TACE: TIME-AWARE CONVOLUTIONAL EMBEDDING LEARNING FOR TEMPORAL KNOWLEDGE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Temporal knowledge graph completion (TKGC) is a challenging task to infer the missing component for quadruples. The key challenge lies at how to integrate time information into the embeddings of entities and relations. Recent TKGC methods tend to capture temporal patterns via linear or multilinear models, which are fast but not expressive enough. In this study, we propose a novel time-aware convolutional embedding model (TaCE) to represent the time-dependent facts in the task of TKGC. It highlights its novelty to feasibly convert timestamps as temporal convolutional filters to fully interact with entities and relations and learn temporal patterns in knowledge graphs (KGs). An extensive comparison proves that our model outperforms the state-of-the-art models on three public benchmark datasets of ICEWS14, ICEWS05-15 and GDELT. Results also demonstrate good temporal expressiveness and computation efficiency performed by our TaCE.

1 INTRODUCTION

Knowledge graphs (KGs), storing facts in tuples, are often faced with incompletion. To solve this problem, knowledge graph embedding (KGE), mapping the entities and relations into a continuous vector space, has been developed to capture the semantic meanings for the task of KG reasoning (KGR) or KG completion (KGC) (Bordes et al., 2013; Trouillon et al., 2016; Sun et al., 2019). Both KGR and KGC aim at inferring the missing facts for a given knowledge graph (Chen et al., 2020). Traditional KGE or KGC approaches usually treat KGs to be static, which means that the nodes and edges of a KG would not evolve with the time (Kazemi et al., 2020; Ji et al., 2021). However, in reality, most facts or events are only valid at a specific point or over a certain period. For example, in "Franklin D.Roosevelt was in office during 1933-1945 and died on April 12th, 1945", it indicates two facts ('in office' and 'death') related to the target person following different time orders and spans. In that case, KGC should be implemented at the temporal scale, and KGE models are supposed to have temporal awareness.

To better capture the knowledge evolution, recent temporal KG reasoning (TKGR) or temporal KG completion (TKGC) researches try to integrate the temporal information into the KGE procedure. Methods can be roughly divided into two categories: the structure-based methods and the sequence-based methods. Acting as an extension of static KGE modelling, the former type aims to project the entities and relations into a time-dependent vector space with the inherent KG structure preserved (Jiang et al., 2016; Dasgupta et al., 2018; Xu et al., 2020; Lacroix et al., 2020); they behave to be time-efficient on account of linear or multi-linear transformations but face with the shallow representation problem. The sequence-based methods (Jin et al., 2020; Wu et al., 2020; Li et al., 2021), splitting the entire KG into a sequence of graph snapshots along the time, rely on the sequence models such as the recurrent neural network (RNN) (Jin et al., 2020; Trivedi et al., 2017; Seo et al., 2018), long and short-term memory (LSTM) (Wu et al., 2020) or gated recurrent unit (GRU) (Li et al., 2021) to inherently encode temporal features. Although the performances based on the sequence models are reasonably good, they are computationally expensive when running over a large-scale KG. Moreover, their prediction results would be undermined by temporal sparsity to some extent due to the reason that only a small fraction of nodes and edges are activated at each time (Wu et al., 2020).

In this paper, we are inclined to apply the structure-based method to manage TKGR or TKGC. Inspired by the successful applications of convolutional networks on static KGE (Dettmers et al., 2018;

Jiang et al., 2019; Balažević et al., 2019a), we hereby build a novel time-aware convolutional embedding (TaCE) model to infer the missing facts for a temporal knowledge graph (TKG). It highlights its advantage in generating the convolution filters constructed from timestamps and convolving the time information into the embeddings of entities and relations. Such a convolutional design enables the model to deeply, comprehensively and efficiently extract the temporal features and the static ones and achieves a better link prediction for the TKGC task. Results demonstrate that TaCE has the superiority in learning temporal information as well as its interactions with the entities and the relations. It balances the tradeoff between expressiveness and training speed. The key contributions of this article can be summarized as follows:

- We creatively propose time-aware convolution method to integrate the time information into the embeddings of entities and relations for a better link prediction.
- An extensive comparison has been conducted between TaCE and the state-of-the-art models to verify their performances on different public datasets.
- Further analyses have been done to prove that our model, TaCE, is able to capture the semantic meanings for the timestamps and learn facts with proper time order and consistency.

2 Related Work

Static KG representation learning To discover the unknown facts in KGs, substantial static KGE methods have been proposed in the last decade. These methods commonly convert entities and relations into continuous vector spaces, and employ a scoring function to measure the plausibility of each candidate for KGC. TransE (Bordes et al., 2013), is one of the most widely-used transitional distance models, to embed entities and relations. Motivated by TransE, a series of similar models including TransH (Wang et al., 2014), TransR (Lin et al., 2015), and TransD (Ji et al., 2015) are developed to achieve better link predictions. RESCAL (Nickel et al., 2011) and its extensions (DistMult (Yang et al., 2014), ComplEx (Trouillon et al., 2016), and TuckER (Balažević et al., 2019b)) represent typical semantic matching models using tensor/matric factorization. Apart from them, ConvE (Dettmers et al., 2018), ConvR (Jiang et al., 2019) and HypER (Balažević et al., 2019a) recently ignite the passions of using convolution networks for KGE; they successfully prove their feature expressiveness better than those linear ones in link prediction. However, there is no study by far introducing convolution modelling into TKGC.

