mathematics

MDPI

Article

Revolutionary Strategy for Depicting Knowledge Graphs with Temporal Attributes

Sihan Li [1,2,†] and Qi Li [1,*,†]

[1] Computer Science and Engineering Department, Shaoxing University, Shaoxing 312000, China
[2] Faculty of Arts & Science, University of Toronto, Toronto, ON M5S 3G3, Canada
[*] Correspondence: liqi0713@usx.edu.cn
[†] These authors contributed equally to this work.

Abstract: In practical applications, the temporal completeness of knowledge graphs is of great importance. However, previous studies have mostly focused on static knowledge graphs, generally neglecting the dynamic evolutionary properties of facts. Moreover, the unpredictable and limited availability of temporal knowledge graphs, together with the complex temporal dependency patterns, make current models inadequate for effectively describing facts that experience temporal transitions. To better represent the evolution of things over time, we provide a learning technique that uses quaternion rotation to describe temporal knowledge graphs. This technique describes the evolution of entities as a temporal rotation change in quaternion space. Compared to the Ermitian inner product in complex number space, the Hamiltonian product in quaternion space is better at showing how things might be connected. This leads to a learning process that is both more effective and more articulate. Experimental results demonstrate that our learning method significantly outperforms existing methods in capturing the dynamic evolution of temporal knowledge graphs, with improved accuracy and robustness across a range of benchmark datasets.

Keywords: temporal attributes; knowledge graph; quaternion rotation; representation learning

MSC: 05C85

check for updates

Citation: Li, S.; Li, Q. Revolutionary Strategy for Depicting Knowledge Graphs with Temporal Attributes. *Mathematics* **2024**, *12*, 1324. https://doi.org/10.3390/math12091324

Academic Editor: Jianping Gou

Received: 18 March 2024
Revised: 19 April 2024
Accepted: 22 April 2024
Published: 26 April 2024

1. Introduction

Knowledge graph technology is crucial in the domain of knowledge representation and reasoning. A knowledge graph is a comprehensive semantic network composed of nodes and edges [1–3]. It may be conceptualized as a data structure based on a graph. A knowledge graph is often shown as a triplet, including a head entity, a tail entity, and a connection between the two entities. Well-known knowledge graphs like FreeBase, YAGO, and Schema.org serve as typical instances of knowledge graphs [4]. A knowledge graph may efficiently communicate the relationships between entities via a structured framework, providing the advantages of scalability and understandability. It is helpful to use the knowledge graph for things like system recommendations and obtaining information.

The knowledge graph remains incomplete and suffers from a significant data sparsity problem, meaning that many facts in the knowledge graph lack associated links or entities. The absence of this information can limit the effectiveness of the knowledge graph and its broader applications, highlighting the need for knowledge graph completion. Knowledge graph completion, also known as link prediction, is a type of knowledge reasoning that attempts to predict missing information by using current facts within the knowledge graph [5]. Initially, the knowledge graph was mostly developed through artificial creation. However, this approach is wasteful and difficult to handle when dealing with a knowledge graph of a very large size. Knowledge representation learning uses low-dimensional vectorisation to represent entities and relationships [1,2,4], effectively addressing the problem of sparse data. At present, knowledge representation learning provides the basis for building and using large-scale knowledge graphs, and it is also the most widely used technique for completing knowledge graphs.

However, most of the existing research focuses on the static knowledge network, which poses challenges in understanding the temporal fluctuations of information. In fact, some pieces of information change over time. For example, the set of three facts constituting the term of office of South Korean President Moon Jae-in was strictly limited to the period from May 2017 to May 2021. Nevertheless, there have been cases where clear triads have been formed, such as South Korea, President Park Geun-hye, and South Korea, President Lee Myung-bak. Ignoring the temporal aspect when answering the question "Who is the President of South Korea?" tends to lead to incorrect results. Therefore, this example demonstrates that including temporal information in knowledge graphs can significantly enhance the understanding and prediction of dynamic relationships within the data.

This study introduces a learning technique called TKGR which is used to model temporal knowledge graphs. TKGR uses quaternion rotation to accurately represent the evolution of things over time. The concept describes the process of changing entities over time as a rotational motion in quaternion space. The Hamiltonian product in quaternion space provides a more efficient and expressive learning process than the Ermitian inner product in complex number space. This allows for a more effective capture of possible interdependencies. The main contributions of this work are listed below:

- To optimize the use of temporal data and to adequately represent the evolution of entities over time, we propose to use a learning method that uses quaternion rotation to describe temporal knowledge graphs. The model uses a rotation transformation in quaternion space to represent the temporal evolution of entities. This approach improves both the efficiency and the expressiveness of the learning process.
- Our approach uses Hamiltonian product-based quaternion rotations to accurately describe the temporal and relational changes in knowledge networks. By exploiting the unique properties of quaternions, we are able to outperform the Ermitian product in complex numbers when it comes to representing relational data.
- Finally, we verify the efficiency and suitability of the TKGR model through thorough experimental comparisons with state-of-the-art algorithms from different perspectives.

2. Related Work

The transformation of entities and interactions within a knowledge network into low-dimensional continuous vectors is a function that can be performed using embedding methods [6,7]. Static knowledge graph embedding and temporal knowledge graph embedding are the topics explicitly covered in this section.

2.1. Static Knowledge Graph Embedding

Researchers have identified a variety of knowledge graph embedding strategies for static knowledge graphs from their study. Knowledge graph embedding based on matrix decomposition, knowledge graph embedding based on convolutional neural networks, and knowledge graph embedding based on graphs are the three main categories that can be used to classify the approaches shown here.

When first presented, TransE [8] was used to solve models of knowledge graph embedding. These models use a score function to map entities h, t and relations r: $h + r \approx t$. The aim is to reduce the distance between the two points, h and t. However, TransE has shortcomings when it comes to its ability to manage complicated relationships. It performs very well only in the case of one-to-one relations; it is unable to handle one-to-N and N-to-one relations. It has been suggested that many extended models can be used to get around these limitations. TransH [9] is an example of such a model. It is characterized by the introduction of a hyperplane representing each relation onto which entities are projected. Because of this, TransH is able to manipulate relations between one and N as well as from one to N. When the entities are transformed into a new vector space, another model is used, called TransR [9]. This model uses a relation-specific matrix that is different from the others. Because of this, TransR is able to efficiently manage many different types of relations. Another extended model that overcomes the limitations of TransE is a model called TransD [10]. Within this framework, additional matrices are introduced to uniquely describe the semantic meaning of elements and relations. As a result, TransD is able to manage complicated interactions and capture information with finer granularity.

