TeAST: Temporal Knowledge Graph Embedding via Archimedean Spiral Timeline

Jiang Li[1,2] **, Xiangdong Su**[1,2] [*]**, Guanglai Gao**[1,2]
[1] College of Computer Science, Inner Mongolia University, Hohhot, China
[2] National & Local Joint Engineering Research Center of Intelligent Information
Processing Technology for Mongolian, Hohhot, China
lijiangimu@gmail.com, cssxd@imu.edu.cn, csggl@imu.edu.cn

Abstract

Temporal knowledge graph embedding (TKGE) models are commonly utilized to infer the missing facts and facilitate reasoning and decision-making in temporal knowledge graph based systems. However, existing methods fuse temporal information into entities, potentially leading to the evolution of entity information and limiting the link prediction performance of TKG. Meanwhile, current TKGE models often lack the ability to simultaneously model important relation patterns and provide interpretability, which hinders their effectiveness and potential applications. To address these limitations, we propose a novel TKGE model which encodes **T**emporal knowledge graph **e**mbeddings via **A**rchimedean **S**piral **T**imeline (TeAST), which maps relations onto the corresponding Archimedean spiral timeline and transforms the quadruples completion to 3th-order tensor completion problem. Specifically, the Archimedean spiral timeline ensures that relations that occur simultaneously are placed on the same timeline, and all relations evolve over time. Meanwhile, we present a novel temporal spiral regularizer to make the spiral timeline orderly. In addition, we provide mathematical proofs to demonstrate the ability of TeAST to encode various relation patterns. Experimental results show that our proposed model significantly outperforms existing TKGE methods. Our code is available at https://github.com/IMU-MachineLearningSXD/TeAST.

1 Introduction

Knowledge graph (KG) expresses the relations of real-world entities and allows for reasoning new facts, which enables a wide range of applications in natural language processing (Chen et al., 2019; Junior et al., 2020; Hu et al., 2021). It stores a vast amount of knowledge in the form of triplets. These triplets are typically denoted as (s, r, o), where s, r and o represent the subject, the relation, and the object. Since knowledge changes over time, researchers introduced timestamps into knowledge graphs to create temporal knowledge graphs (TKGs). In TKGs, each knowledge fact is represented as a quadruple (s, r, o, τ), where τ denotes the timestamp at which the fact was true. This allows for more precise representation and querying of information in knowledge graphs, enabling applications that require an understanding of the evolution of knowledge over time. Given the inherent incompleteness of most KGs and TKGs, knowledge graph embedding (KGE) and temporal knowledge graph embedding (TKGE) have been widely investigated to infer the missing facts using the existing ones. In particular, TKGE has gained significant attention for its ability to represent and analyze knowledge over time. This work focuses on TKGE.

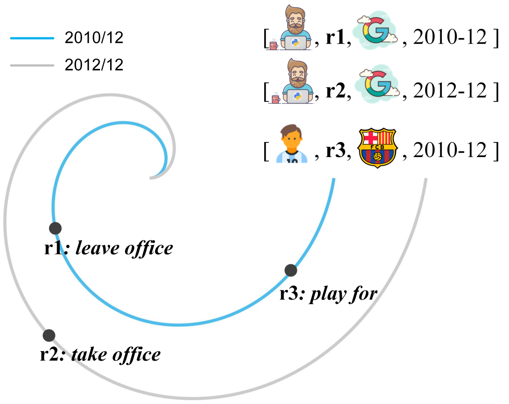


Figure 1: A brief illustration of mapping relations to Archimedean spiral timeline. Three facts are *(Guido van Rossum, r1, Google, 2010-12), (Guido van Rossum, r2, Google, 2012-12) and (Messi, r3, FCB, 2010-12).*

With the advancement of deep learning, researchers have proposed a number of KGE approaches. These approaches typically involve learning low-dimensional embeddings of entities and relations, and then using a score function to measure

[*] Corresponding Author

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 15460–15474
July 9-14, 2023 ©2023 Association for Computational Linguistics

the plausibility of triplets (Ji et al., 2022). While existing KGE approaches have been shown to be effective on static knowledge graphs, they cannot be directly applied to TKGs due to the fact that real-world knowledge is dynamic and changes over time. To address this issue, researchers have designed TKGE models that are capable of capturing the temporal information and dynamic nature of real-world facts. Recent TKGE models (Lacroix et al., 2020; Xu et al., 2020a, 2021; Chen et al., 2022) have shown very impressive completion performance on TKGs.

Nevertheless, there are two problems with these TKGE models. Firstly, the fusion of temporal information into entities led to a potential evolution of entity information, thus limiting the link prediction performance on TKG. In fact, the meaning of entities in quadruples does not change over time, whereas the relations between connected entities do. Secondly, existing TKGE models are not capable of simultaneously encoding important relation patterns and providing interpretability, which hinders their effectiveness and potential applications.

To tackle these issues, we draw inspiration from the Archimedean spiral and design **T**emporal knowledge graph **e**mbeddings via **A**rchimedean **S**piral **T**imeline (TeAST). Specifically, we first map relations onto the corresponding Archimedean spiral timeline and form a unified representation for the timestamp and the relation. As shown in Figure 1, we expect relations at the same time to be on the same timeline and relations evolve over time. That is, we simplify the quadruples (s, r, o, τ) to a triplet $(s, r \circledcirc \tau, o)$, where $\circledcirc$ denotes Archimedean spiral operation. As a result, we transform the TKG embedding as 3th-order tensor completion problem in the complex space. Next, we optimize the graph embeddings through tensor factorization. In addition, we propose a new temporal spiral regularizer to constrain the time representation and make the spiral timeline orderly. We further provide mathematical proofs to demonstrate the ability of TeAST to encode various relation patterns. Experiments show that our method significantly outperforms the existing methods on TKGE benchmarks.

Different from the existing TKGE models, we map relations onto the Archimedean spiral timeline and avoid incorporating temporal information into the entities. It ensures that the relations can evolve over time and the entities remain unchanged in TKGs. This is consistent with real-world facts.

2 Related Work

2.1 Static Knowledge Graph Embedding

Motivated by the translation invariance principle in word2vec (Mikolov et al., 2013), TransE defines the distance between $\boldsymbol{e_s} + \boldsymbol{e_r}$ and $\boldsymbol{e_o}$ with the l_1 or l_2 norm constraint, where $\boldsymbol{e_s}$, $\boldsymbol{e_o}$ denote entity embedding vectors and $\boldsymbol{e_r}$ denote relation embedding vectors. The score function of TransE is defined as $\phi(s, r, o) = ||\boldsymbol{e_s} + \boldsymbol{e_r} - \boldsymbol{e_o}||_p$. Following TransE, TransH (Wang et al., 2014), TransR (Lin et al., 2015) and TransD (Ji et al., 2015) employ different projection strategies to adjust graph embeddings. Different from the above distance based models, RESCAL (Nickel et al., 2011), DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016) and SimplE (Kazemi and Poole, 2018) employ tensor factorization based to model knowledge graphs, in which each relation r is mapped into a latent semantic matrix $\mathbf{M_r}$. In addition, RotatE (Sun et al., 2019) and QuatE (Zhang et al., 2019) treat each relation as a rotation in complex space and in the quaternion space, respectively.