Temporal KG representation learning As forementioned, representation learning for TKG can be roughly categorized into two classes: structure-based models and sequence-based models. The former represent the quadruples in TKG by building time-sensitive embedding models. TTransE (Jiang et al., 2016) and DE-SimplE (Goel et al., 2020) integrate timestamp information into the corresponding the embeddings of relations and entities to infer the missing knowledge. HyTE (Dasgupta et al., 2018) projects the entities and relations to the temporal hyperplanes. ChronoR (Sadeghian et al., 2021) and TeRo (Xu et al., 2020), the most recent works, treat the timestamped relations as a temporal rotation from the head entity to the tail entity. Sequence-based models, including Know-Evolve (Trivedi et al., 2017), GCRN (Seo et al., 2018), RE-NET (Jin et al., 2020) and RE-GCN (Li et al., 2021), attempt to use the sequential networks such as RNN, LSTM and GRU to learn time-dependent facts. However, they are designed for the task of graph extrapolation and "not compatible with TKGC settings (Wu et al., 2020). TeMP (Wu et al., 2020) is one of the sequence-based models, designed for TKGC tasks using graph neural network (GNN) and LSTM to capture the intra-graph patterns and inter-graph relationships, respectively. Therefore, we group TeMP, as the representative for the sequence-based model, into our baselines.

3 Problem Formulation

Before going to the details of our TaCE model, we formally define the key notations and TKGC task.

Temporal Knowledge Graph (TKG) A TKG G is composed by a set of real-world facts $\mathcal{G} = \{(s, r, o, t) \mid s, r \in \mathcal{E}, r \in \mathcal{R}, t \in \mathcal{T}\}$, where, in each quadruple, s denotes the head entity, r is the relation, o is the tail entity, t is a discretized timestamp; $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ stand for the all the entities, relations and timestamps belong to the TKG.

Temporal Knowledge Graph Completion (TKGC) The task of TKGC or TKGR is to infer the missing component, $(?, r, o, t)$, $(s, r, ?, t)$ or $(s, ?, o, t)$, given that the other three elements of the quadruple are known. It is supposed that the missing part exists in $\mathcal{E}$, $\mathcal{R}$. In this article, our job focuses on predicting the missing head or tail entity, but we only take $(s, r, ?, t)$ as the example for modelling expression in later section.

4 METHOD: TACE

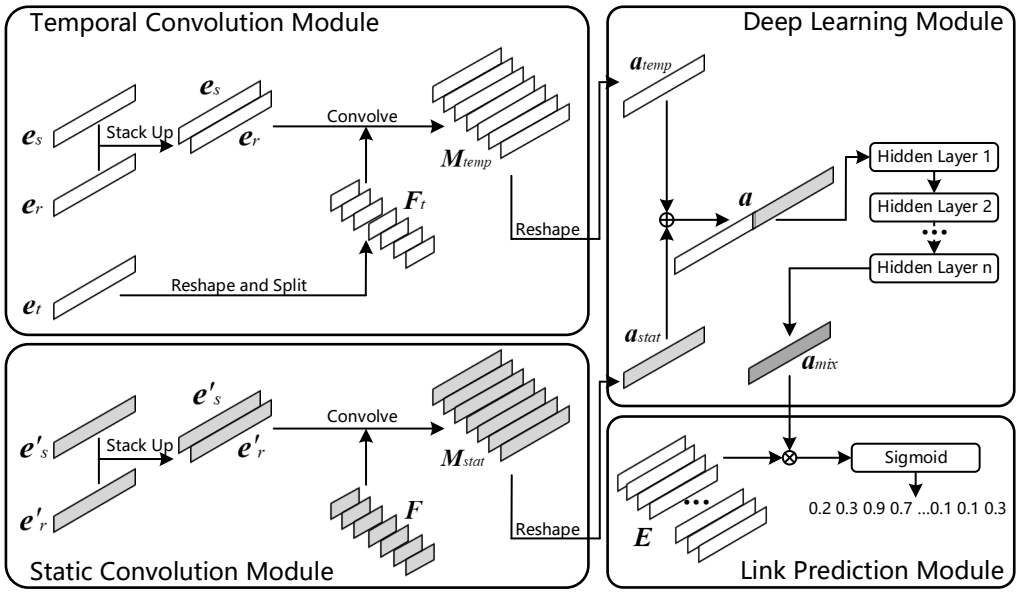

Figure 1: A framework of the proposed TaCE model for TKGC tasks.

4.1 FRAMEWORK OF TACE

The model of TaCE is mainly composed of four components: the temporal convolution module aims to construct temporal convolution filters to go through the input entities and relations to obtain the time-aware representations for all the input; the static convolution module, as the name suggests, is to facilitate the convolution across all the entities and relations again to learn the context information changed without the time; the deep learning module, made up of several hidden layers, is responsible for comprehensively and deeply drawing the representations carried both from the temporal and static modules; after that, the prediction layer delivers the probability for each tail entity candidate to suggest the most likely answer for the incomplete $\mathcal{G}$. Figure 1 displays the architecture of TaCE.

4.2 TEMPORAL CONVOLUTION MODULE

In this module, the temporal information formed by the timestamp $t \in \mathcal{T}$ is fully integrated into the subject entity $s \in \mathcal{E}$ and the relation $r \in \mathcal{R}$ via the convolution filters adaptively constructed from the timestamp t.

- Firstly, to integrate the time information into the entity s and the relation r, the temporal convolution filters are developed. They are constructed from the timestamp embedding $\mathbf{e}_t \in \mathbb{R}^{d_t}$, where d_t is the embedding size of the timestamps. Originally derived from the timestamp t, $\mathbf{e}_t$ is the input into a fully connected layer f_c to get a vector $\mathbf{v}_t$ with a proper length for further processing; then $\mathbf{e}_t$ is further split into a set of 1D convolution filters $\mathbf{F}_t = \left\{\mathbf{k}_t^{(1)}, \mathbf{k}_t^{(2)}, \ldots, \mathbf{k}_t^{(c)}\right\}$ sharing the same size, where $\mathbf{k}_t^{(l)} \in \mathbb{R}^w$ represents the lth convolution filter, w denotes the embedding

size of the filters and c denotes the number of the filters. The procedure of constructing temporal convolution filters is illustrated in Figure 2.