Matrix factorization-based models such as DistMult [11], RESCAL [12], HolE [13], ComplEx [14] and SimplE [15] are responsible for transforming relations into linear transformations of entity embeddings imposed on matrices. ConvE [16] and ConvKB [17] are the two most commonly used in CNN-based models. In addition to transforming entities and relations into multidimensional matrices, ConvE and ConvKB also describe the interactions that occur between entities and relations through the use of convolution and build embeddings that are more expressive through the process of feature learning. Graph-based techniques are the subject of our final discussion. DeepWalk [18] is the first method to create vector embeddings of nodes in a network using a random walk model. This method is a weightless random walk technique. The R-GCN [19] is a graph neural network extension that is considered to be one of the first methods to use graph neural networks to learn node embeddings. This is achieved by aggregating the neighbourhood information of each entity ′, which ultimately results in identical weights for all entities. However, these approaches are not ideal for temporal knowledge graphs because they are unable to capture the temporal information contained within temporal knowledge graphs.

2.2. Temporal Knowledge Graph Embedding

Improving static knowledge graph approaches by incorporating temporal aspects has been the focus of previous research initiatives. Through the use of temporal constraints, an important model known as TTransE [15] was the first to pioneer the learning of entity and relation embeddings. This was achieved by expressing transitions between temporal relations. The sequential relationship between the events "diedIn" and "wasBornIn" is an example of what is considered. Merging temporal information from static models is not a simple extension of the models described above. To incorporate temporal knowledge graphs, the ConT model is an extension of the static knowledge graph model [20] for further explanation. This allows for the model to store the graph data and extrapolate to incorporate more recent information. By incorporating time into the entity relationship structure, HyTE [21] establishes a direct and transparent link between each timestamp and the hyperplane that is relevant to the entity. HyTE not only makes predictions about the temporal range of relational facts that do not contain time annotations, but also uses temporal guidance to infer knowledge graphs. The challenge of predicting the temporal knowledge graph is addressed by TA-DisMult [22], which does this by capturing the encoding of prospective entities and connection types. To capture time-sensitive representations of relationship types, the model uses recurrent neural networks. This allows for the integration of these representations with existing potential factor decomposition methods used for temporal information fusion.

DE-SimplE [23] is an extension of a static knowledge graph model that incorporates a diachronic entity embedding function. This function allows for the representation of entity properties at any point in time. This is different from current approaches to temporal knowledge graph embedding which only provide static properties of entities. The assignment of labels to entities, relationships, timestamps, and locations is performed by SubEE [24] using a vocabulary of a given size. The model, on the other hand, uses a spatio-temporal messaging layer to capture the latent feature vectors of the knowledge network. BoxTE [25] is an extended version of the static knowledge graph embedding model. It demonstrates a high level of expressiveness and a strong aptitude for inductive reasoning in temporal contexts.

While the approaches currently available for completing temporal knowledge graphs are promising, they often fail to capture both the structure of the graph and the temporal connections. Compared to previous techniques, which often focus on either entity properties or connections without integration, TKGR is able to capture both temporal dynamics and graph structure in a unique way. As a result, it produces embeddings that are more accurate for the tasks at hand.

3. Theory of Quaternions

This section provides a preliminary overview of the theoretical foundations of quaternions. The quaternion system $\mathbb{H}$ is an extension of the complex number system and typically consists of one real and three imaginary components, as specified below.

$$q = q_r + q_i\boldsymbol{i} + q_j\boldsymbol{j} + q_k\boldsymbol{k} \tag{1}$$

where q_r, q_i, q_j, and q_k are the corresponding real coefficients. $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ are the imaginary units. If $q_j = q_k = 0$, it is the conventional complex form. These imaginary units also satisfy the following set of rules:

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1 \tag{2}$$

This rule may be used to infer a few more rules that do not correspond to the concept of commutativity, such as $\boldsymbol{ij} = \boldsymbol{k}$, $\boldsymbol{ji} = -\boldsymbol{k}$, $\boldsymbol{jk} = \boldsymbol{i}$, $\boldsymbol{ki} = \boldsymbol{j}$, $\boldsymbol{kj} = -\boldsymbol{i}$, and $\boldsymbol{ik} = -\boldsymbol{j}$. A quaternion vector $\boldsymbol{q} \in \mathbb{H}^n$ is defined as follows:

$$\boldsymbol{q} = \boldsymbol{q}_r + \boldsymbol{q}_i\boldsymbol{i} + \boldsymbol{q}_j\boldsymbol{j} + \boldsymbol{q}_k\boldsymbol{k} \tag{3}$$

where $\boldsymbol{q}_r$, $\boldsymbol{q}_i$, $\boldsymbol{q}_j$ and $\boldsymbol{q}_k$ are n-dimensional real vectors. For the purpose of modeling relational data, this study makes use of these concepts. According to the following definitions, there are a few standard arithmetic rules for quaternions:

Conjugate: The conjugate representation of quaternion $q \in \mathbb{H}$ is defined as follows:

$$\bar{q} = q_r - q_i\boldsymbol{i} - q_j\boldsymbol{j} - q_k\boldsymbol{k} \tag{4}$$

Norm: The norm of quaternion $\boldsymbol{q} \in \mathbb{H}$ is defined as follows:

$$\|q\| = \sqrt{q_r^2 + q_i^2 + q_j^2 + q_k^2} \tag{5}$$

By means of this operation, the definition of the quaternion of the unit $q^{\triangleleft}$ is obtained.

$$q^{\triangleleft} = \frac{q}{\|q\|} \tag{6}$$

The unit representation of the quaternion vector $\boldsymbol{q} \in \mathbb{H}^n$ is defined below.

$$\boldsymbol{q}^{\triangleleft} = \frac{\boldsymbol{q}_r + \boldsymbol{q}_i\boldsymbol{i} + \boldsymbol{q}_j\boldsymbol{j} + \boldsymbol{q}_k\boldsymbol{k}}{\sqrt{\boldsymbol{q}_r^2 + \boldsymbol{q}_i^2 + \boldsymbol{q}_j^2 + \boldsymbol{q}_k^2}} \tag{7}$$

Addition: The addition operation of two quaternions $q_1 = q_{r1} + q_{r1}\boldsymbol{i} + q_{j1}\boldsymbol{j} + q_{k1}\boldsymbol{k}$ and $q_2 = q_{r1} + q_{r2}\boldsymbol{i} + q_{j2}\boldsymbol{j} + q_{k2}\boldsymbol{k}$ is defined as follows:

$$\begin{aligned} q_1 + q_2 = {} & (q_{r1} + q_{r2}) + (q_{i1} + q_{i2})\boldsymbol{i} \\ & + (q_{j1} + q_{j2})\boldsymbol{j} + (q_{k1} + q_{k2})\boldsymbol{k} \end{aligned} \tag{8}$$