2.2 Temporal Knowledge Graph Embedding

Analogously to KGE models, TKGE models add the temporal information and calculates the score function for the quadruples to evaluate its reasonableness. Therefore, most TKGE models are based on existing KGE models. TTransE (Leblay and Chekol, 2018) extends TransE and encodes time stamps τ as translations same as relations. Hence, the score function of TTransE is denoted as $\phi(s, r, o, \tau) = ||\boldsymbol{e_s} + \boldsymbol{e_r} + \boldsymbol{e_\tau} - \boldsymbol{e_o}||_p$. Furthermore, TA-TransE (García-Durán et al., 2018) and TA-DistMult (García-Durán et al., 2018) encode timestamps based on TransE and DistMult, respectively. TComplEx (Lacroix et al., 2020) and TNTComplEx (Lacroix et al., 2020) build on ComplEx and perform a 4th-order tensor decomposition of a TKG. DE-SimplE (Goel et al., 2020) adds a diachronic entity (DE) embedding function to learn the temporal entities. ChronoR (Sadeghian et al., 2021) is based on RotatE and learns a k-dimensional rotation transformation parametrized by relation-time pairs. Next, each subject entity is transformed with the rotation. TeLM (Xu et al., 2021) performs more expressive multivector representations to encode a temporal KG and utilizes the asymmetric geometric product. In addition, Rotate-QVS (Chen et al., 2022) builds on QuatE and encodes both entities and relations as quaternion em-

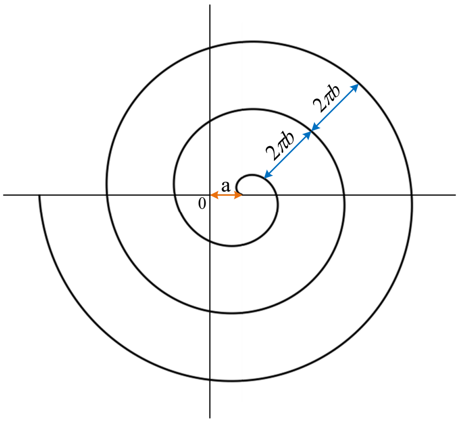


Figure 2: An illustration of an Archimedean spiral.

beddings, in which the temporal entity embeddings are represented as rotations in the quaternion space. Recently, BoxTE (Messner et al., 2022) models the TKGE based on a box embedding model BoxE (Abboud et al., 2020).

3 Background and Notation

3.1 Archimedean Spiral

As mentioned, we expect the relations with the same timestamp to be on the same timeline and all relations evolve over time. We choose the Archimedean spiral to model TKGs in the proposed method. Through the angle of rotation around the origin, Archimedean spiral provides the possibility of distinguishing the relations on the same timeline.

In mathematics, Archimedean spiral (also known as the arithmetic spiral) was named in honor of the Greek mathematician Archimedes. As shown in Figure 2, it is the locus corresponding to the locations over time of a point moving away from a fixed point with a constant speed along a line that rotates with constant angular velocity. Equivalently, in polar coordinates (ξ, θ) it can be described by the equation:

$$\xi = a + b \cdot \theta, \tag{1}$$

where a controls the distance from the starting point of the spiral to the origin, b controls the distance between loops, and θ is the angle of rotation of the spiral. The distance between each loop is $2\pi b$.

3.2 Relation Patterns

Let $\mathcal{E}$ denote the set of entities, $\mathcal{R}$ denote the set of relations, and $\mathcal{T}$ denote the set of the timestamp. Given a temporal knowledge graph $\mathcal{G}$, it can be defined as a collection of quadruples (s, r, o, τ), where $s \in \mathcal{E}$, $r \in \mathcal{R}$, $o \in \mathcal{E}$ and $\tau \in \mathcal{T}$ denote the subject entity, relation, object entity and timestamp, respectively.

As previous studies (Sun et al., 2019; Chen et al., 2022) highlighted, TKGE has focused on several key relations patterns, including:

Definition 1. *A relation r is symmetric, if* $\forall s, o, \tau$, $r(s, o, \tau) \wedge r(o, s, \tau)$ *holds True.*

Definition 2. *A relation r is asymmetric, if* $\forall s, o, \tau$, $r(s, o, \tau) \wedge \neg r(o, s, \tau)$ *holds True.*

Definition 3. *Relation r_1 is the inverse of r_2, if* $\forall s, o, \tau$, $r_1(s, o, \tau) \wedge r_2(o, s, \tau)$ *holds True.*

Definition 4. *Relation r_1 and r_2 are evolving over time from timestamp τ_1 to timestamp τ_2, if* $\forall s, o, \tau$, $r_1(s, o, \tau_1) \wedge r_2(s, o, \tau_2)$ *holds True.*

4 Methodology

4.1 TeAST Model

In this section, we introduce the novel TeAST model, which represents the relations on Archimedean spiral timelines. Since many previous works (Trouillon et al., 2016; Sun et al., 2019; Lacroix et al., 2020; Xu et al., 2020a) have demonstrated that encoding knowledge graphs in complex space can better capture potential links between entities, we also model TKGs in the complex space. For a quadruple (s, r, o, τ), we also use $\boldsymbol{e}_s$, $\boldsymbol{e}_r$, $\boldsymbol{e}_o$ and $\boldsymbol{e}_\tau$ to denote the subject embedding, relation embedding, object embedding and timestamp embedding respectively in the complex space. We have

$$\begin{aligned} \boldsymbol{e}_s = Re(s) + iIm(s), \boldsymbol{e}_r = Re(r) + iIm(r), \\ \boldsymbol{e}_o = Re(o) + iIm(o), \boldsymbol{e}_\tau = Re(\tau) + iIm(\tau), \end{aligned} \tag{2}$$

where $\boldsymbol{e}_s, \boldsymbol{e}_r, \boldsymbol{e}_o, \boldsymbol{e}_\tau \in \mathbb{C}^k$, and $Re(*)$ is the real vector component and $Im(*)$ is an imaginary vector component.