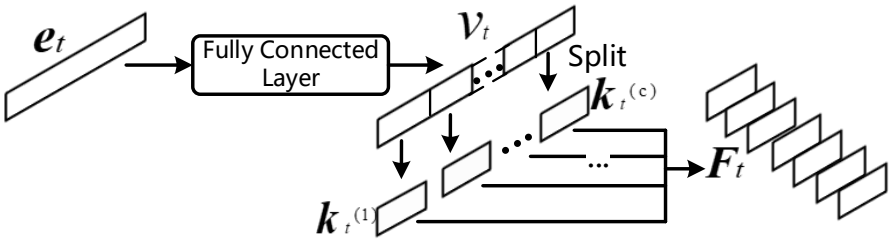

Figure 2: Construction of temporal convolution filters.

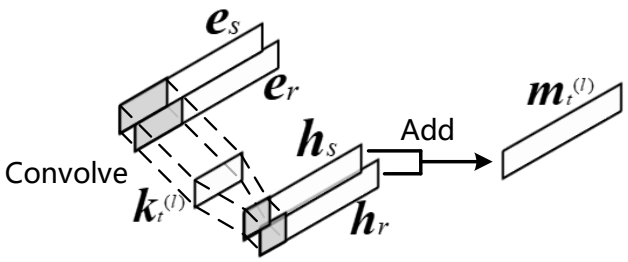

Figure 3: Convolution with temporal convolution filters.

- Secondly, before filtered by temporal convolutional filters $\mathbf{F}_t$, the subject entity embedding $\mathbf{e}_s \in \mathbb{R}^{d_e}$ and the relational embedding $\mathbf{e}_r \in \mathbb{R}^{d_r}$ are stacked up like an 'image' with multi channels, with $d_e = d_r$ arepresenting the embedding size of entities and relations respectively. Such a stacking operation enables both the entity information and relation information to feasibly act with $\mathbf{F}_t$. After convolved by $\mathbf{F}_t$, the corresponding temporal feature maps $\mathbf{M}_{temp} = \left\{\mathbf{m}_t^{(1)}, \mathbf{m}_t^{(2)}, \ldots, \mathbf{m}_t^{(c)}\right\}$, where $\mathbf{m}_t^{(l)} \in \mathbb{R}^{d_e - w + 1}$ is the lth feature map convolved from the subject entity embedding $\mathbf{e}_s$ and the relational embedding $\mathbf{e}_r$, and c is equal to the number of filters. The convolution with temporal convolution filters is illustrated in Figure 3. $\mathbf{h}_s$ and $\mathbf{h}_r$ denote the convolved $\mathbf{e}_s$ and $\mathbf{e}_r$ respectively in Figure 3.
- Finally, the matrix $\mathbf{M}_{temp}$ feed into a fully connected layer f_{temp} to obtain the flatten knowledge feature of $\mathbf{a}_{temp} \in \mathbb{R}^{d_e}$.

The equation of the temporal block is formulated as follow:

$$\mathbf{a}_{temp} = f_{temp}\left([\mathbf{e}_s; \mathbf{e}_r]_3 * vec^{-1}\left(f_c(\mathbf{e}_t)\right)\right) \tag{1}$$

where, vec^{-1} is a splitting operator to reshape the embedding $\mathbf{e}_t$ of the timestamp t into a set of filters, $*$ represents the convolutional operations, and $[\mathbf{e}_s; \mathbf{e}_r]_3$ represents the stacked tensor made up of the embedding $\mathbf{e}_s$ and the embedding $\mathbf{e}_r$.

4.3 Static Convolution Module

To capture the potential static information from a TKG, all the nodes and edges to form a TKG will go through the same set of the time-independent convolutional filters $\mathbf{F} = \left\{\mathbf{k}^{(1)}, \mathbf{k}^{(2)}, \ldots, \mathbf{k}^{(c)}\right\}$ with timestamps omitted. Each filter $\mathbf{k}^{(l)} \in \mathbb{R}^w$ in $\mathbf{F}$ is randomly initialized rather than constructed

from timestamps, as so to make filters time-irrelevant. The procedure of static embedding is described as:

$$\mathbf{a}_{stat} = f_{stat}\left([\mathbf{e}'_s; \mathbf{e}'_r]_3 * \mathbf{F}\right) \tag{2}$$

where, $\mathbf{e}'_s \in \mathbb{R}^{'d_e}$ and $\mathbf{e}'_r \in \mathbb{R}^{'d_r}$ represent the static embeddings for the subject s and the relation r, respectively. f_{stat} is a fully connected layer; $\mathbf{a}_{stat}$ presents the final extracted static KGs patterns.

4.4 Deep Learning Module

Deep learning module brings the temporal feature $\mathbf{a}_{temp}$ (derived from the temporal block) and the static feature $\mathbf{a}_{stat}$ (derived from the static block) into the multiple hidden layers to extract a better representation of $\mathbf{a} \in \mathbb{R}^{d_e}$ for all knowledge. The hidden layers are defined as follows:

$$\begin{cases} \mathbf{u}_1 & = \sigma_1(f_1([\mathbf{a}_{temp}; \mathbf{a}_{stat}])) \\ \mathbf{u}_2 & = \sigma_2(f_2(\mathbf{u}_1)) \\ & \dots \\ \mathbf{u}_n & = \sigma_n(f_n(\mathbf{u}_{n-1})) \end{cases} \tag{3}$$

where, n is the number of hidden layers, f_n denotes the linear operation for the hidden layer n, σ_n is the activation function of the hidden layer n, $\mathbf{u}_i$ is the result of the ith hidden layer, and we use LeakyReLU as the activation function for hidden layers.

