with TCompoundE. This is because our model constructs rotation operations using three elements: entity, relation, and timestamp, allowing spatial transformations to vary according to entity characteristics (see Section 6.3). To conclude, the experimental results across three datastes verify the ability of our proposed TERRF model.

## 6 Analysis

### 6.1 Ablation Study

In order to explore each component of our proposed TERRF, we conduct ablation study experiments. The results are reported in Table 3, where "w/o NS" denotes the model without using the Normalized Scaling operation, and "w/o RR" denotes the model without using the Rotation operation via Rodrigues' Rotation Formula. As shown in Table 3, we can see that the two ablation variants perform worse than the proposed TERRF model across all three datasets. The reason for this can be attributed to two factors: (1) Merely employing a rotation operation is insufficient due to the fixed distance to the origin of the coordinate system, whereas the Normalized Scaling operation provides a proper

| Models | ICEWS14 | | | | ICEWS05-15 | | | | GDELT | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 |
| TransE | 0.326 | 0.154 | 0.430 | 0.644 | 0.330 | 0.152 | 0.440 | 0.660 | 0.155 | 0.060 | 0.178 | 0.335 |
| DistMult | 0.441 | 0.325 | 0.498 | 0.668 | 0.457 | 0.338 | 0.515 | 0.691 | 0.210 | 0.133 | 0.224 | 0.365 |
| ComplEx | 0.442 | 0.400 | 0.430 | 0.664 | 0.464 | 0.347 | 0.524 | 0.696 | 0.213 | 0.133 | 0.225 | 0.366 |
| SimplE | 0.458 | 0.341 | 0.516 | 0.687 | 0.478 | 0.359 | 0.539 | 0.708 | 0.206 | 0.124 | 0.220 | 0.366 |
| TTransE | 0.255 | 0.074 | - | 0.601 | 0.271 | 0.084 | - | 0.616 | 0.115 | 0.0 | 0.160 | 0.318 |
| DE-SimplE | 0.526 | 0.418 | 0.592 | 0.725 | 0.513 | 0.392 | 0.578 | 0.748 | 0.230 | 0.141 | 0.248 | 0.403 |
| TA-DisMult | 0.477 | 0.363 | - | 0.686 | 0.474 | 0.346 | - | 0.728 | 0.206 | 0.124 | 0.219 | 0.365 |
| HyTE | 0.297 | 0.108 | 0.416 | 0.655 | 0.315 | 0.116 | 0.445 | 0.681 | 0.118 | 0.0 | 0.165 | 0.326 |
| ChronoR | 0.625 | 0.547 | 0.669 | 0.773 | 0.675 | 0.596 | 0.723 | 0.820 | - | - | - | - |
| TComplEX | 0.610 | 0.530 | 0.660 | 0.770 | 0.660 | 0.590 | 0.710 | 0.810 | 0.340 | 0.294 | 0.361 | 0.498 |
| TNTComplEx | 0.620 | 0.520 | 0.660 | 0.760 | 0.670 | 0.590 | 0.710 | 0.810 | 0.349 | 0.258 | 0.373 | 0.502 |
| T-GAP | 0.610 | 0.509 | 0.677 | 0.790 | 0.670 | 0.568 | 0.743 | 0.845 | - | - | - | - |
| TeLM | 0.625 | 0.545 | 0.673 | 0.774 | 0.678 | 0.599 | 0.728 | 0.823 | 0.350 | 0.261 | 0.375 | 0.504 |
| BoxTE | 0.613 | 0.528 | 0.664 | 0.763 | 0.667 | 0.582 | 0.719 | 0.820 | 0.352 | 0.269 | 0.377 | 0.511 |
| RotateQVS | 0.591 | 0.507 | 0.642 | 0.754 | 0.633 | 0.529 | 0.709 | 0.813 | 0.270 | 0.175 | 0.293 | 0.458 |
| TLT-KGE | 0.634 | 0.551 | 0.684 | 0.786 | 0.690 | 0.609 | 0.741 | 0.835 | 0.358 | 0.265 | 0.388 | 0.543 |
| TARGAT | 0.631 | 0.545 | 0.683 | 0.793 | 0.685 | 0.608 | 0.736 | 0.825 | - | - | - | - |
| TeAST | 0.637 | 0.560 | 0.682 | 0.782 | 0.683 | 0.604 | 0.732 | 0.829 | 0.371 | 0.283 | 0.401 | 0.544 |
| TCompoundE | 0.644 | 0.561 | 0.694 | 0.795 | 0.692 | 0.612 | 0.743 | 0.837 | 0.433 | 0.347 | 0.469 | 0.595 |
| TERRF (Ours) | **0.650** | **0.570** | **0.698** | **0.800** | **0.701** | **0.621** | **0.752** | **0.847** | **0.469** | **0.397** | **0.496** | **0.603** |