We first map relations onto the corresponding Archimedean spiral timeline. Specifically, we regard each relation as different the angle of rotation θ in Eq. 1, and regard each timestamp as distance control parameter b in Eq. 1. Therefore, the range of embedding values for each relation is $\boldsymbol{e}_r \in (0, 2\pi)$. To prevent crossover between spirals, we set the starting point of all spirals to the origin. That is, we set $a = 0$ for TeAST in Eq. 1. On this basis, we map all relations to the matching spiral timeline, denoted as:

$$\boldsymbol{\xi}_{(\tau,r)} = \boldsymbol{e}_\tau \circ \boldsymbol{e}_r, \tag{3}$$

where $\circ$ denotes the Hadamard product. Since TeAST is modeled in complex space, we employ the Hadamard product to do spiral timeline mapping for the relations accordingly. Further, we have

$$\begin{aligned}\boldsymbol{\xi}_{(\tau,r)} &= Re(\tau)Re(r) - Im(\tau)Im(r)\\ &+iRe(\tau)Im(r) + iIm(\tau)Re(r),\end{aligned} \tag{4}$$

where $Re(r) \in (0, 2\pi)$ and $Im(r) \in (0, 2\pi)$. All relation embeddings are all constrained between 0 and 2π. This ensures that the relations can be effectively mapped to the corresponding spiral timelines.

Following previous tensor factorization models (Trouillon et al., 2016; Lacroix et al., 2020), the score function of TeAST is denoted as:

$$\phi(s, r, o, \tau) = Re(< \boldsymbol{e}_s, \boldsymbol{\xi}_{(\tau,r)}, \bar{\boldsymbol{e}}_o >). \tag{5}$$

Then, we optimize the graph embeddings through the score function.

Furthermore, since Archimedean spiral is based on the polar coordinate system, we can regard $\boldsymbol{\xi}_{(\tau,r)}$ as a modulus part. During the model training process, we note that there are inevitably equal modulus cases on different spiral timelines, leading to confusion between semantic relations. Therefore, we employ timestamp phase information $\boldsymbol{e}'_\tau = Re(\tau') + iIm(\tau')$ to avoid the bad cases, where $Re(\tau'), Im(\tau') \in \mathbb{R}^{\frac{k}{2}}$. Additionally, we use absolute values to constrain the temporal phase information to be isotropic over time. This is done to enforce consistency and avoid any directional bias. As phases have periodic characteristics, we employ a sine function to measure the timestamp phase embeddings similar to HAKE (Zhang et al., 2020). Combining the modulus part and the phase part, we get

$$\begin{aligned}\boldsymbol{\xi}'_{(\tau,r)} &= (Re(\tau)Re(r) + \sin(Re(\tau')))\\ &-(Im(\tau)Im(r) + \sin(Im(\tau')))\\ &+i(Re(\tau)Im(r) + \sin(Re(\tau')))\\ &+i(Im(\tau)Re(r) + \sin(Im(\tau'))).\end{aligned} \tag{6}$$

The improved score function of TeAST is given by

$$\phi(s, r, o, \tau) = Re(< \boldsymbol{e}_s, \boldsymbol{\xi}'_{(\tau,r)}, \bar{\boldsymbol{e}}_o >). \tag{7}$$

It is worth noting that the number of parameters of TeAST increases linearly with embedding dimension k. Hence, the space complexity of TeAST model is $O(k)$, similar to TNTComplEx (Lacroix et al., 2020). In addition, we calculate the score function of TeAST with Hadamard product between k-dimensional complex vector embeddings as TNTComplEx. The time complexity of TeAST and TNTComplEx equals to $O(k)$.

4.2 Loss Function

Following TNTComplEx (Lacroix et al., 2020) and TeLM (Xu et al., 2021), we use reciprocal learning to simplify the training process, and the loss function is defined as follows:

$$\begin{aligned}\mathcal{L}_\mu &= -\log\left(\frac{\exp(\phi(s, r, o, \tau))}{\sum_{s'\in\mathcal{E}} \exp(\phi(s', r, o, \tau))}\right)\\ &-\log\left(\frac{\exp(\phi(o, r^{-1}, s, \tau))}{\sum_{o'\in\mathcal{E}} \exp(\phi(o', r^{-1}, s, \tau))}\right)\\ &+\lambda_\mu \sum_{i=1}^{k}(\|\boldsymbol{e}_s\|_3^3 + \|\boldsymbol{\xi}'_{(\tau,r)}\|_3^3 + \|\boldsymbol{e}_o\|_3^3),\end{aligned} \tag{8}$$

where λ_μ denotes N3 regularization weight and r^{-1} is the inverse relation. According to several studies, N3 regularization improves the performance of the KGE models (Lacroix et al., 2018; Xu et al., 2020b) and TKGE models (Lacroix et al., 2020; Xu et al., 2021) based on tensor factorization.

4.3 Temporal Regularization

The temporal regularization can constrain the temporal embedding information and thus better model TKGs. TNTComplEx (Lacroix et al., 2020) expects neighboring timestamps to have close representations. Hence, the smoothing temporal regularizer is defined as:

$$\Lambda^3 = \frac{1}{N_\tau - 1} \sum_{i=1}^{N_\tau - 1} \|\boldsymbol{e}_{\tau(i+1)} - \boldsymbol{e}_{\tau(i)}\|_3^3, \tag{9}$$

where N_τ is the number of time steps.

Recently, TeLM (Xu et al., 2021) introduces the linear temporal regularizer by adding a bias component between the neighboring temporal embeddings, which can be defined as:

$$\Omega^3 = \frac{1}{N_\tau - 1} \sum_{i=1}^{N_\tau - 1} \|\boldsymbol{e}_{\tau(i+1)} - \boldsymbol{e}_{\tau(i)} - \boldsymbol{e}_b\|_3^3, \tag{10}$$

where $\boldsymbol{e}_b$ denotes the randomly initialized biased embedding, which is then learned from the training process.

In this work, we employ the Archimedean spiral to model TKGs. The previous temporal regularization methods expect the adjacent timestamps to be close to each other. For our model TeAST, this leads to the spiral timeline overlapping scenarios. To avoid these bad scenarios, we develop a novel temporal spiral regularizer by adding the phase timestamp embedding $\boldsymbol{e}'_\tau$ to the smoothing temporal regularizer. The temporal regularization function is defined as:

$$\mathcal{L}_\tau{}^3 = \frac{1}{N_\tau - 1} \sum_{i=1}^{N_\tau - 1} \|(\boldsymbol{e}_{\tau(i+1)} - \boldsymbol{e}_{\tau(i)}) + (\boldsymbol{e}'_{\tau(i+1)} - \boldsymbol{e}'_{\tau(i)})\|_3^3. \tag{11}$$

The total loss function of TeAST is defined as:

$$\mathcal{L} = \mathcal{L}_\mu + \lambda_\tau \mathcal{L}_\tau{}^3, \tag{12}$$

where λ_τ is the weight of the temporal regularizer.