4.5 Link Prediction Module

The sigmoid is chosen as the scoring function φ to predict the missing object entity o for the quadtuple $(s, r, ?, t)$:

$$\varphi(s, r, o, t) = f(\mathbf{e}_o \mathbf{u}_n) \tag{4}$$

where f is the sigmoid function; $\mathbf{u}_n$ is the hidden representation from the deep learning module. The complete formulation of φ for the task of TKGC is defined as:

$$\begin{aligned} &\varphi\left(s, r, o, t\right) = \\ &\quad f(\mathbf{e}_o \sigma_n(f_n(\dots(\sigma_1(f_1([f_{temp}([\mathbf{e}_s; \mathbf{e}_r]_3 * vec^{-1} f_c(\mathbf{e}_t); f_{stat}([\mathbf{e}'_s; \mathbf{e}'_r]_3 * \mathbf{F}))])))))) \end{aligned} \tag{5}$$

TaCE is capable of providing scores for all candidate object entities contained by $\mathcal{E}$, by employing an 1-N strategy. Such the 1-N strategy can speed up the entire scoring procedure which can be referenced to Dettmers et al. (2018) and Balažević et al. (2019a). We apply the binary cross entropy (BCE) loss function $\mathcal{L}$ to train the model:

$$\mathcal{L} = -\frac{1}{n} \sum_{i=1}^{n} y_i log(p_i) + (1 - y_i) log(1 - p_i) \tag{6}$$

where n presents the number of candidate entities; y_i labels the true $(y_i = 1)$ or false $(y_i = 0)$ prediction for the i^{th} candidate tail entity; p_i denotes the probability of the i^{th} candidate object entity upon the score function φ.

5 Experiments

5.1 Experimental settings

Datasets We test TaCE on six public benchmarks, namely, ICEWS14, ICEWS05-15, GDELT, YAGO11k, WIKIDATA12k and YAGO15k. Among them ICEWS14 and ICEWS05-15 are collected from the Integrated Crisis Early Warning System (ICEWS) with different time spans (Ward et al., 2013); the GDELT dataset is obtained from the Global Database of Events, Language, and Tone (GDELT) (Leetaru & Schrodt, 2013), spanning from April 1st, 2015 to March 31st, 2016; YAGO11k and WIKIDATA12k are extracted from YAGO3 (Mahdisoltani et al., 2014) and Wikidata (Erxleben et al., 2014) with time intervals (Xu et al., 2020); YAGO15k is the only one among the six that has incomplete time information, with 73.4% of the total number having no timestamps (Lacroix et al., 2020). Data details can be referred to Table 1.

Table 1: Statistics of the datasets. ($\|\mathcal{E}\|$, $\|\mathcal{R}\|$ and $\|\mathcal{T}\|$ are the total number of entities, relations and timestamps, respectively. $\|train\|$, $\|validation\|$ and $\|test\|$ are the number of quadruples in training, validation and test sets. $\|\mathcal{G}\|$ is the sum of $\|train\|$, $\|validation\|$ and $\|test\|$)

Dataset	$\|\mathcal{E}\|$	$\|\mathcal{R}\|$	$\|\mathcal{T}\|$	$\|$train$\|$	$\|$validation$\|$	$\|$test$\|$	$\|\mathcal{G}\|$
ICEWS14	7,128	230	365	72,826	8,941	8,963	90,730
ICEWS05-15	10,488	251	4017	368,962	46,275	46,092	461,329
GDELT	500	20	366	2,735,685	341,961	341,961	3,419,607
YAGO11k	10,622	10	398	16,408	2,050	2,051	20,509
WIKIDATA12k	12,554	24	614	32,497	4,062	4,062	40,621
YAGO15k	15,403	34	170	110,441	13,815	13,800	138,056

Evaluation Metrics Metrics including Hits@1, Hits@3, Hits@10, and Mean Reciprocal Rank (MRR) are involved to measure the performances of TaCE against the baseline models. Metric formulations can be found in Appendix A.2.

Baselines For a broad comparison, we collect as many KGE models as we can find. These baseline models are grouped into two: 1) the static ones, including TransE (Bordes et al., 2013), DistMult (Yang et al., 2014), ComplEx Trouillon et al. (2016), ConvE (Dettmers et al., 2018) and HypER (Balažević et al., 2019a); 2) the temporal ones, covering TTransE (Jiang et al., 2016), HyTE (Dasgupta et al., 2018), TA-DistMult (García-Durán et al., 2018), DE-SimplE (Goel et al., 2020), TIMEPLEX (Jain et al., 2020), TNTComplEx (Lacroix et al., 2020), TeRo (Xu et al., 2020), TeMP (Wu et al., 2020), and ChronoR (Sadeghian et al., 2021). The latter three, TNTComplEx, TeMP and ChronoR, to the best of our knowledge, are most recent models.

Parameter Settings For training and evaluation, the embedding size for entity, for relation and for timestamp are set equally the same, $d_e = d_r = d_t = 200$; the batch size is set to 512. We adopt Adam Optimizer (Kingma & Ba, 2015) for parameter training. More details about the parameter settings can be found in Appendix A.3.