Table 2: Summary of results on three datasets, namely ICEWS14, ICEWS05-15, and GDELT. The best score is in **bold** and the second best is underlined.

| Models | ICEWS14 | | | | ICEWS05-15 | | | | GDELT | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 | MRR | H@1 | H@3 | H@10 |
| TERRF | **0.650** | **0.570** | **0.698** | **0.800** | **0.701** | **0.621** | **0.752** | **0.847** | **0.469** | **0.397** | **0.496** | **0.631** |
| w/o NS | 0.643 | 0.561 | 0.693 | 0.793 | 0.698 | 0.618 | 0.749 | 0.845 | 0.453 | 0.378 | 0.482 | 0.593 |
| w/o RR | 0.621 | 0.536 | 0.671 | 0.782 | 0.675 | 0.588 | 0.731 | 0.835 | 0.416 | 0.324 | 0.453 | 0.592 |

Table 3: Ablation study on three datasets, , namely ICEWS14, ICEWS05-15, and GDELT. The best score is in **bold** and the second best is underlined.

initial position. (2) Utilizing our designed rotation strategy via Rodrigues' Rotation Formula allows for more flexible spatial transformations, thus enabling the modeling of more complex relational and temporal patterns. Furthermore, we observe that individually employing rotation (w/o NS) consistently outperforms the variant that only utilizes Normalized Scaling (w/o RR). This demonstrates that our rotation learning strategy plays a more crucial role in link prediction. The experimental results of the ablation study confirm that both the United Scaling and the Rotation via Rodrigues' Rotation Formula contribute to improving the performance of TERRF.

| Normalization | ICEWS14 | | | |
|---|---|---|---|---|
| | MRR | H@1 | H@3 | H@10 |
| - | 0.641 | 0.561 | 0.688 | 0.790 |
| Min-Max | 0.644 | 0.563 | 0.692 | 0.792 |
| Softmax-Weight | 0.646 | 0.565 | 0.693 | 0.795 |
| Z-Score | 0.648 | 0.567 | 0.693 | 0.798 |
| Vector-Norm (Ours) | **0.650** | **0.570** | **0.698** | **0.800** |
| $\chi = 0.1$ | 0.645 | 0.567 | 0.692 | 0.792 |
| $\chi = 1$ | 0.641 | 0.561 | 0.688 | 0.790 |
| $\chi = 10$ | 0.640 | 0.559 | 0.688 | 0.791 |

Table 4: Results on different normalizations for scaling.

### 6.2 Impact of the Normalization for Scaling

In this section, we investigate the impact of vector normalization in the Normalized Scaling. We compare our strategy (*i.e.*, convert the scaling factor to a unit vector) with several variants, including one without normalization, and others with different normalization methods: Min-Max normalization, Softmax weighted strategy, and Z-Score technique

| M# | Elements | | | GDELT | | | |
|---|---|---|---|---|---|---|---|
| | $s$ | $r$ | $\tau$ | **MRR** | **H@1** | **H@3** | **H@10** |
| M1 | - | - | - | 0.416 | 0.324 | 0.453 | 0.592 |
| M2 | | ✓ | ✓ | 0.411 | 0.319 | 0.447 | 0.589 |
| M3 | ✓ | ✓ | | 0.415 | 0.324 | 0.451 | 0.591 |
| M4 | ✓ | | ✓ | 0.460 | 0.380 | 0.494 | **0.609** |
| M5 | ✓ | ✓ | ✓ | **0.469** | **0.397** | **0.496** | 0.603 |

Table 5: Results on GDELT dataset in terms of different constructions of axis and angle of rotation. M1 stands for the model without using rotation learning.

