A Simple But Powerful Graph Encoder for Temporal Knowledge Graph Completion

Anonymous ACL submission

Abstract

While knowledge graphs contain rich semantic knowledge about various entities and the relational information among them, temporal knowledge graphs (TKGs) describe and model the interactions of the entities over time. In this context, automatic temporal knowledge graph completion (TKGC) has gained great interest. Recent TKGC methods aim to integrate advanced deep learning techniques, e.g., Transformers, to boost model performance. However, we find that instead of adopting various kinds of complex modules, it is more beneficial to capture more extensive temporal information. In this paper, we propose a simple but powerful graph encoder for TKGC, namely, TARGCN. TARGCN is parameter-efficient, and it extensively utilizes the information from the whole temporal context. We perform experiments on three benchmark datasets. Our model can achieve a more than 46% relative improvement on the GDELT dataset compared with state-of-the-art models. Meanwhile, it outperforms the strongest baseline on the ICEWS05-15 dataset with around 18% fewer parameters.

1 Introduction

A *Knowledge Graph* (KG) is a graph-structured *Knowledge Base* (KB) that stores relational facts. KGs have drawn increasing research interest since they serve as key drivers for a wide range of downstream tasks in artificial intelligence, e.g., question answering (Zhang et al., 2018), commonsense reasoning (Xing et al., 2021), and recommender systems (Wang et al., 2019). A fact in a KG is described as a triplet (s, r, o), e.g., (*Joe Biden*, *is president of*, *USA*), where s, o, r denote the subject entity, the object entity, and the relation between s and o. While KGs contain rich semantic knowledge about entities and the relational information among them, they do not consider the nature of ever-evolving relational facts over time. For example, consider a KG triplet (*Donald Trump*, *is president of*, *USA*). According to world knowledge, this triplet is valid only before *Joe Biden* took the place of *Donald Trump* as the president of the *USA*. This implies a shortcoming of KGs and calls for the introduction of *Temporal Knowledge Graphs* (TKGs). In TKGs, every fact is augmented with a specific timestamp t such that it can be described with a quadruple (s, r, o, t). In this way, every fact in TKGs has its own time validity and this enables TKGs to capture the factual information in a time-varying context.

Temporal Knowledge Graph Completion (TKGC) is a task aiming to infer the missing facts in TKGs. There exist two lines of TKGC methods. (1) A lot of prior methods attempt to incorporate temporal information into the existing KG reasoning scoring models and build novel time-aware score functions for TKGs (Leblay and Chekol, 2018; García-Durán et al., 2018; Ma et al., 2019; Lacroix et al., 2020; Messner et al., 2021). (2) Another line of models take advantage of the recent progress of *Graph Neural Networks* (GNNs) (Niepert et al., 2016; Kipf and Welling, 2017) and develop time-aware relational graph encoders for TKGC (Wu et al., 2020; Jung et al., 2021). Experimental results show that time-aware relational graph encoders help to achieve state-of-the-art performance on the TKGC task. However, employing an additional graph encoder on top of the existing KG score functions normally leads to a higher number of model parameters. The parameter consumption increases even more when these models are equipped with advanced deep learning structures, e.g., attention mechanisms and Transformers (Vaswani et al., 2017).

In this paper, we follow the trend of the second line of methods and propose a time-aware relational graph encoder: ***T**ime-**a**ware **R**elational **G**raph **C**onvolutional **N**etwork* (TARGCN). We find that our light-weighted time-aware relational graph encoder performs well on the TKGC task, and it requires relatively few parameters. The contribution of our work can be summarized as follows:

- We propose a novel time-aware relational graph encoder: ***T**ime-**a**ware **R**elational **G**raph **C**onvolutional **N**etwork* (TARGCN) for the TKGC task. TARGCN extensively utilizes all available information in the temporal context and generates time-aware graph representations. Instead of directly learning representations restricted to specific timestamps, it learns temporal information by modeling time differences among entities appearing at different time with a functional time encoder.