Scalar multiplication: The product of a scalar λ and a quaternion $q \in \mathbb{H}$ is defined as below.

$$\lambda q = \lambda q_r + \lambda q_i\boldsymbol{i} + \lambda q_j\boldsymbol{j} + \lambda q_k\boldsymbol{k} \tag{9}$$

Inner product: Similar to the inner product operation on vectors, the result of the inner product operation on quaternions can be obtained by computing the product of the corresponding components of two quaternions $q_1 \in \mathbb{H}$ and $q_2 \in \mathbb{H}$ and then summing them.

$$q_1 \cdot q_2 = q_{r1}q_{r2} + q_{i1}q_{i2} + q_{j1}q_{j2} + q_{k1}q_{k2} \tag{10}$$

The inner product operation of two quaternion vectors $q_1 \in \mathbb{H}^n$ and $q_2 \in \mathbb{H}^n$ is defined below.

$$\boldsymbol{q}_1 \cdot \boldsymbol{q}_2 = \boldsymbol{q}_{r1}^T\boldsymbol{q}_{r2} + \boldsymbol{q}_{i1}^T\boldsymbol{q}_{i2} + \boldsymbol{q}_{j1}^T\boldsymbol{q}_{j2} + \boldsymbol{q}_{k1}^T\boldsymbol{q}_{k2} \tag{11}$$

Hamiltonian product: The Hamiltonian product is also called quaternion multiplication. The Hamiltonian product $\otimes$ of two quaternions $q_1 \in \mathbb{H}$ and $q_2 \in \mathbb{H}$ is defined as

$$\begin{aligned} q_1 \otimes q_2 = {} & (q_{r1}q_{r2} - q_{i1}q_{i2} - q_{j1}q_{j2} - q_{k1}q_{k2}) \\ & + (q_{i1}q_{r2} + q_{r1}q_{i2} - q_{k1}q_{j2} + q_{j1}q_{k2})\boldsymbol{i} \\ & + (q_{j1}q_{r2} + q_{k1}q_{i2} + q_{r1}q_{j2} - q_{i1}q_{k2})\boldsymbol{j} \\ & + (q_{k1}q_{r2} - q_{j1}q_{i2} + q_{i1}q_{j2} + q_{r1}q_{k2})\boldsymbol{k} \end{aligned} \tag{12}$$

The Hamiltonian product of two quaternion vectors $q_1 \in \mathbb{H}^n$ and $q_2 \in \mathbb{H}^n$ is defined as

$$\begin{aligned} \boldsymbol{q}_1 \otimes \boldsymbol{q}_2 = {} & \left(\boldsymbol{q}_{r1} \circ \boldsymbol{q}_{r2} - \boldsymbol{q}_{i1} \circ \boldsymbol{q}_{i2} - \boldsymbol{q}_{j1} \circ \boldsymbol{q}_{j2} - \boldsymbol{q}_{k1} \circ \boldsymbol{q}_{k2}\right) \\ & + \left(\boldsymbol{q}_{i1} \circ \boldsymbol{q}_{r2} + \boldsymbol{q}_{r1} \circ \boldsymbol{q}_{i2} - \boldsymbol{q}_{k1} \circ \boldsymbol{q}_{j2} + \boldsymbol{q}_{j1} \circ \boldsymbol{q}_{k2}\right)\boldsymbol{i} \\ & + \left(\boldsymbol{q}_{j1} \circ \boldsymbol{q}_{r2} + \boldsymbol{q}_{k1} \circ \boldsymbol{q}_{i2} + \boldsymbol{q}_{r1} \circ \boldsymbol{q}_{j2} - \boldsymbol{q}_{i1} \circ \boldsymbol{q}_{k2}\right)\boldsymbol{j} \\ & + \left(\boldsymbol{q}_{k1} \circ \boldsymbol{q}_{r2} - \boldsymbol{q}_{j1} \circ \boldsymbol{q}_{i2} + \boldsymbol{q}_{i1} \circ \boldsymbol{q}_{j2} + \boldsymbol{q}_{r1} \circ \boldsymbol{q}_{k2}\right)\boldsymbol{k} \end{aligned} \tag{13}$$

where $\circ$ represents the multiplication between corresponding elements. It can be seen from Equation (12) that the Hamiltonian product has non-commutativity, i.e., $q_1 \otimes q_2 \neq q_2 \otimes q_1$.

4. Proposed Model

We use (h, r, t, τ) to represent quaternions in a temporal knowledge graph. Temporal knowledge graph $\mathcal{G}$ can be seen as a set of quaternions. For each training quaternion in the graph, it is necessary to generate corresponding negative samples using negative sampling technology to support effective representation learning of entities and relations. In this section, $\mathcal{G}\prime$ is used to represent the set of negative samples obtained by replacing the head or tail entities. Given a temporal knowledge graph, the purpose of the model is to learn an efficient quaternion representation and a scoring function $g(h, r, t, \tau) \in \mathbb{R}$ for the head entity, the tail entity, the relation, and the timestamp, respectively. The purpose of this scoring function is to measure the degree of authenticity of each quad, with the scores of true and effective quads being higher than those of invalid quads.

4.1. Model Definition

The purpose of the model is to use quaternion embedding to model entities, relationships, and timestamps. Given quaternion (h, r, t, τ), quaternion representations $\boldsymbol{q}_h$, $\boldsymbol{q}_r$, $\boldsymbol{q}_t$ and $\boldsymbol{q}_\tau \in \mathbb{H}^n$ corresponding to each constituent element are defined as follows:

$$\boldsymbol{q}_h = \boldsymbol{a}_h + \boldsymbol{b}_h\boldsymbol{i} + \boldsymbol{c}_h\boldsymbol{j} + \boldsymbol{d}_h\boldsymbol{k} \tag{14}$$

$$\boldsymbol{q}_r = \boldsymbol{a}_t + \boldsymbol{b}_t\boldsymbol{i} + \boldsymbol{c}_t\boldsymbol{j} + \boldsymbol{d}_t\boldsymbol{k} \tag{15}$$

$$\boldsymbol{q}_t = \boldsymbol{a}_h + \boldsymbol{b}_h\boldsymbol{i} + \boldsymbol{c}_h\boldsymbol{j} + \boldsymbol{d}_h\boldsymbol{k} \tag{16}$$

$$\boldsymbol{q}_\tau = \boldsymbol{a}_\tau + \boldsymbol{b}_\tau\boldsymbol{i} + \boldsymbol{c}_\tau\boldsymbol{j} + \boldsymbol{d}_\tau\boldsymbol{k} \tag{17}$$

where the coefficients of the real part as well as each imaginary part unit are n-dimensional vectors in real space.