| M# | Elements | | | ICEWS05-15 | | | |
|---|---|---|---|---|---|---|---|
| | $s$ | $r$ | $\tau$ | **MRR** | **H@1** | **H@3** | **H@10** |
| M1 | - | - | - | 0.698 | 0.618 | 0.749 | 0.845 |
| M2 | | ✓ | ✓ | 0.694 | 0.612 | 0.746 | 0.844 |
| M3 | ✓ | ✓ | | 0.693 | 0.609 | 0.746 | 0.845 |
| M4 | ✓ | | ✓ | **0.701** | 0.620 | **0.753** | **0.848** |
| M5 | ✓ | ✓ | ✓ | **0.701** | **0.621** | 0.752 | 0.847 |

Table 6: Results on ICEWS05-15 dataset in terms of different constructions of axis and angle of rotation. M1 stands for the model without using rotation learning.

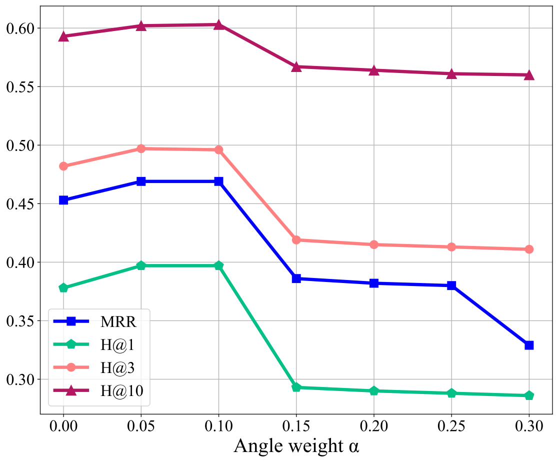



Figure 4: Results on GDELT dataset regarding the different angle weight $\alpha$.

(as shown in Table 4). From the results, we find that using normalization obtains more performance gains compared with the one without normalization, where our proposed Vector-Norm (regards the scaling factor as a unit vector) performs the best. We speculate that this is due to the use of appropriate normalization operations, which helps prevent overfitting.

Additionally, there is also a possibility that the numerical size of the scaling factor affects the results. Therefore, we replace the denominator $||h(\mathbf{e}_r + \mathbf{e}_\tau)^{[3i:\,3i+3]}||$ in Eq. 3 with a constant $\chi$. We conduct experiment to investigate the impact of changes in $\chi$ on the TKGC performance. From Table 4, we observe that changes in $\chi$ have a minimal impact on the performance of link prediction, which indirectly supports our above conclusion.

### 6.3 Analysis of the Rotation Learning

To explore aspects of rotation learning, we conduct experiments on the construction of the axis and angle in Rodrigues' Rotation Formula. We remove different elements (*i.e.*, $s$, $r$, $\tau$) to redefine the axis vector $\mathbf{k}$ and angle $\theta$. These model variants are compared in Table 5 and Table 6. From the results, we observe that combining all the elements consistently and significantly enhances the performance (M5). Additionally, we find that the performance varies when any two elements from $s$, $r$, and $\tau$ are combined together (M2-4). Notably, simultaneously combining $s$ and $\tau$ is the primary reason for the significant performance improvements (M4 vs. M2-3). Nevertheless, it verifies that integrating information of entity, relation and time to construct axis and angle is more stable and effective.

Moreover, we also analyze the effect of the angle weight $\alpha$, which is shown in Figure 4. The results show that when $\alpha = 0.05$ or $0.1$, TERFF obtains the remarkable performances across all the metrics, while the further increasing of $\alpha$ leads to worse results (*i.e.*, $\alpha \in \{0.15, 0.2, 0.25, 0.3\}$). Nevertheless, our TERRF beats the one removing rotation operation ($\alpha = 0$). This indicates that a suitable angle weight can enhance the rotation learning.

## 7 Conclusion

This paper proposes to learn embeddings via a 3D rigid body transformation for TKGC task. On the one hand, we utilize a Normalized Scaling operation to set an initial position for entity to provide a more flexible range of rotations. On the other hand, we propose a efficient rotation strategy via Rodrigues' Rotation Formula, which merely requires an axis and angle representation. Empirical results show that our proposed TERRF achieves the promising results and outperforms state-of-the-art models. Furthermore, we have mathematically derived that TERRF can model multiple important temporal relation patterns.