4.4 Modeling Various Relation Patterns

TeAST can model important relation patterns, including symmetric, asymmetric, inverse and temporal evolution patterns. We list all the propositions here and provide the proofs in Appendix.

Proposition 1. *TeAST can model the symmetric relation pattern. (See proof in Appendix A)*

Proposition 2. *TeAST can model the asymmetric relation pattern. (See proof in Appendix B)*

Proposition 3. *TeAST can model the inverse relation pattern. (See proof in Appendix C)*

Proposition 4. *TeAST can model the temporal evolution pattern. (See proof in Appendix D)*

5 Experiments

5.1 Datasets

We evaluate TeAST on three TKGE benchmark datasets. **ICEWS14** and **ICEWS05-15** (García-Durán et al., 2018) are both extracted from the *Integrated Crisis Early Warning System (ICEWS)* dataset (Lautenschlager et al., 2015), which consists of temporal sociopolitical facts starting from 1995. ICEWS14 consists of sociopolitical events in 2014 and ICEWS05-15 involves events occurring from 2005 to 2015. **GDELT** is a subset of the larger *Global Database of Events, Language, and Tone (GDELT)* TKG dataset (Leetaru and Schrodt, 2013). The GDELT contains facts with daily timestamps between April 1, 2015 and March 31, 2016, and only contains 500 most common entities and 20 most frequent relations. It is worth noting that GDELT holds a large number of quadruples (2M) but does not describe enough entities (500). Hence, The GDELT requires a strong temporal inductive capacity.

	ICEWS14	**ICEWS05-15**	**GDELT**
$\mathcal{E}$	7,128	10,488	500
$\mathcal{R}$	230	251	20
$\mathcal{T}$	365	4017	366
#Train	72,826	386,962	2,735,685
#Vaild	8,963	46,092	341,961
#Test	8,941	46,275	341,961
Timespan	1 year	11 years	1 year
Granularity	Daily	Daily	Daily

Table 1: Statistics of TKGE datasets in the experiment.

5.2 Evaluation Protocol

In this paper, we evaluate our TKGE model using the benchmarks mentioned above. Following the strong baselines (Lacroix et al., 2020; Xu et al., 2021; Chen et al., 2022), the quality of the ranking of each test triplet is evaluated by calculating all possible substitutions of subject entity and object entity: (s', r, o, τ) and (s, r, o', τ), where s', $o' \in \mathcal{E}$. And then, we sort the score of candidate quadruples under the timewise filtered settings (Lacroix et al., 2020; Xu et al., 2021; Chen et al., 2022). The performance is evaluated using standard evaluation metrics, including Mean Reciprocal Rank (MRR) and Hits@n. Hits@n measures the percentage of correct entities in the top n predictions. Higher values of MRR and Hits@n indicate better performance. Hits ratio with cut-off values $n = 1, 3, 10$. In this paper, we utilize H@n to denote Hits@n for convenience.

5.3 Baselines

We compare our model with the state-of-the-art TKGE models, including TTransE (Leblay and Chekol, 2018), DE-SimplE (Goel et al., 2020), TA-DistMult (García-Durán et al., 2018), ChronoR (Sadeghian et al., 2021), TComplEx (Lacroix et al., 2020), TNTComplEx (Lacroix et al., 2020), TeLM (Xu et al., 2021), BoxTE (Messner et al., 2022) and RotateQVS (Chen et al., 2022).

	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
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-DistMult	0.477	0.363	-	0.686	0.474	0.346	-	0.728	0.206	0.124	0.219	0.365
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	0.800	0.340	0.249	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
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
TeAST(ours)	**0.637**	**0.560**	**0.682**	**0.782**	**0.683**	**0.604**	**0.732**	**0.829**	**0.371**	**0.283**	**0.401**	**0.544**

Table 2: Link prediction results on ICEWS14, ICEWS05-15 and GDELT. All results are taken from the original papers. Results of ♡ are the best results reported in the original papers. They are ChronoR (k=2), TComplEx (x10), TNTComplEx (x10) and BoxTE (k=5), respectively. Dashes: results are not reported in the responding literature.

Note that TComplEx and TNTComplEx are also based on tensor factorization TKGE methods in the complex space, and thus we consider TComplEx and TNTComplEx as the main baselines. Furthermore, TeLM performs multivector tensor factorization for a TKG. Hence, TeLM has twice the space complexity of TeAST, TComplEx and TNTComplEx. Among the existing TKGE methods, TeLM obtains SOTA results on ICEWS14 and ICEWS05-15 and BoxTE achieves SOTA results on GDELT dataset.

5.4 Experimental Setup

We implement our proposed model TeAST via pytorch based on TNTComplEx (Lacroix et al., 2020) training framework[1]. All experiments are trained on a single NVIDIA Tesla V100 with 32GB memory. We use Adagrad (Duchi et al., 2011) optimizer and employ grid search to find the best hyperparameters based on the performance on the validation datasets. The learning rate is set to 0.1 and the embedding dimension k is set to 2000 in all cases. The best models are selected by early stopping on the validation datasets, and the max epoch is 200. The optimal hyperparameters for TeAST are as follows:

- **ICEWS14:** $\lambda_\mu = 0.0025, \lambda_\tau = 0.01$
- **ICEWS05-15:** $\lambda_\mu = 0.002, \lambda_\tau = 0.1$
- **GDELT:** $\lambda_\mu = 0.003, \lambda_\tau = 0.003$

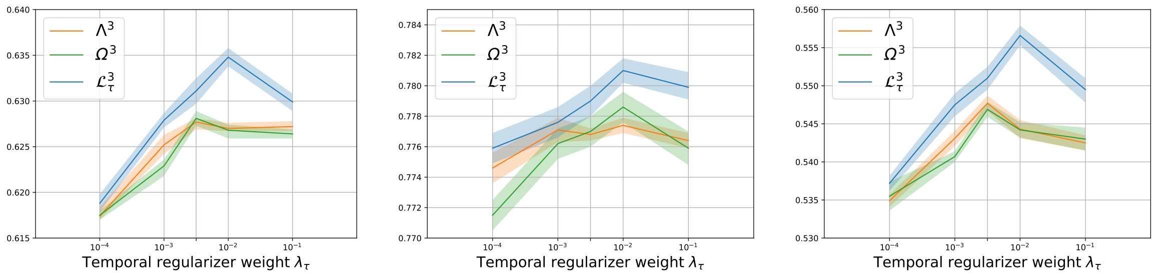
We report the average results on the test set for five runs. We omit the variance as it is generally low. The training processes of TeAST on ICEWS14, ICEWS05-15 and GDELT cost less than half an hour, less than an hour and five hours, respectively.