While static knowledge graph completion models are able to learn multiple relational interactions between entities, they ignore the temporal factor and are therefore unable to reason effectively on temporal knowledge graphs. To address this problem, our model defines temporal evolution as a Hamiltonian product-based rotation transformation in quaternion space. Specifically, the model uses the unit vector form $\boldsymbol{q}_\tau^\triangleleft$ of the temporal quaternion to represent the rotation operator and rotates the head entity vector $\boldsymbol{q}_h$ and the tail entity vector $\boldsymbol{q}_t$ to obtain a temporal entity representation, respectively.

$$\boldsymbol{q}_{h,\tau} = \boldsymbol{q}_h \otimes \boldsymbol{q}_\tau^\triangleleft \tag{18}$$

$$\boldsymbol{q}_{t,\tau} = \boldsymbol{q}_t \otimes \boldsymbol{q}_\tau^\triangleleft \tag{19}$$

The quaternion vectors of the head and tail entities can be considered as a point in the quaternion space, while $\boldsymbol{q}_\tau^\triangleleft$ denotes a rotation transformation. After obtaining the time-dependent entities, the model uses the relation as a rotation transformation based on the Hamiltonian product, with the aim of rotating the time-dependent head entity to the vicinity of the time-dependent tail entity through the relation transformation. The general structure of the model is shown in Figure 1. To effectively represent the rotation transformation, we normalize the relation quaternion vector. For real quaternions, the model expects to satisfy $\boldsymbol{q}_{h,\tau} \otimes \boldsymbol{q}_\tau^\triangleleft \approx \boldsymbol{q}_{t,\tau}$. With two Hamiltonian product-based rotations, the entities, relations, and timestamps can be allowed to interact sufficiently to capture the potential interdependencies between these three.

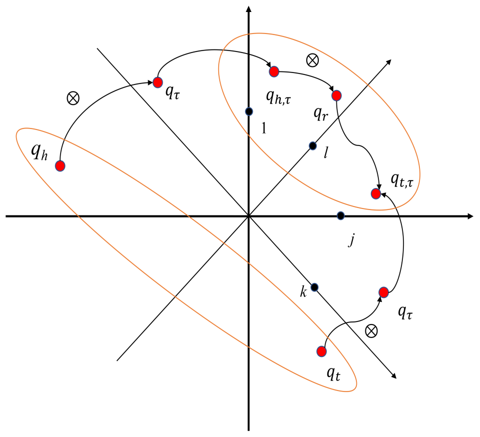

Figure 1. Schematic diagram of quaternion rotation.

Unlike the Euler distance-based scoring function used in previous research, our model uses the angle between two vectors to measure the similarity between them. To measure the quaternion hold, the scoring function of our model $g\,(h, r, t, \tau)$ is defined as follows:

$$g(h, r, t, \tau) = \boldsymbol{q}_{h,\tau} \otimes \boldsymbol{q}_r^\triangleleft \cdot \boldsymbol{q}_{t,\tau} \tag{20}$$

4.2. Model Training

To learn the model parameters, it is necessary to define the corresponding loss function based on the scoring function. For each quadruple in the training set, the head or tail entity is replaced by another entity in the entity set to obtain the corresponding negative sample. We define the following loss function to ensure that the score of the negative sample of the model is lower than that of the positive sample.

$$L = \sum_{(h,r,t,\tau)\in\mathcal{G}} \log\left(\frac{1+\exp(-l\cdot g(h,r,t,\tau))}{\sum_{T\in\mathcal{G}}\exp(T)}\right) + \lambda\|\boldsymbol{W}\|_2^2 \tag{21}$$

where θ denotes the learnable parameters of the model.

To prevent the model from overfitting and to improve the model's ability to generalize invisible facts, we add parametric regularisation constraints to the entity, relationship, and timestamp representations as shown in Equation (22).

$$L_{regular}(\theta) = \sum_{(h,r,t,\tau)} \left(\|\boldsymbol{q}_h + \boldsymbol{q}_r + \boldsymbol{q}_t + \boldsymbol{q}_\tau\|_2^2 + \left\|\boldsymbol{q}_{h,\tau} + \boldsymbol{q}_{t,\tau}\right\|_2^2 \right) \tag{22}$$

Furthermore, we assume that the existing knowledge of past time intervals can be used to accurately capture the progression pattern of the whole graph during model training. Literature [26] presents experimental results indicating that subgraphs of successive time steps exhibit minimal changes, and the embedding space of successive time steps should also exhibit smoothness. Consequently, the TKGR model must have smoothing requirements on successive time intervals.

$$L_{smooth}(\theta) = \mid \mathcal{T} \mid \sum_{(h,r,t,\tau)\in\mathcal{G}} |q_\tau + q_h + q_t + q_r|_2^2 + (\mid \mathcal{T} \mid -1) \sum_{(h,r,t,\tau)\in\mathcal{G}} |q_r|_2^2 \tag{23}$$

where $\mid \mathcal{T} \mid$ denotes the number of time steps. By adding parameter regularization constraints as well as temporal smoothing constraints to the model, the final loss function of the TKGR model is shown below.

$$\mathcal{L}(\theta) = L(\theta) + \lambda_1 L_{regular}(\theta) + \lambda_2 L_{smooth}(\theta) \tag{24}$$

where λ_1 and λ_2 represents the coefficient of the regularization term.

Meanwhile, a suitable initialization method can improve the training speed and reduce the risk of gradient explosion or gradient disappearance. It has been shown that the parameters of the hyper-complex representation cannot simply be initialized randomly. Therefore, in order to improve the convergence of the TKGR model, we adopt the following method to initialize the model parameters:

$$w_r = \varphi \cos(\theta) \tag{25}$$

$$w_i = \varphi \mathcal{Q}_{img_i}^{\triangleleft} \sin(\theta) \tag{26}$$

$$w_j = \varphi \mathcal{Q}_{img_j}^{\triangleleft} \sin(\theta) \tag{27}$$

$$w_k = \varphi \mathcal{Q}_{img_k}^{\triangleleft} \sin(\theta) \tag{28}$$

where w_r, w_i, w_j, and w_k represent the real and imaginary parts of the initialized quaternion, respectively. θ is randomly generated over the interval $[-\pi, \pi]$. $\mathcal{Q}_{img}^{\triangleleft}$ denotes the normalized unit quaternion, generated according to a uniform distribution in the interval $[0, 1]$. φ is randomly generated based on the value of quaternion and the chosen initialization principle.