5.2 Results and comparison

The experimental results associated with each model are summarized: the results on ICEWS14, ICEWS05-15 and GDELT are shown in Table 2; the results on YAGO11k, WIKIDATA12k and YAGO15k are shown in Table 5 in Appendix A.1 due to the page limitation. In general, our TaCE achieves the better performances on the datasets. When driving five traditional static models over the six TKG datasets, it is not surprising that none of them can provide satisfactory link predictions due to incapability of considering temporal information. Impressively, TaCE generally achieves better performances against its temporal opponents. On ICEWS14 and ICEWS05-15, TaCE delivers the upmost improvement of 4.9% on MRR and 6.5% on Hits@1 against the latest competitor, ChronoR. Although TaCE does not outperform TeMP on Hits@10, the rest marks on ICEWS14 and ICEWS05-15 indicate the superiority of TaCE. On GDELT dataset, TaCE leverages the rates of four metrics by at least 7% against its temporal opponents. Compared with our compellers, TaCE achieves the best on YAGO11k and WIKIDATA12k in terms of MRR, Hits@1 and Hits@3; results on YAGO15k are also acceptable, with Hits@1 reaching the highest and MRR and Hits@3 slightly lower than the SOTA (ChronoR). Overall, TaCE presents its effectiveness on link prediction by employing temporal convolutional filters to represent its interactions with entities and with relations.

5.3 Embedding visualization

To demonstrate the temporal expressiveness learned by TaCE, we use T-SNE to project the trained embeddings of $\mathbf{e}_t$ (for timestamps) and $\mathbf{a}_{temp}$ (for entity+relation) onto a 2D plane. Results are displayed in Figure 4&5.

Visualization of $\mathbf{e}_t$ According to Figure 4, the dimension-reduced $\mathbf{e}_t$ representing for the timestamps shows a good clustering pattern at different time scales: (a) and (b) demonstrate that the $\mathbf{e}_t$ points for ICEWS05-15 within the same year are clustered well together, transitioning from one

Table 2: Link prediction results on ICEWS14, ICEWS05-15 and GDELT datasets. The best results for each metric are marked in bold. All numbers of results are multiplied by 100%. Missing scores not reported are denoted by "–". Due to limited space, we use H@1, H@3 and H@10 to represent Hits@1, Hits@3 and Hits@10, respectively.

Model	*ICEWS14*				*ICEWS05-15*				*GDELT*			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
TransE(2013)	32.6	15.4	43.0	64.4	33.0	15.2	44.0	66.0	15.5	6.0	17.8	33.5
DistMult(2014)	44.1	32.5	49.8	66.8	45.7	33.8	51.5	69.1	21.0	13.3	22.4	36.5
ComplEx(2016)	44.2	40.0	43.0	66.4	46.4	34.7	52.4	69.6	21.3	13.3	22.5	36.6
ConvE(2018)	46.2	34.2	52.5	70.0	46.7	34.4	53.1	70.5	18.1	9.9	19.3	33.9
HypER(2019)	47.0	35.1	53.3	70.6	48.2	35.8	54.9	72.0	19.7	11.2	21.2	36.4
TTransE(2016)	25.5	7.4	-	60.1	27.1	8.4	-	61.6	11.5	0.0	16.0	31.8
HyTE(2018)	29.7	10.8	41.6	65.5	31.6	11.6	44.5	68.1	11.8	0.0	16.5	32.6
TA-DistMult(2018)	47.7	-	36.3	68.6	47.4	34.6	-	72.8	20.6	12.4	21.9	36.5
DE-SimplE(2020)	52.6	41.8	59.2	72.5	51.3	39.2	57.8	74.8	23.0	14.1	24.8	40.3
TIMEPLEX(2020)	60.4	51.5	-	77.1	64.0	54.5	-	81.8	-	-	-	-
TNTComplEx(2020)	60.7	51.9	65.9	77.2	66.6	58.3	71.8	81.7	-	-	-	-
TeRo(2020)	56.2	46.8	62.1	73.2	58.6	46.9	66.8	79.5	-	-	-	-
TeMP(2020)	60.7	48.4	68.4	**84.0**	68.0	55.3	76.9	**91.3**	23.2	15.2	24.5	37.7
ChronoR(2021)	62.5	54.7	66.9	77.3	67.5	59.6	72.3	82.0	-	-	-	-
TaCE(ours)	**67.4**	**61.2**	**71.0**	78.8	**68.3**	**61.2**	**72.6**	81.1	**31.4**	**22.6**	**34.0**	**48.6**

year to another; interestingly, the $\mathbf{e}_t$ for GDELT in (c) and (d) within the same months consecutively form a curving chain. In summary, the time embedding $\mathbf{e}_t$, trained by the convolution networks in TaCE, can automatically learn good sematic meanings for temporal order by itself.

Visualization of $\mathbf{a}_{temp}$ In Figure 5, $\mathbf{a}_{temp}$ stands for the mapping representations of facts *(s: boko haram, r: use conventional military force)* in different years. It can be observed that this sampled temporal fact based on ICEW05-15 is evolving with time in 2011, 2013 and 2015; points sharing close distances are usually those falling in the same year. Hence, it convinces that our model has the capability of capturing evolving facts with semantic meanings.

5.4 The awareness of time

To further demonstrate the time awareness of TaCE in factual prediction, we further plot the probabilities provided by TaCE for the fact who (*Umaru Musa Yar'Adua or Muhammadu Buhari*) potentially made a "statement" for the "Government (Nigeria)" during 2005-2015 as shown in Figure 6. Upon the knowledge shown in Appendix A.4, *Umaru Musa Yar'Adua* and *Muhammadu Buhari* hold their president tenure in different years. Our TaCE infers that *Umaru Musa Yar'Adua* has the highest scores for the fact *(s: ?, r: make statement, o: Government (Nigeria), 2007-2011)*, but Muhammadu Buhari for *(s: ?, r: make statement, o: Government (Nigeria), 2015)*. The reasoning results match the grounding truth, implying that our model has good time awareness to distinguish fact rankings along the time. A list of Nigerian presidents and their tenure during 2005-2015 can be found in Appendix A.4.