6 Results and Analysis

6.1 Main Results

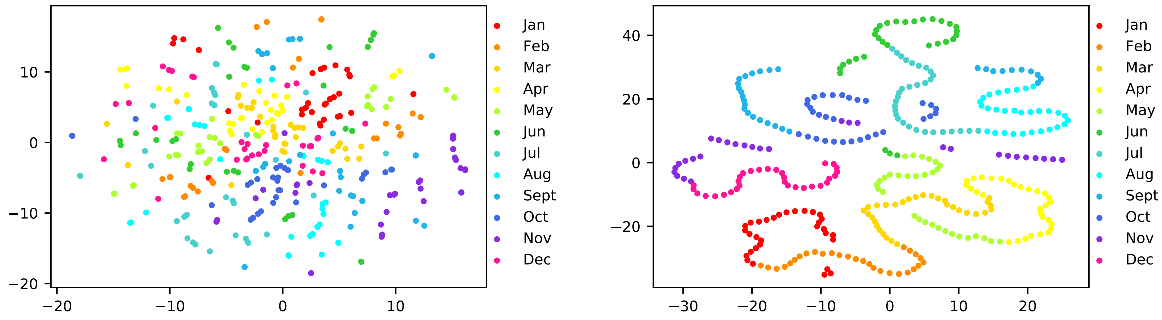
The link prediction results on ICEWS14, ICEWS05-15 and GDELT are shown in Table 2. We observe that TeAST surpasses all baselines on ICEWS14, ICEWS05-15 and GDELT regarding all metrics. Since TeAST employs the temporal Archimedean spiral to encode relation embeddings, this allows relations that occur at the same moment to be mapped onto the same spiral timeline and all relations evolve over time. It builds a close connection between the relation and timestamp and avoids incorporating temporal information into the entities for TKG. It proves that mapping the relations to Archimedean spiral timeline is an effective way to learn graph embeddings. TeAST can better encode temporal knowledge graphs and captures the latent information between subject entities and object entities. Meanwhile, the temporal spiral regularizer in TeAST avoids spiral timeline overlapping scenarios and further improves the performance. BoxTE (Messner et al., 2022) has shown that GDELT requires a high level of temporal inductive capacity for effective encoding. This is because GDELT exhibits a significant degree of temporal variability, with some facts lasting across multiple consecutive time stamps while others are momentary and sparse.

[1] https://github.com/facebookresearch/tkbc



(a) MRR (b) Hits@10 (c) Hits@1

Figure 3: Link prediction results of TeAST trained with different temporal regularizers on ICEWS14.



(a) No Temporal Regularizer (b) Temporal Spiral Regularizer $\mathcal{L}_\tau{}^3$

Figure 4: Visualisations of the learned timestamp embeddings on ICEWS14. (a) not used the temporal regularizer and (b) employs the temporal spiral regularizer. Different colors indicate different months.

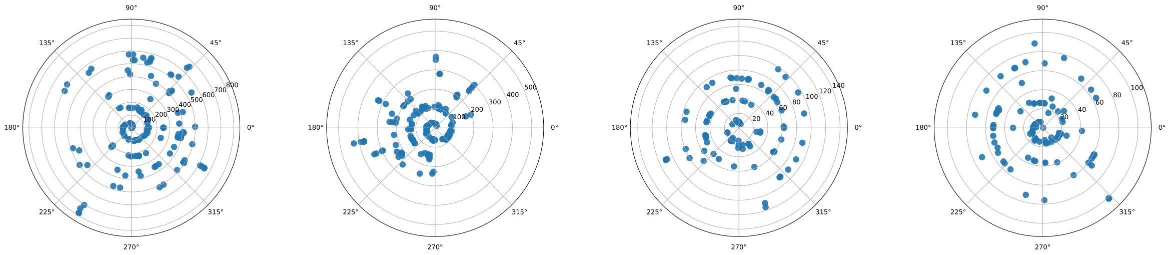
In comparison to the SOTA method BoxTE on GDELT, TeAST achieves superior results on all metrics.

6.2 Effect of Temporal Regularizer

We study the effect of temporal regularization on ICEWS14, and compare the performance of TeAST with the previously proposed temporal regularizers, including the smoothing temporal regularizer Λ^3 in Eq. 9, the linear temporal regularizer Ω^3 in Eq. 10 and our proposed temporal spiral regularizer $\mathcal{L}_\tau{}^3$ in Eq. 11. We set the temporal regularization weight $\lambda_\tau \in \{0.0001, 0.001, 0.005, 0.01, 0.1\}$. Detailed results of the effect of temporal regularization on ICEWS14 are given in Figure 3. The blue line denotes the temporal spiral regularizer. Compared with the previously proposed temporal regularizers, the temporal spiral regularizer improved MRR by 0.8 points, Hits@10 by 0.3 points, and Hits@1 by 1.2 points, respectively. Since the temporal spiral regularizer adds a phase timestamp embedding to avoid the overlap of Archimedean spiral timelines and thus can better discriminate timestamp information.

Furthermore, we utilize t-SNE (Van der Maaten and Hinton, 2008) to visualize the trained timestamp embeddings of TeAST, which with and without the temporal spiral regularizer. The visualization results are shown in Figure 4. We observe that the distribution of adjacent temporal embeddings of TeAST without temporal spiral regularization trained is scattered. There are only a few months that come together, such as January, October and November. In addition, we observe some overlapping scenarios of the learned time embeddings, suggesting that the learned time embedding is not inaccurate. It will further hinder the effectiveness of learning the facts associated with a specific timestamp.

On the contrary, using the temporal spiral regularizer in TeAST can learn time embedding information effectively, resulting in orderly time clusters. This demonstrates the effectiveness of the temporal spiral regularizer in improving the ability of the model to accurately capture and retain information about specific timestamps. In addition,



(a) 2014-1-2 (b) 2014-3-1 (c) 2014-4-6 (d) 2014-5-3

Figure 5: Visualisations of the learned relation embeddings are mapped the corresponding timelines from ICEWS14.

			ICEWS14				ICEWS05-15			
Mapping Entity	Mapping Relation	Phase	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
✔			0.598	0.531	0.649	0.749	0.639	0.542	0.710	0.798
✔		✔	0.611	0.542	0.658	0.752	0.651	0.556	0.727	0.800
	✔		0.621	0.545	0.665	0.763	0.671	0.589	0.722	0.812
	✔	✔	**0.637**	**0.560**	**0.682**	**0.782**	**0.683**	**0.604**	**0.732**	**0.829**

Table 3: Ablation results on ICEWS14 and ICEWS05-15. Mapping Entity: Projecting entities onto the corresponding spiral timeline. Mapping Relation: Projecting relations onto the corresponding spiral timeline.

we notice a very interesting phenomenon: TeAST also learned deep information about the order between months with the temporal spiral regularizer and the temporal embedding of the same month presented on the same line. The results further suggest a good fit with our initial motivation that each relation should be mapped onto a temporal spiral and the relations with the same timestamp should be on the same timeline.