Finally, we use the AdaGrad algorithm to optimize the objective function. Algorithm 1 describes the learning process of the TKGR model.

Algorithm 1 Learning process of the TKGR model

```
Require: Temporal knowledge graph entity set $\mathcal{E}$. Relationship set $\mathcal{R}$. Timestamp set $\mathcal{T}$.
    Training set $\mathcal{G}_{train} = \{(h, r, t, \tau)\}$. Dimension of quaternion vector $n$. Learning rate
    $\alpha$. Batch size $b$. Regular term coefficient $\lambda_1$ and $\lambda_2$. Number of negative samples $\gamma$.
    Number of model iterations $M$.
Ensure: Vector representation of entities, relationships, and timestamps
 1: Initialize model parameters according to Equation (21)
 2: for iteration = 1 to $N$ do
 3:     $\mathcal{E} \leftarrow uniform(\mathcal{E}) foreache \in \mathcal{E}$
 4:     $S_{batch} \leftarrow Sample(\mathcal{G}_{train}, b)$
 5:     for each$(h, r, t, \tau) \in S_{batch}$ do
 6:         $S(h\prime, r, t\prime, \tau) \leftarrow Sample\left(S_{(h,r,t,\tau)}\right)$
 7:     end for
 8:     Determine the numerical values of the scoring function for both positive and nega-
    tive samples using Equation (16)
 9:     Determine the numerical value of the loss function based on Equation (20)
10:     Update model parameters
11: end for
```

5. Experiments

The main purpose of this section is to verify the superiority of the model proposed in this paper. Firstly, we list the state-of-the-art algorithms and experimental setups, and then we conduct experimental comparisons from different perspectives. The following are the specific experimental sessions.

5.1. Datasets

We select the three most commonly used temporal knowledge graph datasets ICEWS14, ICEWS05-15, and GDELT which are standard datasets used to evaluate temporal knowledge graph completion models, and Table 1 lists the detailed statistical information of these three datasets.

ICEWS14: This dataset consists of four tuples taken from social news related to political events. ICEWS14 is a subset of ICEWS that focuses on events from the year 2014. It has 7128 entities and 230 relationships and spans 365 time steps.

ICEWS05-15: ICEWS05-15 is a subset of the ICEWS dataset containing events that occurred between 2005 and 2015. The dataset contains 10,488 entities and 251 relationships and spans 4017 time steps.

GDELT: This dataset is a collection of human social relationships. We isolate a subset from source [27] that represents events that occurred between 2015 and 2016. This subset contains 500 entities and 20 relationships and spans 366 time steps.

Table 1. Details of the datasets.

Datasets	Entities	Relations	Time Steps	Training	Validation	Test
ICEWS14	7128	230	365	72,826	8941	8963
ICEWS05-15	10,488	251	4017	386,962	46,275	46,092
GDELT	500	20	366	2,735,685	341,961	341,961

5.2. Evaluation Metrics

We use the same evaluation measures for the dynamic knowledge graph completion task as for the static knowledge graph completion task. Firstly, for each quaternion in the test set, two types of candidate quaternions are created by replacing either the head or tail entity. Any potential quaternion found in $\mathcal{G}$ must be removed. The exact formula is given below.

$$h_{candidate} = \{(h',r,t,\tau) \mid h' \in \varepsilon, (h',r,t,\tau \notin \mathcal{G})\} \tag{29}$$

$$h_{candidate} = \{(h,r,t',\tau) \mid t' \in \varepsilon, (h,r,t',\tau \notin \mathcal{G})\} \tag{30}$$

It is then necessary to calculate the ratings of the test quaternion and all candidate quaternions and rank the ratings in a descending order to obtain the ranking of the test quaternion in each of the two candidate sets, which is denoted by f_h and f_t, respectively. Based on this ranking, we select MRR and Hit@k (k = 1, 3, 10) as the model evaluation metrics, respectively, where MRR is the inverse of the average ranking and Hit@k is the percentage of the ranking in the top k. The formula for MRR is shown below.

$$MRR = \frac{1}{2 \cdot \mid \mathcal{G}_{test} \mid} \sum_{(h,r,t,\tau)\in\mathcal{G}_{test}} \left(\frac{1}{f_h} + \frac{1}{f_t}\right) \tag{31}$$

where $\mid \mathcal{G}_{test} \mid$ is the size of the test set. If the MRR value is larger, it means that the performance of the model is better. The formula for Hit@k is shown below.

$$Hit@k = \frac{1}{2 \cdot \mid \mathcal{G}_{test} \mid} \sum_{(h,r,t,\tau)\in\mathcal{G}_{test}} (I(f_h \leqslant k) + I(f_t \leqslant k)) \tag{32}$$

where $I(\cdot)$ denotes the indicator function, e.g., if $f_h \leqslant k$, then $I(f_h \leqslant k) = 1$, otherwise $I(f_h \leqslant k) = 0$. In the experiments, the Hit@k results are multiplied by 100.

5.3. Experimental Setup

We use the PyTorch framework to train our model, taking advantage of the powerful GeForce RTX 2080 GPU for improved performance. For coefficients λ_1 and λ_2 of the parameter regularization term and the temporal smoothing constraint term, we assign the same values to both coefficients as they have the same number of paradigms. We then select the most appropriate values from the range $\{0.1, 0.01, 0.001\}$.

The embedding dimension is 500 and the learning rate is 0.1. Both ICEWS datasets have a negative-to-positive sample ratio of 500, whereas the GDELT dataset has a ratio of 5. The batch size is set to 512. Different hyperparameters are used for each dataset to optimize the training process, taking into account their different sizes. For example, GDELT is much larger than the other two datasets, resulting in reduced hyperparameters. The model is iteratively trained 5000 times for the ICEWS14 and ICEWS05-15 datasets and 1000 times for the GDELT dataset. The training technique uses the AdaGrad optimization algorithm within the Small Batch Random Gradient Descent methodology.

5.4. Baselines

We compare different knowledge graph embedding techniques, including TransE [8], DistMult [28], SimplE [29], ConT [20], TTransE [30], HyTE [21], TA-DisMult [22], DE-SimplE [23], RoAN [31], SubEE [24], T-GAE [32], GLANet [33], BoxTE [25], DualMatch [34], DKGE [24], and TLmod [24]. These methods use different mechanisms to handle entity relationships, temporal information, global and local structure, and reasoning processes, with the aim of improving the expressiveness and inference accuracy of knowledge graphs.