- TARGCN serves as a light-weighted, parameter-efficient, and robust TKG reasoning model. To achieve the same performance, our model requires much fewer parameters compared with two recently proposed GNN-based TKG reasoning models, TeMP (Wu et al., 2020) and T-GAP (Jung et al., 2021). Additionally, TARGCN achieves superior performance in link inference on irregular timestamped data, and generalizes well to unseen timestamps, thus showing its robustness.

- We evaluate our model on three benchmark datasets of TKGC. Our model achieves state-of-the-art performance on all datasets. On the GDELT (Leetaru and Schrodt, 2013) dataset, it achieves a more than 46% relative improvement compared with the best baseline.

2 Preliminaries and Related Work

2.1 Knowledge Graph Embedding Models

Knowledge graph embedding (KGE) models have shown great success in KG reasoning tasks. TransE (Bordes et al., 2013) is the first KGE model that introduces translational embeddings into KG representation learning. Many further works (Lin et al., 2015; Sun et al., 2019; Abboud et al., 2020) are inspired and extend the relational translations in different spaces to capture complex relational information. Another line of KGE methods are tensor factorization-based models (Nickel et al., 2011; Yang et al., 2015; Balazevic et al., 2019). They encode entity and relation embeddings as vectors and then use bilinear functions to compute the plausibility scores for KG facts. Apart from these two mainstream types of KGE models, neural-based relational graph encoders have been rapidly developed and have shown great power in capturing structural information of KGs. R-GCN (Schlichtkrull et al., 2018) incorporates relation information into a *Graph Convolutional Network* (GCN) (Kipf and Welling, 2017) to enable relational reasoning on KGs. Recently, CompGCN (Vashishth et al., 2020) extends this idea and leverages a variety of composition operations between KG entities and relations. It shows great effectiveness on KG reasoning tasks.

2.2 Temporal Knowledge Graph Embedding Models

Temporal knowledge graph embedding (TKGE) models can be categorized into several classes according to their temporal information encoding techniques. A series of models treat every timestamp separately and assign a high-dimensional vector as its embedding (Tresp et al., 2017; Leblay and Chekol, 2018; Lacroix et al., 2020). The assigned timestamp embeddings lie in the same space as entity and relation embeddings. Another series of models assume that every entity has a time-aware embedding that evolves over time (Xu et al., 2020a; Goel et al., 2020). To achieve time-aware property, an entity together with a timestamp are input into a function (or neural network) to yield a time-aware entity representation at this timestamp. Besides, García-Durán et al. jointly encode entity, relation and time information with *Recurrent Neural Network* (RNN) to learn time-aware graph representations (García-Durán et al., 2018). Instead of modeling timestamp information, some recent models attempt to model time difference, i.e., time displacement, between the query event and known events (Wu et al., 2020; Jung et al., 2021). It turns out that time displacement modeling can contribute to superior performance on TKG reasoning tasks.

2.2.1 Temporal Knowledge Graph Completion

Let $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ denote a finite set of entities, relations and timestamps, respectively. A temporal knowledge graph $\mathcal{G}$ is a graph which represents the evolution of interactions among entities over time. At any timestamp $t \in \mathcal{T}$, $\mathcal{G}(t)$ is called the TKG snapshot at t, and it can be taken as a static KG containing the facts valid at t. Any fact, i.e., event, can be described with a quadruple (s, r, o, t), where $s \in \mathcal{E}$ represents the subject, $o \in \mathcal{E}$ represents the object, $r \in \mathcal{R}$ represents the relation between s and o, and $t \in \mathcal{T}$ indicates the timestamp when this fact is valid. Therefore, at t, the TKG snapshot can be summarized as a finite set of all the valid facts at this timestamp t, i.e., $\mathcal{G}(t) = \{(s, r, o, t) | s, o \in \mathcal{E}, r \in \mathcal{R}\}$. We denote a TKG as a sequence of TKG snapshots $\mathcal{G} = \{\mathcal{G}(1), ..., \mathcal{G}(T)\}$, where $T = |\mathcal{T}|$ is the number of timestamps. Similarly, we can also denote a TKG as a finite set of all valid facts which happen at any timestamp $t \in \mathcal{T}$, i.e., $\mathcal{G} = \{(s, r, o, t) | s, o \in \mathcal{E}, r \in \mathcal{R}, t \in \mathcal{T}\}$.