5.5 Ablation study

We conduct ablation studies on ICEWS14 to explore to what extent the temporal and static components may impact on the model prediction. The testing results are reported in Table 3.

Impact of temporal embedding module When we remove the temporal embedding module from TaCE, the model only performs as a static KGE model. The prediction results for MRR, Hits@1, Hits@3 and Hits@10 all bear significant drops compared to the states under the full condition. It highlights that the model closely relies on the temporal module to capture the evolving knowledge to enhance its prediction accuracy.

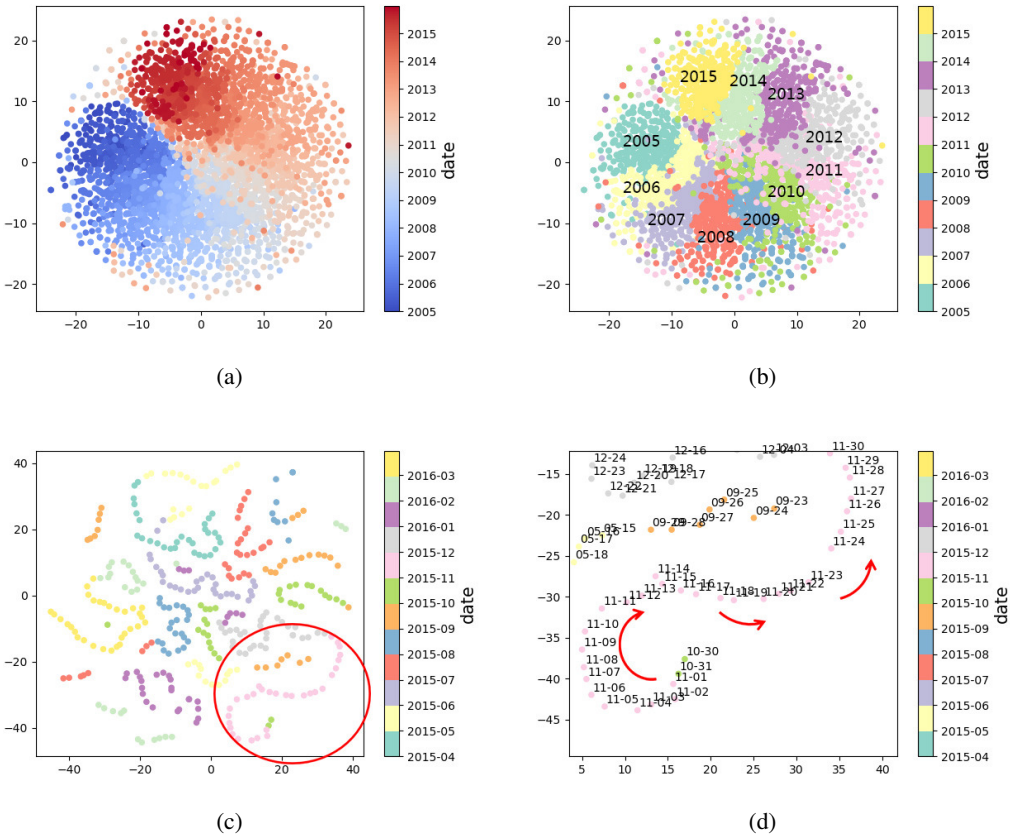

Figure 4: Visualization of $\mathbf{e}_t$. (a)&(b) map out all the trained temporal embeddings based on ICEWS05-15, with the ones in the same year marked by the same color; (c) maps out all the timestamps contained in GDELT with the ones in the same month marked by the same color; (d) is the zoom-in region highlighted by the red circle in subgraph (c).

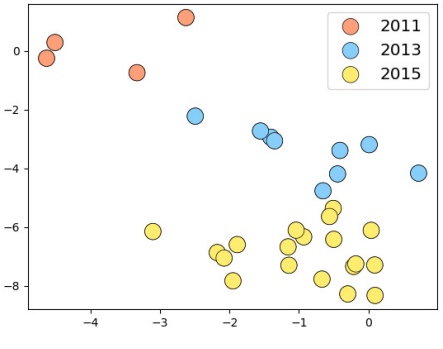

Figure 5: Visualization of $\mathbf{a}_{temp}$ representing the incomplete fact of *s: boko haram, r: use conventional military force* in different years, based on the training of ICEWS05-15.

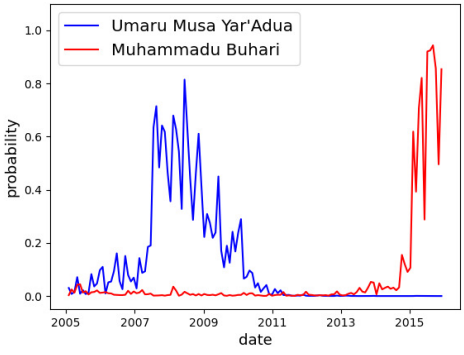

Figure 6: Probabilities for the fact *(s:?, r: make statement, o: Government (Nigeria))* druing 2005-2015, based on the training of ICEWS05-15.

Impact of static embedding module The static embedding module is responsible for learning the context knowledge of a KG. It can be observed that the performance of TaCE is slightly undermined when removing the static component off the model. This is due to the reason that ICEWS are highly dynamic data. If a KG contains more static facts, i.e., *(Beijing, located in, China)* or *(Barack Obama, spouse of, Michelle Obama)*, the KGE model is necessary to incorporate the static embedding module to represent the context knowledge properly.

Table 3: Ablation studies of TKGC tasks on ICEWS14. MRR, Hits@1, Hits@3 and Hits@10 are calculated with all numbers are multiplied by 100%.