## Limitations

Despite our method enabling more efficient implementation of rotation operations and achieving significant results, the current application of Rodrigues' Rotation Formula is limited to rotations in three-dimensional space. Efficient spatial transformations in higher dimensions remain a significant challenge. Additionally, while our method partially avoids the dimension expansion of entities and relations, which would lead to a total parameter count highly dependent on the number of entities and relations, the parameter count of our method still largely depends on the three linear transformation matrices used to convert scaling factors, direction axes, and angles, as determined by the embedding dimension. In the future, we will continually solve these limitations for a better temporal knowledge graph embedding solution.

## References


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

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Kai Chen, Ye Wang, Yitong Li, and Aiping Li. 2022. RotateQVS: Representing temporal information as rotations in quaternion vector space for temporal knowledge graph completion. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5843–5857, Dublin, Ireland. Association for Computational Linguistics.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. HyTE: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2001–2011, Brussels, Belgium. Association for Computational Linguistics.

Yao Dong, Qingchao Kong, Lei Wang, and Yin Luo. 2024. Dual complex number knowledge graph embeddings. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5391–5400, Torino, Italia. ELRA and ICCL.

Yao Dong, Lei Wang, Ji Xiang, Xiaobo Guo, and Yuqiang Xie. 2022. RotateCT: Knowledge graph embedding by rotation and coordinate transformation in complex space. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4918–4932, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. 2014. Introducing wikidata to the linked data web. In *The Semantic Web–ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I 13*, pages 50–65. Springer.

M Fabian, Kasneci Gjergji, WEIKUM Gerhard, et al. 2007. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In *16th International world wide web conference, WWW*, pages 697–706.

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Conference on Empirical Methods in Natural Language Processing*.

Xiou Ge, Yun-Cheng Wang, Bin Wang, and C-C Jay Kuo. 2022. Compounde: Knowledge graph embedding with translation, rotation and scaling compound operations. *arXiv preprint arXiv:2207.05324*.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3988–3995.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 687–696, Beijing, China. Association for Computational Linguistics.

Jaehun Jung, Jinhong Jung, and U Kang. 2020. T-gap: Learning to walk across time for temporal knowledge graph completion. *ArXiv*, abs/2012.10595.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. *Advances in neural information processing systems*, 31.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. *ArXiv*, abs/2004.04926.

Jennifer Lautenschlager, Steve Shellman, and Michael Ward. 2015. ICEWS Event Aggregations.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. *Companion Proceedings of the The Web Conference 2018*.

Kalev Leetaru and Philip A. Schrodt. 2013. Gdelt: Global data on events, location, and tone. *ISA Annual Convention*.

Jiang Li, Xiangdong Su, and Guanglai Gao. 2023. TeAST: Temporal knowledge graph embedding via archimedean spiral timeline. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15460–15474, Toronto, Canada. Association for Computational Linguistics.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

Johannes Messner, Ralph Abboud, and Ismail Ilkan Ceylan. 2022. Temporal knowledge graph completion using box embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7779–7787.

Chen Qiao and Xiao Hu. 2020. A neural knowledge graph evaluator: Combining structural and semantic evidence of knowledge graphs for predicting supportive knowledge in scientific qa. *Inf. Process. Manag.*, 57:102309.

Ali Sadeghian, Mohammadreza Armandpour, Anthony Colas, and Daisy Zhe Wang. 2021. Chronor: Rotation based temporal knowledge graph embedding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6471–6479.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. 2014. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yun Tang, Jing Huang, Guangtao Wang, Xiaodong He, and Bowen Zhou. 2020. Orthogonal relation transforms with graph context modeling for knowledge graph embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2713–2722, Online. Association for Computational Linguistics.

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

Kun Wang and Jian S Dai. 2023. The dual euler-rodrigues formula in various mathematical forms and their intrinsic relations. *Mechanism and Machine Theory*, 181:105184.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28.

Zhihao Wang and Xin Li. 2019. Hybrid-te: Hybrid translation-based temporal knowledge graph embedding. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1446–1451.