We define the TKGC task as follows. For every snapshot $\mathcal{G}(t)$ in an *observed* TKG $\mathcal{G} = \{\mathcal{G}(1), ..., \mathcal{G}(T)\}$, it contains all the *observed* facts at t. Let $\bar{\mathcal{G}}(t)$ denote the set of all the *true* facts at t such that $\mathcal{G}(t) \in \bar{\mathcal{G}}(t)$. TKGC aims to predict the ground truth object (or subject) entities of queries $(s, r, ?, t)$ (or $(?, r, o, t)$), where $(s, r, o, t) \in \bar{\mathcal{G}}(t)$ but $(s, r, o, t) \notin \mathcal{G}(t)$, given any $t \in \mathcal{T}$.

TKGC has recently gained increasing interest. Researchers have paid great attention to better modeling the temporal information brought by the nature of TKGs. As fancier techniques and advanced deep learning methods, e.g., attention mechanisms and Transformers (Vaswani et al., 2017), being extensively studied, recent TKG reasoning models (Wu et al., 2020; Jung et al., 2021) benefit from them and show great performance on TKGC.

3 Our Method

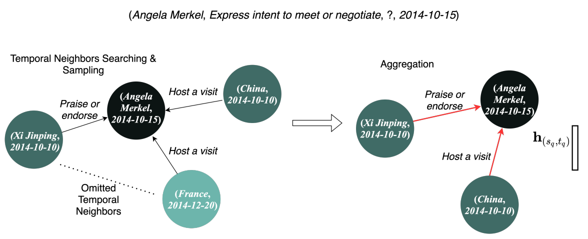

Figure 1: The encoding process in TARGCN for the query (*Angela Merkel*, *Express intent to meet or negotiate*, ?, *2014-10-15*). The color darkness on each node implies its probability of being sampled as an input at the aggregation step (the darker the higher).

To solve the TKGC task, our relational graph encoder TARGCN extensively collects information from the whole temporal context and updates the time-aware representations of entities. For every link prediction query $(s_q, r_q, ?, t_q)$, TARGCN first creates a subgraph for the subject s_q, according to its temporal associated neighbors. Then it derives time-aware representations for the neighbors from the temporal neighborhood, and performs aggregation. After s_q's time-aware representation is updated, a knowledge graph decoder (score function) is utilized to compute scores for every candidate object, which yields the plausibility of every candidate object being the ground truth object in the link prediction query $(s_q, r_q, ?, t_q)$. Note that we only consider object prediction queries $(s_q, r_q, ?, t_q)$ in our work since we add reciprocal relations for every quadruple , i.e., adding (o, r^{-1}, s, t) for every (s, r, o, t). The restriction to only predict object entities does not lead to a loss of generality. An example is presented in Figure 1 which shows the encoding process of our model. For the query subject *Angela Merkel* appearing at *2014-10-15*, TARGCN selects its temporal neighbors with a time difference-dependent probability. Node aggregation is then performed to learn a contextualized representation $\mathbf{h}_{(s_q,t_q)}$, where s_q, t_q correspond to *Angela Merkel* and *2014-10-15*, respectively.