Configuration	MRR	Hits@1	Hits@3	Hits@10
TaCE	67.4	61.2	71.0	78.8
TaCE without temporal embedding module	48.9	37.3	54.7	72.0
TaCE without static embedding module	66.2	59.6	70.0	78.4

5.6 Training efficiency

To test the training efficiency of TaCE for the TKGC task, we conduct a comparison of our model against its two competitors, TNTComplEx (a structure-based model) and TeMP (a sequence-based model). The tests are implemented on training 72,826 quadruples for ICEWS14 under the same computing environment. The results are shown in Table 4. Among the three tested models, TNTComplEx performs as the fastest one due to owning the tensor factorization method in its algorithm. TaCE, running convolution networks over a KG, can achieve a satisfying efficiency. TeMP costs two orders of magnitude of time to complete the whole train task, as it applies a complex GRU model to learn temporal patterns. The runtime will soar up to a high level when running over billions or trillions of facts.

Table 4: Training efficiency (in seconds) on ICEWS14.

Model	Runtime (one epoch)
TNTComplEx	2.5
TeMP	316.8
TaCE(our model)	4.9

6 Conclusion

In this paper, a convolution-based model called TaCE is proposed to manage the task of TKGC. The key idea is to generate the timestamp-based convolution filters based on timestamps to convolve the time-dependent entities and relations contained in a knowledge graph. Such a design allows the model to have a good expressiveness for the evolving knowledge and enables it to distinguish the facts with the time order and consistency. Furthermore, it can easily combine the static knowledge features together to achieve a better link prediction. The model has good efficiency in training procedure which allows it to run over those large-scale KGs. Additional properties such as spatial information or the description of an entity are expected to add in easily, however, this would refer to another paper in the future.

References

Ivana Balažević, Carl Allen, and Timothy M Hospedales. Hypernetwork knowledge graph embeddings. In *International Conference on Artificial Neural Networks*, pp. 553–565. Springer, 2019a.

Ivana Balažević, Carl Allen, and Timothy M Hospedales. Tucker: Tensor factorization for knowledge graph completion. *Proceedings of the 2019 Conference on Empirical Methods in Natural*

Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019b.

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

Xiaojun Chen, Shengbin Jia, and Yang Xiang. A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications*, 141:112948, 2020.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2001–2011, 2018.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. Introducing wikidata to the linked data web. In *International semantic web conference*, pp. 50–65. Springer, 2014.

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

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

Prachi Jain, Sushant Rathi, Soumen Chakrabarti, et al. Temporal knowledge base completion: New algorithms and evaluation protocols. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.

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

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Baobao Chang, Sujian Li, and Zhifang Sui. Towards time-aware knowledge graph completion. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 1715–1724, 2016.

Xiaotian Jiang, Quan Wang, and Bin Wang. Adaptive convolution for multi-relational learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 978–987, 2019.

Woojeong Jin, He Jiang, Meng Qu, Tong Chen, Changlin Zhang, Pedro Szekely, and Xiang Ren. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.

Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: A survey. *J. Mach. Learn. Res.*, 21 (70):1–73, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR(Poster)*, 2015.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. Tensor decompositions for temporal knowledge base completion. *International Conference on Learning Representations*, 2020.

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

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. Temporal knowledge graph reasoning based on evolutional representation learning. *arXiv preprint arXiv:2104.10353*, 2021.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. Yago3: A knowledge base from multilingual wikipedias. In *7th biennial conference on innovative data systems research*. CIDR Conference, 2014.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Icml*, 2011.

Ali Sadeghian, Mohammadreza Armandpour, Anthony Colas, and Daisy Zhe Wang. Chronor: Rotation based temporal knowledge graph embedding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*, pp. 362–373. Springer, 2018.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *International Conference on Learning Representations*, 2019.

Rakshit Trivedi, Mehrdad Farajtabar, Yichen Wang, Hanjun Dai, Hongyuan Zha, and Le Song. Know-evolve: Deep reasoning in temporal knowledge graphs. *arXiv preprint arXiv:1705.05742*, 2017.

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

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

Michael D Ward, Andreas Beger, Josh Cutler, Matthew Dickenson, Cassy Dorff, and Ben Radford. Comparing gdelt and icews event data. *Analysis*, 21(1):267–297, 2013.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L Hamilton. Temp: temporal message passing for temporal knowledge graph completion. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. Tero: A time-aware knowledge graph embedding via temporal rotation. *Proceedings of the 28th International Conference on Computational Linguistics*, 2020.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

A Appendix

A.1 Experiments on YAGO11k, WIKIDATA12k and YAGO15k

The experimental results on YAGO11k, WIKIDATA12k and YAGO15k are shown in Table 5.

Table 5: Link prediction results on YAGO11k, WIKIDATA12k and YAGO15k datasets. The best results for each metric are marked in bold. All numbers of results are multiplied by 100%. Missing scores not reported are denoted by "–". Due to limited space, we use H@1, H@3 and H@10 to represent Hits@1, Hits@3 and Hits@10, respectively.