Azmine Toushik Wasi. 2024. HRGraph: Leveraging LLMs for HR data knowledge graphs with information propagation-based job recommendation. In *Proceedings of the 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024)*, pages 56–62, Bangkok, Thailand. Association for Computational Linguistics.

Zhiwen Xie, Runjie Zhu, Jin Liu, Guangyou Zhou, and Jimmy Xiangji Huang. 2023. Targat: A time-aware relational graph attention model for temporal knowledge graph embedding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2246–2258.

Chengjin Xu, Yung-Yu Chen, Mojtaba Nayyeri, and Jens Lehmann. 2021. Temporal knowledge graph completion using a linear temporal regularizer and multivector embeddings. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2569–2578, Online. Association for Computational Linguistics.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020. TeRo: A time-aware knowledge graph embedding via temporal rotation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1583–1593, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations*.

Rui Ying, Mengting Hu, Jianfeng Wu, Yalan Xie, Xiaoyi Liu, Zhunheng Wang, Ming Jiang, Hang Gao, Linlin Zhang, and Renhong Cheng. 2024. Simple but effective compound geometric operations for temporal knowledge graph completion. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11074–11086, Bangkok, Thailand. Association for Computational Linguistics.

Fuwei Zhang, Zhao Zhang, Xiang Ao, Fuzhen Zhuang, Yongjun Xu, and Qing He. 2022. Along the time:

Timeline-traced embedding for temporal knowledge graph completion. CIKM '22, page 2529–2538, New York, NY, USA. Association for Computing Machinery.

Geng Zhang, Jin Liu, Guangyou Zhou, Kunsong Zhao, Zhiwen Xie, and Bo Huang. 2024. Question-directed reasoning with relation-aware graph attention network for complex question answering over knowledge graph. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:1915–1927.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. *Advances in neural information processing systems*, 32.

Caleb Ziems, William Held, Jane Dwivedi-Yu, and Diyi Yang. 2024. Measuring and addressing indexical bias in information retrieval. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12860–12877, Bangkok, Thailand. Association for Computational Linguistics.

## A Proof of Relation Pattern Modeling

In this section we provide the proofs that our TERRF can model various relational patterns

### A.1 Definition of Relation Patterns

**Definition 1**: Relation $r$ is symmetric, if $\forall s, o, \tau$, $(s, r, o, \tau) \wedge (o, r, s, \tau) \in \mathcal{G}$;

**Definition 2**: Relation $r$ is asymmetric, if $\forall s, o, \tau$, $(s, r, o, \tau) \in \mathcal{G} \wedge (o, r, s, \tau) \notin \mathcal{G}$;

**Definition 3**: Relation $r_1$ and $r_2$ are asymmetric, if $\forall s, o, \tau$, $(s, r_1, o, \tau) \wedge (o, r_2, s, \tau) \in \mathcal{G}$;

**Definition 4**: Relation $r_1$ and $r_2$ are evolving over time from timestamp $\tau_1$ to timestamp $\tau_2$, if $\forall s, o, \tau$, $(s, r_1, o, \tau_1) \wedge (o, r_2, s, \tau_2) \in \mathcal{G}$;

**Definition 5**: Relation $r$ reflects the many-to-one pattern when $s_1$ and $s_2$ are simultaneously associated with $o$ at the timestamp $\tau$, *i.e.*, $(s_1, r, o, \tau) \wedge (s_2, r, o, \tau) \in \mathcal{G}$.

### A.2 Matrix Notation for Rodrigues' Rotation Formula

$\forall \mathbf{v} \in \mathbb{R}^3$, the rotated vector $\mathbf{v}_{rot}$ using Rodrigues' Rotation Formula can be expressed in matrix form as:

$$\begin{aligned} \mathbf{v}_{rot} &= Rot(\mathbf{v}, \mathbf{k}, \theta) \\ &= \mathbf{R}\mathbf{v}, \end{aligned} \tag{12}$$

where

$$\mathbf{R} = \mathbf{I} + (\sin\theta)\mathbf{K} + (1 - \cos\theta)\mathbf{K}^2, \tag{13}$$

and $\mathbf{I} \in \mathbb{R}^{3\times 3}$ is the identity matrix, while $\mathbf{K} \in \mathbb{R}^{3\times 3}$ denotes the skew-symmetric matrix generated by the rotation vector $\mathbf{k}$.