5.5. Link Prediction

Tables 2 and 3 show the results of the TKGR and baseline models in predicting temporal connections for three datasets, respectively. The experimental results show that TKGR outperforms the mainstream baseline model on most metrics, highlighting the effectiveness of TKGR. Quaternion rotation procedures based on Hamiltonian products allow for more extensive and complex interactions between things, time, and connections. TKGR shows superior performance to the TeRo model on the ICEWS05-15 dataset, particularly in terms of MRR and Hit@10. It achieves a significant gain of 2.55% in MRR and 1.38% in Hit@10. The ICEWS14 dataset has a larger number of entities, relationships, timestamps, and training samples. TKGR has the ability to represent complicated connection categories, such as symmetric/anti-symmetric and inverse relationships, making it very suitable for complex datasets.

Table 2. Link prediction results on datasets ICEWS14 and ICEWS05-15.

Model	ICEWS14				ICEWS05-15			
	MRR	**Hit@1**	**Hit@3**	**Hit@10**	**MRR**	**Hit@1**	**Hit@3**	**Hit@10**
TransE	0.28	9.4	-	63.7	0.294	9.0	-	66.3
DistMult	0.439	32.3	-	67.2	0.456	33.7	-	69.1
SimplE	0.458	34.1	51.6	68.7	0.478	35.9	53.9	70.8
ConT	0.185	11.7	20.5	31.5	0.163	10.5	18.9	27.2
TTransE	0.255	7.4	-	60.1	0.271	8.4	-	61.6
HyTE	0.297	10.8	41.6	65.5	0.316	11.6	44.5	68.1
TA-DistMult	0.477	36.3	-	68.6	0.474	34.6	-	72.8
DE-SimplE	0.526	41.8	59.2	72.5	0.513	39.2	57.8	74.8
RoAN	0.560	45.3	63.1	74.2	0.587	47.1	66.9	79.7
SubEE	0.519	43.5	62.4	72.1	0.572	46.3	65.4	78.1
GLANet	0.509	44.6	63.1	73.5	0.564	46.7	65.3	78.7
T-GAE	0.554	44.8	62.1	73.8	0.552	47.5	65.1	77.2
DKGE	0.549	45.2	62.9	73.9	0.591	47.3	65.8	79.2
BoxTE	0.549	45.2	62.9	74.1	0.594	47.9	66.3	78.2
TKGR	0.568	45.9	63.6	74.8	0.602	48.3	67.9	80.8

Table 3. Link prediction results on dataset GDELT.

Model	**MRR**	**Hit@1**	**Hit@3**	**Hit@10**
TransE	0.123	12.7	15.1	31.2
DistMult	0.206	13.7	21.2	34.7
SimplE	0.216	12.3	22.2	35.9
ConT	0.143	11.8	15.7	27.1
TTransE	0.107	12.6	16.3	30.8
HyTE	0.112	13.1	16.4	33.6
TA-DistMult	0.215	12.3	21.1	35.5
DE-SimplE	0.226	14.1	24.8	37.3
RoAN	0.232	13.9	23.7	38.4
SubEE	0.224	12.4	22.6	38.8
GLANet	0.235	13.6	23.9	38.4
T-GAE	0.217	12.4	22.6	37.7
DKGE	0.225	13.1	22.9	38.1
BoxTE	0.229	13.7	23.5	38.5
TKGR	0.257	15.2	25.1	39.5

Although the performance of the TKGR on the GDELT dataset is not particularly remarkable when compared to other models considered to be state of the art, it is still worthy of praise. A slight improvement can be seen in all measures when comparing the BoxTE, T-GAE, and RoAN models. On the other hand, the level of performance improvement is not particularly remarkable. The number of elements and links included in the GDELT dataset is extremely limited. On the other hand, both the training and the test sets can be considered quite large. All the models receive relatively low test scores as a result of this increased difficulty in the task they have to perform.

Furthermore, traditional representation learning strategies, such as TransE and DistMult models, are typically used to analyze static knowledge graphs. As a consequence, their ability to predict temporal relations is often insufficient. Both approaches ignore temporal information, which makes it difficult for them to accurately capture the temporal dynamics of their respective knowledge networks. In conclusion, the results of the experiments show that TKGR is not only appropriate, but also absolutely necessary. This is due to its ability to effectively capture the dynamic evolutionary properties of things and to effectively exploit temporal information.

5.6. Impact of Regularization

The final objective function is modified by TKGR to incorporate parameter regularization and temporal smoothing requirements. These modifications are derived from the results of previous experiments. In order to demonstrate the impact of these two regularization parameters on the overall performance of the model, we perform an experimental validation using the ICEWS14 dataset. This can be achieved by analyzing the fluctuations in the experimental results on the test set and examining them when the objective function does not include these two regularisation factors. At the same time, the other parameters are kept at their initial levels.

The results of the two comparative experiments are shown in Table 4. It is clear from the table that the regularization term has a more significant impact on the results. For example, the MRR and Hit@1 metrics improve the performance of the model by 1% and 1.8%, respectively, compared to the performance of the model before the imposition of the time smoothing constraints and parameter regularization. Therefore, these two regularisation terms are more beneficial to the model.

Table 4. Effect of regularization terms on dataset ICEWS14.

Loss Function	MRR	Hit@1	Hit@3	Hit@10
$L(\theta) + \lambda_1 L_r(\theta) + \lambda_2 L_s(\theta)$	0.568	45.9	63.6	74.8
$L(\theta)$	0.563	45.1	63.2	74.4

5.7. Evolution of Relations

According to the results of the current research, it is clear that different temporal knowledge graph completion models use temporal information in different ways. For example, the TA-TransE model incorporates both relational and temporal data in order to acquire a relational representation that incorporates temporal information. The ability to capture temporal representations of things and interpersonal relationships is one of the capabilities of the HyTE model. In this research, TKGR is only used to represent the progression of entities, and the use of static representations is retained when it comes to relationships.

If we focus only on the temporal evolution of relationships, the performance of the variant model is negatively affected across the board. On the other hand, if we consider the simultaneous evolution of entities and relations, the performance of the variant model is comparable to that of the TKGR modelling approach. The following conclusion can be drawn from the analysis of the data obtained from the experiment. It is possible that this is the result of a mismatch between the number of relations in a dataset and the number of entities in their dataset.

As a result, the evolution of entities over time provides a more accurate measure of the efficient use of temporal information. It also appears that entities tend to change their nature over time, whereas partnerships tend to be more stable in nature. This is something that can be understood through direct experience in actual life circumstances. In partnerships, it is possible for a person's state to change at different times before and after the partnership. Due to the nature of the relationship, which tends to make the opposite true, this is the current state of affairs. One of TKGR's main objectives is to ensure that the units continue to grow. This is performed in order to maintain the simplicity of the model and the geometric significance of the changes that take place within the relationships (Table 5).