Model	*YAGO11k*				*WIKIDATA12k*				*YAGO15k*			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
TransE(2013)	10.0	1.5	13.8	24.4	17.8	10.0	19.2	33.9	29.6	22.8	-	46.8
DistMult(2014)	15.8	10.7	16.1	26.8	22.2	11.9	23.8	46.0	27.5	21.5	-	43.8
ComplEx(2016)	16.7	10.6	15.4	28.2	23.3	12.3	25.3	43.6	36.0	29.0	36.0	54.0
ConvE(2018)	13.9	8.8	13.3	25.4	21.5	12.2	23.3	41.4	10.8	3.7	9.4	31.2
HypER(2019)	14.8	9.4	14.8	25.4	21.1	11.8	22.3	42.5	10.4	3.3	8.7	32.8
TTransE(2016)	10.8	2.0	15.0	25.1	17.2	9.6	18.4	32.9	32.1	23.0	-	51.0
HyTE(2018)	10.5	1.5	14.3	27.2	18.0	9.8	19.7	33.3	-	-	-	-
TA-DistMult(2018)	16.1	10.3	17.1	29.2	21.8	12.2	23.2	44.7	29.1	21.6	-	47.6
DE-SimplE(2020)	15.1	8.8	-	26.7	25.3	14.7	-	49.1	-	-	-	-
TIMEPLEX(2020)	23.6	16.9	-	36.7	33.4	22.8	-	**53.2**	-	-	-	-
TNTComplEx(2020)	18.0	11.0	-	31.9	30.1	19.7	-	50.7	35.9	28.5	36.8	53.8
TeRo(2020)	18.7	12.1	19.7	31.9	29.9	19.8	32.9	50.7	-	-	-	-
TeMP(2020)	-	-	-	-	-	-	-	-	-	-	-	-
ChronoR(2021)	-	-	-	-	-	-	-	-	**36.6**	29.2	**37.9**	**53.8**
TaCE(ours)	**23.8**	**17.1**	**25.3**	36.4	**33.6**	**23.4**	**36.1**	52.7	36.2	**30.4**	37.0	47.5

A.2 The details of definitions on evaluation metrics

MRR, Hits@1, Hits@3 and Hits@10 are employed to score measure the model performance. MRR is defined as follow:

$$MRR = \frac{1}{2\|test\|} \sum_{(s,r,o,t)\in test} (\frac{1}{rank_o} + \frac{1}{rank_s}) \tag{7}$$

where $rank_o$ and $rank_s$ denote the ranking of tail entity o and head entity s for tail and head queries, respectively. *Hits@K, K=1,2,3...*, is defined as follow:

$$Hits@K = \frac{1}{2\|test\|} \sum_{(s,r,o,t)\in test} (I(rank_o \leq k) + I(rank_s \leq k)) \tag{8}$$

where $k = 1, 2, 3 \ldots$, I denotes the indicator function.

A.3 Parameter Settings

The parameter settings for TaCE is listed in Table 6.

A.4 Nigerian presidents and their tenure from 2005 to 2015

There mainly refer to four presidents of Nigeria from 2005 to 2015 by checking on the Wikipedia. Their tenure information is listed in Table 7.

A.5 The number of parameters (Space Complexity) of TKGC models

The number of parameters or the space complexity of TKGC models are shown in Table 8.

Table 6: Parameter settings in TaCE.

Dataset	ICEWS14	ICEWS05-15	GDELT	YAGO11k	WIKIDATA12k	YAGO15k
Embedding size	200	200	200	200	200	200
Batch size	512	512	512	512	512	512
Embedding dropout rate	0.2	0.2	0.2	0.2	0.2	0.2
Feature map dropout rate	0.2	0.1	0.1	0.2	0.2	0.2
Projection dropout rate	0.3	0.2	0.2	0.3	0.3	0.3
Label smoothing	0.0	0.0	0.0	0.1	0.1	0.1
Number of feature maps	32	32	32	32	32	32
Convolutional filer size	1x9	1x9	1x9	1x9	1x9	1x9
Number of deep layers	1	1	1	1	1	1
Learning rate (Adam)	0.0005	0.0001	0.0001	0.001	0.001	0.001
Exponential learning rate decay	0.995	0.995	0.995	0.99	0.99	0.99

Table 7: The list of Nigerian presidents and their tenure from 2005 to 2015.

Name	Start date of tenure	End date of tenure
Olusegun Obasanjo	May 29th, 1999	May 29th, 2007
Umaru Musa Yar'Adua	May 29th, 2007	May 5th, 2010
Goodluck Ebele Jonathan	May 29th, 2010	May 29th, 2015
Muhammadu Buhari	May 29th, 2015	Till date

Table 8: The number of parameters (Space Complexity) of TKGC models. n_e, n_r and n_t denote the number of entities, relations and timestamps respectively. d_e, d_r and d_t denote the dimension of entity, relation and timestamp embedding respectively. n_{token} represents the token of time, which is defined in (García-Durán et al., 2018).

Model	The Number of Paramters (Space Complexity)
TransE(2013)	$\mathcal{O}(n_e d_e + n_r d_r)$
DistMult(2014)	$\mathcal{O}(n_e d_e + n_r d_r)$
ComplEx(2016)	$\mathcal{O}(2n_e d_e + 2n_r d_r)$
ConvE(2018)	$\mathcal{O}(n_e d_e + n_r d_r)$
HypER(2019)	$\mathcal{O}(n_e d_e + n_r d_r)$
TTransE(2016)	$\mathcal{O}(n_e d_e + n_r d_r + n_t d_t)$
HyTE(2018)	$\mathcal{O}(n_e d_e + n_r d_r + n_t d_t)$
TA-DistMult(2018)	$\mathcal{O}(n_e d_e + n_r d_r + n_{token} d_t)$
DE-SimplE(2020)	$\mathcal{O}(n_e d_e + n_r d_r)$
TIMEPLEX(2020)	$\mathcal{O}(2n_e d_e + 6n_r d_r + 2n_t d_t)$
TNTComplEx(2020)	$\mathcal{O}(2n_e d_e + 2n_r d_r + 2n_t d_t)$
TeRo(2020)	$\mathcal{O}(n_e d_e + n_r d_r + n_t d_t)$
TeMP(2020)	$\mathcal{O}(n_e d_e + n_r d_r)$
ChronoR(2021)	$\mathcal{O}(2n_e d_e + 2n_r d_r + 2n_t d_t)$
TaCE(ours)	$\mathcal{O}(2n_e d_e + 2n_r d_r + n_t d_t)$