Therefore, Eq. 6 can be rewritten as:

$$\begin{aligned} &\mathbf{e}_s^{\star[3i:\,3i+3]} = Rot(\tilde{\mathbf{e}_s}^{[3i:\,3i+3]}, \mathbf{k}^{[3i:\,3i+3]}, \theta_i) \\ &= \mathbf{R}_{s,r,\tau}^{(i)} \tilde{\mathbf{e}_s}^{[3i:\,3i+3]} \\ &= \mathbf{R}_{s,r,\tau}^{(i)} \left( \frac{h(\mathbf{e}_{r,\tau})^{[3i:\,3i+3]}}{||h(\mathbf{e}_{r,\tau})^{[3i:\,3i+3]}||} \circ \mathbf{e}_s^{[3i:\,3i+3]} \right) \\ &= \mathbf{R}_{s,r,\tau}^{(i)} \left( \mathbf{s}_{r,\tau}^{(i)} \circ \mathbf{e}_s^{[3i:\,3i+3]} \right) \\ &= \left( \mathbf{R}_{s,r,\tau}^{(i)} \circ \mathbf{s}_{r,\tau}^{(i)} \right) \mathbf{e}_s^{[3i:\,3i+3]} \\ &= \mathbf{M}_{s,r,\tau}^{(i)} \mathbf{e}_s^{[3i:\,3i+3]} \end{aligned} \tag{14}$$

where $\mathbf{M}_{s,r,\tau}^{(i)} \in \mathbb{R}^{3\times 3}$ represents a matrix that contains all the spatial transformation operations in the TERRF model. To facilitate subsequent expressions, we uniformly use $\mathbf{e}_s$, $\mathbf{M}_{s,r,\tau}$ and $\mathbf{e}_o$ to represent $\mathbf{e}_s^{[3i:\,3i+3]}$, $\mathbf{M}_{s,r,\tau}^{[3i:\,3i+3]}$ and $\mathbf{e}_o^{[3i:\,3i+3]}$, respectively.

### A.3 Symmetric Pattern

For the symmetric pattern, we have $\phi(s, r, o, \tau) = \phi(o, r, s, \tau)$, and we derive:

$$\begin{aligned} &\mathbf{M}_{s,r,\tau}\mathbf{e}_s{\mathbf{e}_o}^{\mathrm{T}} = \mathbf{M}_{o,r,\tau}\mathbf{e}_o{\mathbf{e}_s}^{\mathrm{T}} \\ &\Rightarrow \mathbf{M}_{o,r,\tau}^{-1}\mathbf{M}_{s,r,\tau} = \mathbf{I} \;\; or \;\; \mathbf{M}_{s,r,\tau}^{-1}\mathbf{M}_{o,r,\tau} = \mathbf{I}. \end{aligned} \tag{15}$$

This demonstrates that the TERRF model can capture the symmetric pattern when $\mathbf{M}_{s,r,\tau}$ and $\mathbf{M}_{o,r,\tau}$ are inverse matrices.

### A.4 Asymmetric Pattern

For the asymmetric pattern, we have $\phi(s, r, o, \tau) \neq \phi(o, r, s, \tau)$. According to the proof in Appendix A.3, the TERRF model can capture the asymmetric pattern when $\mathbf{M}_{s,r,\tau}$ and $\mathbf{M}_{o,r,\tau}$ are not inverse matrices.

### A.5 Inverse Pattern

In the inverse pattern, $\phi(s, r_1, o, \tau) = \phi(o, r_2, s, \tau)$ holds, leading to:

$$\begin{aligned} &\mathbf{M}_{s,r_1,\tau}\mathbf{e}_s{\mathbf{e}_o}^{\mathrm{T}} = \mathbf{M}_{s,r_2,\tau}\mathbf{e}_s{\mathbf{e}_o}^{\mathrm{T}} \\ &\Rightarrow \mathbf{M}_{s,r_2,\tau}^{-1}\mathbf{M}_{s,r_1,\tau} = \mathbf{I} \text{ or } \mathbf{M}_{s,r_1,\tau}^{-1}\mathbf{M}_{s,r_2,\tau} = \mathbf{I}. \end{aligned} \tag{16}$$

This shows that the TERRF model can capture the inverse pattern when $\mathbf{M}_{s,r_1,\tau}$ and $\mathbf{M}_{s,r_2,\tau}$ are inverse matrices.