Table 5. Experimental results for different variant models on dataset ICEWS14.

Variations	MRR	Hit@1	Hit@3	Hit@10
$\boldsymbol{q}_{h,\tau} \otimes \boldsymbol{q}_r^{\triangleleft} \cdot \boldsymbol{q}_{t,\tau}$	0.568	45.9	63.6	74.8
$\boldsymbol{q}_h \otimes \boldsymbol{q}_{r,\tau}^{\triangleleft} \cdot \boldsymbol{q}_t$	0.563	45.1	63.2	74.4
$\boldsymbol{q}_{h,\tau} \otimes \boldsymbol{q}_{r,\tau}^{\triangleleft} \cdot \boldsymbol{q}_{t,\tau}$	0.563	45.1	63.2	74.4

5.8. Parameter Adjustment

This section is dedicated to the study of regularization coefficients λ_1 and λ_2. We determine and select the optimal settings. We start by running the algorithm repeatedly for a total of one thousand iterations. We then analyze the results of Hit@10 on the validation set to determine the parameters.

Figure 2 visually demonstrates the effect of adjusting the value of the regularization factor on the performance of the model using the ICEWS14 dataset. The results indicate that the model achieves optimal performance when the value of λ is set to λ_1 and λ_2, both equal to 0.01. In the trials described above, both λ_1 and λ_2 are assigned a value of 0.01.

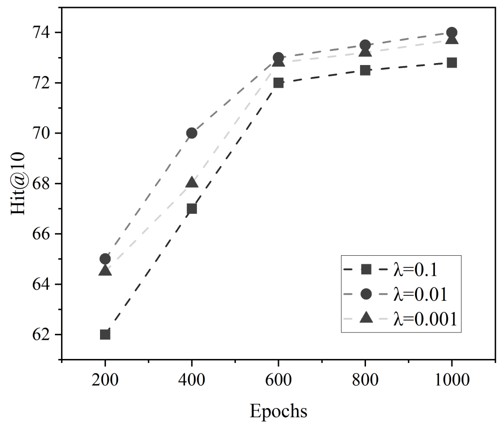

Figure 2. The influence of regularization coefficient on the TKGR model.

The summary of findings highlights:

Model Performance Superiority: Our TKGR model, utilizing quaternion rotations, consistently outperforms state-of-the-art methods across all benchmark datasets (ICEWS14, ICEWS05-15, and GDELT). This underscores its efficacy in capturing the temporal evolution of knowledge graphs, as demonstrated by improved metrics such as Hit@10 and Mean Reciprocal Rank (MRR).

Temporal Dependency Handling: The model adeptly handles complex temporal dependencies, as evidenced by its enhanced predictive accuracy for time-varying entity relationships. For instance, in the case of South Korean presidential succession, our model accurately captures the temporal progression of leadership changes, illustrating how the inclusion of temporal information can significantly enhance the understanding and forecasting of such events in knowledge graphs.

Parameter Sensitivity Analysis: We demonstrate the sensitivity of model performance to the regularization coefficient λ, identifying optimal values (λ = 0.01) that balance model complexity and generalization. These findings ensure the model's robustness across different dataset characteristics, validating the effectiveness of our chosen regularization strategy.

Methodological Innovation: Our work introduces a groundbreaking application of quaternion rotations to temporal knowledge graph representation. This innovation not only improves the learning process's expressiveness and efficiency compared to complex number-based approaches, but also offers a fresh perspective on temporal knowledge graph modeling that addresses the limitations of prior works. This approach effectively captures the inherently dynamic nature of real-world knowledge, as exemplified in our experimental results.

6. Conclusions

This paper presents a revolutionary strategy for depicting knowledge graphs with temporal attributes using quaternion rotations. The proposed TKGR model effectively captures the evolution of entities and their interdependencies in complex number spaces, as demonstrated by its superior performance on benchmark datasets. The utilization of quaternion rotations enables a more expressive and efficient learning process compared to traditional complex number-based approaches, particularly in representing temporal transitions in knowledge graphs.

While our work represents a significant advancement in temporal knowledge graph representation, we acknowledge several limitations that warrant consideration and future exploration:

Scalability: Although the TKGR model demonstrates promising results on benchmark datasets, its scalability to extremely large knowledge graphs remains to be investigated. Future research could explore parallelization strategies or more computationally efficient quaternion operations to accommodate massive, real-world knowledge graphs with extensive temporal information.

Temporal Granularity: Our current approach handles temporal transitions at a relatively coarse level. Investigating methods to handle finer-grained temporal resolutions, such as sub-daily or event-specific timestamps, could further enhance the model's ability to capture intricate temporal patterns and dynamics.

Dynamic Integration of New Data: Real-world knowledge graphs continuously evolve over time, necessitating models that can seamlessly integrate new temporal information. Future work could explore online learning mechanisms or incremental update strategies for the TKGR model to adapt to temporal changes in near-real time.

Interpretability: While the quaternion-based representation offers a novel perspective on temporal knowledge graph modeling, enhancing the interpretability of the learned quaternion representations and their relationship to temporal phenomena would deepen our understanding of the model's decision-making process and foster trust in its predictions.

Cross-domain Transferability: The effectiveness of the TKGR model across diverse domains and types of knowledge graphs should be assessed. Future studies could examine the model's performance on knowledge graphs from different industries, such as healthcare, finance, or social sciences, to assess its generalizability and potential domain-specific adaptations.

In conclusion, this study introduces a novel and effective technique for representing temporal knowledge graphs using quaternion rotations. Despite its promising results, we recognize several limitations that present opportunities for future research. Addressing these issues through continued innovation and refinement will contribute to the development of increasingly powerful and versatile models for temporal knowledge graph representation, ultimately enhancing their utility in a wide range of applications.