6.3 Analysis on Relation Embeddings

As for TeAST, we employ the Archimedean spiral to map relations into the polar coordinate system. Therefore, we map the learned relation embedding of the same time to the corresponding timeline in the polar coordinate system. The results are shown in Figure 5. The mapping algorithm is based on the implementation of Eq. 3. The Figure 5 shows the relation embedding projection for four different times. We can see that the relation embeddings of the same timestamp are fitted as an Archimedean spiral timeline. This is further evidence that TeAST can effectively encode relations onto the corresponding spiral timeline.

6.4 Ablation Studies

In this part, we conduct ablation studies on mapping entities and mapping relations of TeAST and the phase item. Table 3 shows the results on ICEWS14 and ICEWS05-15 benchmark datasets. The results of the comparison of mapping entities and mapping relations on the spiral timeline indicate that mapping relations on the spiral timeline is more effective than mapping entities on the spiral timeline for TeAST. This is further proof that the design motivation of TeAST is the meanings of the entities in quadruples do not change as time evolves, while the relations between entities change in TKGs. In addition, we also observe that TeAST achieves better link prediction results with phase vectors, because it can well distinguish relations at the same level of semantic hierarchy. It is worth noting that TeAST also obtains better or more competitive results without phase vectors than TComplEx and TNTComplEx on ICEWS14 and ICEWS05-15. The results show that TeAST maps relations on the corresponding Archimedean spiral timelines, which can effectively model temporal knowledge graphs.

7 Conclusion

This paper proposes a novel and interesting TKGE method TeAST, which maps relations onto the corresponding Archimedean spiral timeline. The experimental results fully illustrate that TeAST can better model TKG than previous methods and learn the relation information over time. We also provide formal mathematical proofs to demonstrate that TeAST can encode the key relation patterns. In addition, the temporal spiral regularizer learns the latent information about the order between months better and improves the link prediction performances. This work will hopefully stimulate further research on TKGE models and provide a novel perspective on the subject.

Limitations

As previously mentioned, TeAST maps relations onto the corresponding Archimedean spiral timeline and transforms the quadruples completion to 3th-order tensor factorization. It is required to store the values and this slightly increase the space requirement and training time in the embedding learning process. Among all the baselines, TComplEx, TNTComplEx and TeLM are all tensor factorization based models. Table 4 compares training time and space requirement between our model and baselines on ICEWS14. TComplEx is the smallest model and takes the minimum training time. Compared with TComplEx, our model is about 4.6% bigger than TComplEx, and takes 21.4% more training time.

	ICEWS14		
Method	#Params(M)	#Train-time	MRR
TComplEx	**31.81**	**14 min**	0.610
TNTComplEx	32.65	16 min	0.620
TeLM	63.63	19 min	0.625
TeAST(ours)	33.28	17 min	**0.637**

Table 4: Comparison with existing TKGE models based on tensor factorisation. All experiments are trained on a single NVIDIA Tesla V100 with 32GB memory.

Acknowledgement

This work was funded by National Natural Science Foundation of China (Grant No. 61762069), Key Technology Research Program of Inner Mongolia Autonomous Region (Grant No. 2021GG0165), Key R&D and Achievement Transformation Program of Inner Mongolia Autonomous Region (Grant No. 2022YFHH0077), The Central Government Fund for Promoting Local Scientific and Technological Development (Grant No. 2022ZY0198), Big Data Lab of Inner Mongolia Discipline Inspection and Supervision Committee (Grant No. 21500-5206043).

References

Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5843–5857. Association for Computational Linguistics.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2019. Bidirectional attentive memory networks for question answering over knowledge bases. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2913–2923. Association for Computational Linguistics.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12:2121–2159.

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4816–4821. Association for Computational Linguistics.

Rishab Goel, Seyed Mehran Kazemi, Marcus A. Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3988–3995. AAAI Press.

Zikun Hu, Yixin Cao, Lifu Huang, and Tat-Seng Chua. 2021. How knowledge graph and attention help? A qualitative analysis into bag-level relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages

4662–4671. Association for Computational Linguistics.

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 of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 687–696. The Association for Computer Linguistics.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514.

Ademar Crotti Junior, Fabrizio Orlandi, Damien Graux, Murhaf Hossari, Declan O'Sullivan, Christian Hartz, and Christian Dirschl. 2020. Knowledge graph-based legal search over german court cases. In *The Semantic Web: ESWC 2020 Satellite Events - ESWC 2020 Satellite Events, Heraklion, Crete, Greece, May 31 - June 4, 2020, Revised Selected Papers*, volume 12124 of *Lecture Notes in Computer Science*, pages 293–297. Springer.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4289–4300.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2869–2878. PMLR.

Jennifer Lautenschlager, Steve Shellman, and Michael Ward. 2015. Icews event aggregations.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1771–1776. ACM.

Kalev Leetaru and Philip A Schrodt. 2013. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, volume 2, pages 1–49. Citeseer.

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 Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2181–2187. AAAI Press.

Johannes Messner, Ralph Abboud, and İsmail İlkan Ceylan. 2022. Temporal knowledge graph completion using box embeddings. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 7779–7787. AAAI Press.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 809–816. Omnipress.

Ali Sadeghian, Mohammadreza Armandpour, Anthony Colas, and Daisy Zhe Wang. 2021. Chronor: Rotation based temporal knowledge graph embedding. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6471–6479. AAAI Press.

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.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 1112–1119. AAAI Press.

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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2569–2578. Association for Computational Linguistics.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020a. Tero: A time-aware knowledge graph embedding via temporal rotation. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1583–1593. International Committee on Computational Linguistics.

Chengjin Xu, Mojtaba Nayyeri, Yung-Yu Chen, and Jens Lehmann. 2020b. Knowledge graph embeddings in geometric algebras. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 530–544. International Committee on Computational Linguistics.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2731–2741.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3065–3072. AAAI Press.