### A.6 Temporal Evolution Pattern

For the temporal evolution pattern, we have $\phi(s, r_1, o, \tau_1) = \phi(s, r_2, o, \tau_2)$, resulting in:

$$\begin{aligned}&\mathbf{M}_{s,r_1,\tau_1}\mathbf{e}_s\mathbf{e}_o{}^{\mathrm{T}} = \mathbf{M}_{s,r_2,\tau_2}\mathbf{e}_s\mathbf{e}_o{}^{\mathrm{T}}\\ &\Rightarrow \mathbf{M}^{-1}_{s,r_2,\tau_2}\mathbf{M}_{s,r_1,\tau_2} = \mathbf{I} \text{ or } \mathbf{M}^{-1}_{s,r_1,\tau_1}\mathbf{M}_{s,r_2,\tau_2} = \mathbf{I}.\end{aligned} \tag{17}$$

This indicates that the TERRF model can capture the temporal evolution pattern when $\mathbf{M}_{s,r_1,\tau_1}$ and $\mathbf{M}_{s,r_2,\tau_2}$ are inverse matrices.

### A.7 Many-to-One Pattern

In this scenario, we have $\phi(s_1, r, o, \tau) = \phi(s_2, r, o, \tau)$, as illustrated in Figure 3.

First, let us investigate the case where entity information is not incorporated into the spatial transformation construction of TERRF. Specifically, we replace $\mathbf{M}_{s,r,\tau}$ with $\mathbf{M}_{r,\tau}$, which corresponds to M7 in Table 5. By definition, we derive:

$$\begin{aligned}&\mathbf{M}_{r,\tau}\mathbf{e}_{s_1}\mathbf{e}_o{}^{\mathrm{T}} = \mathbf{M}_{r,\tau}\mathbf{e}_{s_2}\mathbf{e}_o{}^{\mathrm{T}}\\ &\Rightarrow \mathbf{M}^{-1}_{r,\tau}\mathbf{M}_{r,\tau}\mathbf{e}_{s_1}\mathbf{e}_o{}^{\mathrm{T}} = \mathbf{e}_{s_1}\mathbf{e}_o{}^{\mathrm{T}} = \mathbf{e}_{s_2}\mathbf{e}_o{}^{\mathrm{T}}\\ &\Rightarrow \mathbf{e}_{s_1} = \mathbf{e}_{s_2}\end{aligned} \tag{18}$$

As shown above, when only using the relation and timestamp for transformation (*i.e.*, $\mathbf{M}_{r,\tau}$), the embeddings of $s_1$ and $s_2$ will converge to the same value. However, in many cases, there are significant semantic differences between $s_1$ and $s_2$. Thus, this approach cannot adequately model the many-to-one pattern.

When using our strategy (*i.e.*, $\mathbf{M}_{s,r,\tau}$), we have:

$$\begin{aligned}&\mathbf{M}_{s_1,r,\tau}\mathbf{e}_{s_1}\mathbf{e}_o{}^{\mathrm{T}} = \mathbf{M}_{s_2,r,\tau}\mathbf{e}_{s_2}\mathbf{e}_o{}^{\mathrm{T}}\\ &\Rightarrow \mathbf{M}_{s_1,r,\tau}\mathbf{e}_{s_1} = \mathbf{M}_{s_2,r,\tau}\mathbf{e}_{s_2}\\ &\Rightarrow \mathbf{e}_{s_1} = \mathbf{M}^{-1}_{s_1,r,\tau}\mathbf{M}_{s_2,r,\tau}\mathbf{e}_{s_2}.\end{aligned} \tag{19}$$

Therefore, we observe that the TERRF model can capture the many-to-one pattern when $\mathbf{e}_{s_1} = \mathbf{M}^{-1}_{s_1,r,\tau}\mathbf{M}_{s_2,r,\tau}\mathbf{e}_{s_2}$.