Author Contributions: Conceptualization, S.L. and Q.L.; methodology, S.L.; software, Q.L.; validation, Q.L.; formal analysis, S.L.; writing—original draft preparation, S.L.; writing—review and editing, Q.L. and S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zeng, X.; Tu, X.; Liu, Y.; Fu, X.; Su, Y. Toward better drug discovery with knowledge graph. *Curr. Opin. Struct. Biol.* **2022**, *72*, 114–126. [CrossRef] [PubMed]
2. Wang, J.; Wang, B.; Gao, J.; Li, X.; Hu, Y.; Yin, B. TDN: Triplet Distributor Network for Knowledge Graph Completion. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 13002–13014. [CrossRef]

3. Huang, C.; Zhong, Y. A Network Representation Learning Method Fusing Multi-dimensional Classification Information of Nodes. *IAENG Int. J. Comput. Sci.* **2023**, *50*, 94–105.
4. Peng, C.; Xia, F.; Naseriparsa, M.; Osborne, F. Knowledge graphs: Opportunities and challenges. *Artif. Intell. Rev.* **2023**, *56*, 13071–13102. [CrossRef] [PubMed]
5. Zhou, B.; Shen, X.; Lu, Y.; Li, X.; Hua, B.; Liu, T.; Bao, J. Semantic-aware event link reasoning over industrial knowledge graph embedding time series data. *Int. J. Prod. Res.* **2023**, *61*, 4117–4134. [CrossRef]
6. Guo, Y.; Wang, L.; Zhang, Z.; Cao, J.; Xia, X.; Liu, Y. Integrated modeling for retired mechanical product genes in remanufacturing: A knowledge graph-based approach. *Adv. Eng. Inform.* **2024**, *59*, 102254. [CrossRef]
7. Cao, B.; Zhao, J.; Gu, Y.; Ling, Y.; Ma, X. Applying graph-based differential grouping for multiobjective large-scale optimization. *Swarm Evol. Comput.* **2020**, *53*, 100626. [CrossRef]
8. Bordes, A.; Usunier, N.; Garcia-Durán, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. In Proceedings of the 26th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; Volume 2, pp. 2787–2795.
9. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge graph embedding by translating on hyperplanes. In Proceedings of the AAAI Conference on Artificial Intelligence, Quebec City, QC, Canada, 27–31 July 2014; Volume 28.
10. Wang, X.; Lyu, S.; Wang, X.; Wu, X.; Chen, H. Temporal knowledge graph embedding via sparse transfer matrix. *Inf. Sci.* **2023**, *623*, 56–69. [CrossRef]
11. Li, M.M.; Huang, K.; Zitnik, M. Graph representation learning in biomedicine and healthcare. *Nat. Biomed. Eng.* **2022**, *6*, 1353–1369. [CrossRef] [PubMed]
12. Bai, L.; Chai, D.; Zhu, L. RLAT: Multi-hop temporal knowledge graph reasoning based on Reinforcement Learning and Attention Mechanism. *Knowl. Based Syst.* **2023**, *269*, 110514. [CrossRef]
13. Nickel, M.; Rosasco, L.; Poggio, T. Holographic embeddings of knowledge graphs. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
14. Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; Bouchard, G. Complex embeddings for simple link prediction. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 2071–2080.
15. Baghershahi, P.; Hosseini, R.; Moradi, H. Self-attention presents low-dimensional knowledge graph embeddings for link prediction. *Knowl.-Based Syst.* **2023**, *260*, 110124. [CrossRef]
16. Dettmers, T.; Minervini, P.; Stenetorp, P.; Riedel, S. Convolutional 2d knowledge graph embeddings. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LO, USA, 2–7 February 2018; Volume 32.
17. Li, Z.; Zhao, Y.; Zhang, Y.; Zhang, Z. Multi-relational graph attention networks for knowledge graph completion. *Knowl.-Based Syst.* **2022**, *251*, 109262. [CrossRef]
18. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710.
19. Schlichtkrull, M.; Kipf, T.N.; Bloem, P.; Van Den Berg, R.; Titov, I.; Welling, M. Modeling relational data with graph convolutional networks. In Proceedings of the The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Greece, 3–7 June 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 593–607.
20. Ma, Y.; Tresp, V.; Daxberger, E.A. Embedding models for episodic knowledge graphs. *J. Web Semant.* **2019**, *59*, 100490. [CrossRef]
21. Dasgupta, S.S.; Ray, S.N.; Talukdar, P. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2001–2011.
22. Garcia-Duran, A.; Dumančić, S.; Niepert, M. Learning Sequence Encoders for Temporal Knowledge Graph Completion. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 4816–4821.
23. Goel, R.; Kazemi, S.M.; Brubaker, M.; Poupart, P. Diachronic embedding for temporal knowledge graph completion. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 3988–3995.
24. Wan, G.; Zhou, Z.; Zheng, Z.; Du, B. Sub-Entity Embedding for inductive spatio-temporal knowledge graph completion. *Future Gener. Comput. Syst.* **2023**, *148*, 240–249. [CrossRef]
25. Messner, J.; Abboud, R.; Ceylan, I.I. Temporal knowledge graph completion using box embeddings. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; Volume 36, pp. 7779–7787.
26. Xu, Y.; Sun, S.; Zhang, H.; Yi, C.; Miao, Y.; Yang, D.; Meng, X.; Hu, Y.; Wang, K.; Min, H.; et al. Time-aware graph embedding: A temporal smoothness and task-oriented approach. *ACM Trans. Knowl. Discov. Data TKDD* **2021**, *16*, 1–23. [CrossRef]
27. Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; Association for Computational Linguistics: Pittsburgh, PA, USA, 2014; p. 1724.
28. Yang, B.; Yih, W.T.; He, X.; Gao, J.; Deng, L. Embedding entities and relations for learning and inference in knowledge bases. *arXiv* **2014**, arXiv:1412.6575.
29. Kazemi, S.M.; Poole, D. SimplE embedding for link prediction in knowledge graphs. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 2–8 December 2018; pp. 4289–4300.

30. Jiang, T.; Liu, T.; Ge, T.; Sha, L.; Chang, B.; Li, S.; Sui, Z. Towards time-aware knowledge graph completion. In Proceedings of the COLING 2016, the 26th International Conference on Computational Linguistics, Osaka, Japan, 11–16 December 2016; Technical Papers; pp. 1715–1724.
31. Bai, L.; Ma, X.; Meng, X.; Ren, X.; Ke, Y. RoAN: A relation-oriented attention network for temporal knowledge graph completion. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106308. [CrossRef]
32. Hou, X.; Ma, R.; Yan, L.; Ma, Z. T-GAE: A Timespan-aware Graph Attention-based Embedding Model for Temporal Knowledge Graph Completion. *Inf. Sci.* **2023**, *642*, 119225. [CrossRef]
33. Wang, J.; Lin, X.; Huang, H.; Ke, X.; Wu, R.; You, C.; Guo, K. GLANet: Temporal knowledge graph completion based on global and local information-aware network. *Appl. Intell.* **2023**, *53*, 19285–19301. [CrossRef]
34. Liu, X.; Wu, J.; Li, T.; Chen, L.; Gao, Y. Unsupervised Entity Alignment for Temporal Knowledge Graphs. In Proceedings of the ACM Web Conference 2023, Austin, TX, USA, 30 April–4 May 2023; pp. 2528–2538.