A Proof of Propositions 1

The score function of TeAST is defined as:

$$\begin{aligned}
\phi(s,r,o,\tau) &= Re(<\boldsymbol{e}_s, \boldsymbol{\xi}'_{(\tau,r)}, \bar{\boldsymbol{e}}_o >) \\
&= Re(\sum_{k=1}^{K} \boldsymbol{e}_{sk} \boldsymbol{\xi}'_{(\tau,r)k} \bar{\boldsymbol{e}}_{ok}) \\
&=< Re(\boldsymbol{e}_s), Re(\boldsymbol{\xi}'_{(\tau,r)}), Re(\boldsymbol{e}_o) > \\
&+ < Im(\boldsymbol{e}_s), Re(\boldsymbol{\xi}'_{(\tau,r)}), Im(\boldsymbol{e}_o) > \\
&+ < Re(\boldsymbol{e}_s), Im(\boldsymbol{\xi}'_{(\tau,r)}), Im(\boldsymbol{e}_o) > \\
&- < Im(\boldsymbol{e}_s), Im(\boldsymbol{\xi}'_{(\tau,r)}), Re(\boldsymbol{e}_o) > .
\end{aligned} \tag{13}$$

Following ComplEx (Trouillon et al., 2016), we employ the standard componentwise multilinear dot product $< a, b, c >:= \sum_k a_k b_k c_k$ in Eq. 13. For symmetric pattern, we have $r(s,o,\tau) \wedge r(o,s,\tau)$ according to Definition 1. Hence, we get

$$\phi(s,r,o,\tau) = \phi(o,r,s,\tau). \tag{14}$$

One can easily check that Eq. 14 meet the symmetric pattern conditions when $\boldsymbol{\xi}'_{(\tau,r)}$ is real (i.e. its imaginary part is zero). We have

$$\begin{aligned}
\phi(s,r,o,\tau) &= < Re(\boldsymbol{e}_s), Re(\boldsymbol{\xi}'_{(\tau,r)}), Re(\boldsymbol{e}_o) > \\
&+ < Im(\boldsymbol{e}_s), Re(\boldsymbol{\xi}'_{(\tau,r)}), Im(\boldsymbol{e}_o) > \\
&= < Re(\boldsymbol{e}_o), Re(\boldsymbol{\xi}'_{(\tau,r)}), Re(\boldsymbol{e}_s) > \\
&+ < Im(\boldsymbol{e}_o), Re(\boldsymbol{\xi}'_{(\tau,r)}), Im(\boldsymbol{e}_s) > \\
&= \phi(o,r,s,\tau).
\end{aligned} \tag{15}$$

Therefore, a sufficient necessary condition for TeAST to be able to model symmetric pattern is $Im(\boldsymbol{\xi}'_{(\tau,r)}) = 0$.

B Proof of Propositions 2

For asymmetric pattern, we have $r(s,o,\tau) \wedge \neg r(o,s,\tau)$ according to Definition 2. Hence, we get

$$\phi(s,r,o,\tau) \neq \phi(o,r,s,\tau). \tag{16}$$

One can easily check that Eq. 16 meet the asymmetric pattern conditions when $\boldsymbol{\xi}'_{(\tau,r)}$ is purely imaginary (i.e. its real part is zero). We have

$$\begin{aligned} \phi(s,r,o,\tau) &=< Re(\boldsymbol{e}_s), Im(\boldsymbol{\xi}'_{(\tau,r)}), Im(\boldsymbol{e}_o) > \\ &\quad - < Im(\boldsymbol{e}_s), Im(\boldsymbol{\xi}'_{(\tau,r)}), Re(\boldsymbol{e}_o) >, \\ \phi(o,r,s,\tau) &=< Re(\boldsymbol{e}_o), Im(\boldsymbol{\xi}'_{(\tau,r)}), Im(\boldsymbol{e}_s) > \\ &\quad - < Im(\boldsymbol{e}_o), Im(\boldsymbol{\xi}'_{(\tau,r)}), Re(\boldsymbol{e}_s) > . \end{aligned} \tag{17}$$

We can get $\phi(s,r,o,\tau) \neq \phi(o,r,s,\tau)$. Therefore, a sufficient necessary condition for TeAST to be able to model asymmetric pattern is $Re(\boldsymbol{\xi}'_{(\tau,r)}) = 0$.

C Proof of Propositions 3

For inverse pattern, we have $r_1(s,o,\tau) \wedge r_2(o,s,\tau)$ according to Definition 3. Hence, we get

$$\begin{aligned} \phi(s,r_1,o,\tau) = \phi(o,r_2,s,\tau) \Leftrightarrow \\ \boldsymbol{e}_{r1} = \bar{\boldsymbol{e}}_{r2} \Leftrightarrow \\ Re(r_1) + Re(r_2) = 0 \wedge Im(r_1) - Im(r_2) = 0, \end{aligned} \tag{18}$$

where $\bar{\boldsymbol{e}}_{r2}$ is the conjugate of $\boldsymbol{e}_{r1}$.

D Proof of Propositions 4

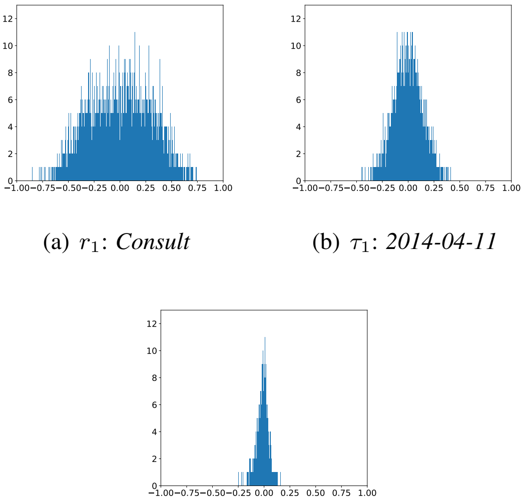
For temporal evolution pattern, we have $r_1(s,o,\tau_1) \wedge r_2(s,o,\tau_2)$ according to Definition 4. Hence, we have

$$\begin{aligned} \phi(s,r_1,o,\tau_1) = \phi(s,r_2,o,\tau_2) \Leftrightarrow \\ \boldsymbol{\xi}'_{(\tau_1,r_1)} = \boldsymbol{\xi}'_{(\tau_2,r_2)}. \end{aligned} \tag{19}$$

It is worth noting that $\boldsymbol{\xi}'_{(\tau_1,r_1)} = \boldsymbol{\xi}'_{(\tau_2,r_2)}$ just means the values of their modulus part add phase part are equal. The relations at the same time are mapped on the corresponding Archimedean spiral timeline in the polar spatial representation.

E Analysis and Case Study for Several Key Relation Patterns

To illustrate the learned relation patterns that contain symmetric, asymmetric, inverse and temporal evolution patterns, we visualize some examples by visualizing the histograms of the learned embeddings. All cases are from ICEWS14 dataset (García-Durán et al., 2018).



(a) r_1: *Consult*

(b) τ_1: *2014-04-11*

(c) $Im(\boldsymbol{\xi}'_{(\tau_1,r_1)})$

Figure 6: The histograms of learned embeddings for symmetric pattern. Two existing facts are *(Kazakhstan, Consult, Afghanistan, 2014-04-11)* and *(Afghanistan, Consult, Kazakhstan, 2014-04-11)* and *Consult* is a symmetric relation.

E.1 Symmetric Pattern

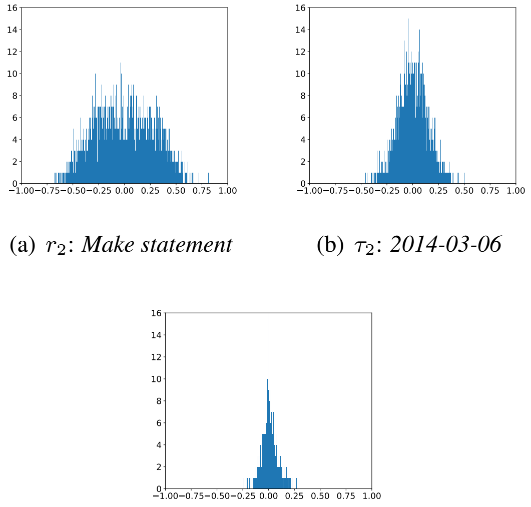
As shown the proof of Propositions 1 (see Appendix A), TeAST can encode symmetric pattern when $Im(\boldsymbol{\xi}'_{(\tau,r)}) = 0$ is satisfied. As shown in Figure 6, tow facts *(Kazakhstan, Consult, Afghanistan, 2014-04-11)* and *(Afghanistan, Consult, Kazakhstan, 2014-04-11)* from ICEWS14, and *Consult* is a symmetric relation. We observe that the learned $Im(\boldsymbol{\xi}'_{(\tau_1,r_1)})$ in Figure 6(c) is close to 0. The result demonstrates that TeAST can model the symmetric pattern.

E.2 Asymmetric Pattern

Opposite to symmetric pattern, TeAST can encode asymmetric pattern when $Re(\boldsymbol{\xi}'_{(\tau,r)}) = 0$ is satisfied. Figure 7 shows an example of asymmetric pattern and *Make statement* is taken an asymmetric relation. Figure 7(c) shows that our TeAST can model the asymmetric pattern.

E.3 Inverse Pattern

As shown the proof of Propositions 3 (see Appendix C), if r_4 is the inverse of the r_3, and we have $Re(r_3) + Re(r_4) = 0 \wedge Im(r_3) - Im(r_4) = 0$. Two existing facts *(Iraq, Host a visit, Nuri al-Maliki, 2014-06-13)* and *(Nuri al-Maliki, Make*



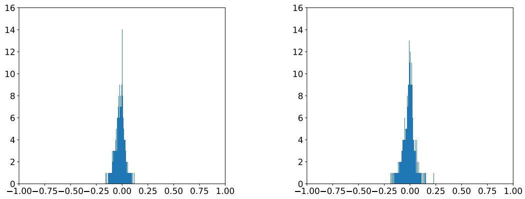
(a) r_2: *Make statement*

(b) τ_2: *2014-03-06*

(c) $Re(\boldsymbol{\xi}'_{(\tau_2,r_2)})$

Figure 7: The histograms of learned embeddings for asymmetric pattern. A existing fact is *Ministry (Afghanistan), Make statement, Impose restrictions on political freedoms, 2014-03-06)* and *Make statement* is a asymmetric relation.

a visit, Iraq, 2014-06-13) from ICEWS14, which the relation *Host a visit* is the inverse of the relation *Make a visit*. Figure 8 shows that TeAST satisfies the above conditions.



(a) $Re(r_3) + Re(r_4)$

(b) $Im(r_3) - Im(r_4)$

Figure 8: The histograms of learned embeddings for inverse pattern. Two existing fact is *(Iraq, Host a visit, Nuri al-Maliki, 2014-06-13)* and *(Nuri al-Maliki, Make a visit, Iraq, 2014-06-13)*. Inverse relation pair: *Host a visit* vs. *Make a visit*.

E.4 Temporal Evolution Pattern

As shown in Proof of Propositions 4 (see Appendix D), if a relation r_5 and a relation r_6 are evolving over time from τ_5 from τ_6, we have $\boldsymbol{\xi}'_{(\tau_5,r_5)} = \boldsymbol{\xi}'_{(\tau_6,r_6)}$. To verify that TeAST can model the temporal evolution pattern, we randomly select five facts, including *(Nuri al-Maliki, Make a visit, Iraq, 2014-06-13)*, *(Nuri al-Maliki, Consult, Iraq, 2014-06-23)*, *(Nuri al-Maliki, Make statement, Iraq, 2014-06-29)*, *(Nuri al-Maliki, Mobilize or increase police power, Iraq, 2014-08-11)* and *(Nuri al-Maliki, Praise or endorse, Iraq, 2014-11-10)*. The five quadruples above belong to the temporal evaluation pattern. As shown in Figure 9, we mutually calculate the cosine similarity between $\boldsymbol{\xi}'_{(\tau_i,r_i)}$ of the five quadruples. We can observe that the $\boldsymbol{\xi}'_{(\tau_i,r_i)}$ of the corresponding quadruples are all close. Results further demonstrate that TeAST can effectively model the temporal evolution pattern.

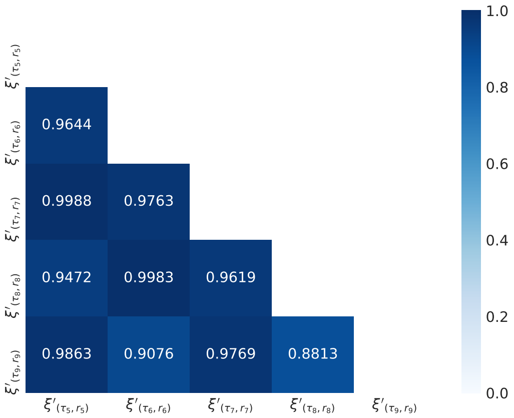


Figure 9: The cosine similarity of $\boldsymbol{\xi}'_{(\tau_i,r_i)}$ among five quadruples. Five existing fact are *(Nuri al-Maliki, Make a visit, Iraq, 2014-06-13)*, *(Nuri al-Maliki, Consult, Iraq, 2014-06-23)*, *(Nuri al-Maliki, Make statement, Iraq, 2014-06-29)*, *(Nuri al-Maliki, Mobilize or increase police power, Iraq, 2014-08-11)* and *(Nuri al-Maliki, Praise or endorse, Iraq, 2014-11-10)*, respectively.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
8

□ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

1

☑ B1. Did you cite the creators of artifacts you used?
5

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
The code repository is governed by Apache-2.0 license.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☒ Did you run computational experiments?

Left blank.

□ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
No response.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

□ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
No response.

□ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
No response.

□ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
No response.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.