3.1 Subgraph Sampling in Temporal Neighborhood

Given a TKGC query $(s_q, r_q, ?, t_q)$, TARGCN aims to learn a contextualized representation for the subject entity s_q. Inspired by the inference graph proposed in (Han et al., 2021), we sample a *Temporal Neighboring Graph* (TNG) for (s_q, t_q) in TKGC context, where (s_q, t_q) is the node representing s_q at t_q. We first find out all the temporal neighbors of (s_q, t_q), which can be described as a set $\mathcal{N}_{(s_q,t_q)} = \{(e, t) | (e, r, s_q, t) \in \mathcal{G}; e \in \mathcal{E}, t \in \mathcal{T}, r \in \mathcal{R}\}$. The entity e of a temporal neighbor (e, t) forms a link with s_q at timestamp t and s_q bears an incoming edge derived from the temporal associated quadruple (e, r, s_q, t). Note that in TKGC, though we cannot observe all the true quadruples, we still can observe part of true quadruples at *every* timestamp. This enables TARGCN to search for the temporal neighbors of (s_q, t_q) along the whole time axis. Then we employ weighted sampling strategy according to the absolute time difference $|t_q - t|$ between (s_q, t_q) and the corresponding temporal neighbor (e, t). For every temporal neighbor (e, t), the probability of it being sampled into (s_q, t_q)'s TNG is computed by: $exp(-|t_q - t|)/\Sigma_{(e,t') \in \mathcal{N}_{(s_q,t_q)}} exp(-|t_q - t'|)$. In this way, higher probabilities are assigned to the temporal neighbors who are closer to (s_q, t_q) along the time axis. We adopt this sampling strategy since we assume that for the inference of a fact at t_q, it is more likely to find clues from the factual information at nearer timestamps. Besides, we use a hyperparameter to limit the maximum number of the temporal neighbors included in (s_q, t_q)'s TNG to prevent over sampling *less-concerned* temporal neighbors. An example illustrating (s_q, t_q)'s temporal neighborhood is shown in Appendix A.

3.2 Time-aware Relational Aggregation

After sampling TNG for the subject entity s_q, we then attempt to learn its contextualized representation through neighborhood aggregation. Since we have access to temporal neighbors from the whole timeline, we implicitly incorporate temporal information. For every temporal neighbor, we employ the functional time encoding method proposed in (Xu et al., 2020b) to learn a time-aware node representation. In this way, we are able to distinguish the temporal neighbors, (e, t) and (e, t'), who root from the same entity e but emerge at different timestamps t and t'. The time-aware node representation is computed as:

$$\mathbf{h}_{(e,t)} = f(\mathbf{h}_e \| \mathbf{\Phi}(t, t_q)), \tag{1}$$

where $\mathbf{h}_e \in \mathbb{R}^{d_e}$ denotes the time-invariant entity-specific representation of the entity e. $\mathbf{\Phi}(\cdot, \cdot)$ is a time difference encoder, mapping $t - t_q$ to a finite dimensional functional space $\mathbb{R}^{d_t}$. We concatenate the time-invariant entity representation with its corresponding time difference representation, and learn a combined representation of them with a layer of feed-forward neural network f. Note that the sign of $t - t_q$ will affect the output of the time difference encoding module. The complete form of $\mathbf{\Phi}(\cdot, \cdot)$ is stated in Appendix F.

We aggregate the information from (s_q, t_q)'s temporal neighbors with a relational graph aggregator:

$$\mathbf{h}_{(s_q,t_q)} = \frac{1}{|\bar{\mathcal{N}}_{(s_q,t_q)}|} \sum_{(e,t) \in \bar{\mathcal{N}}_{(s_q,t_q)}} \mathbf{W}(\mathbf{h}_{(e,t)} \| \mathbf{h}_r). \tag{2}$$

$\bar{\mathcal{N}}_{(s_q,t_q)}$ denotes a finite set of temporal neighbors sampled from (s_q, t_q)'s temporal neighborhood, i.e., all the neighbors in (s_q, t_q)'s TNG. r is the relation appearing in the temporal associated quadruple (e, r, s_q, t) where temporal neighbor (e, t) is sampled. We assume that relation representations are time-invariant and we incorporate relational information into the graph encoder by concatenating time-aware node representations with them. Our graph encoder outputs the time-aware representation of s_q at query time t_q, by combining not only the raw entity representation $\mathbf{h}_e$ but also the implicit time difference information from its temporal neighbors. The prior work T-GAP also pays attention to modeling time displacement, i.e., time difference, to better learn time-aware entity representations. However, T-GAP explicitly models time displacement with a discretized embedding, and it includes three different weight matrices in their graph encoder for the facts happening in the past, at present, or in the future, thus increasing parameter consumption. We will discuss this later and compare its efficiency with our model's.

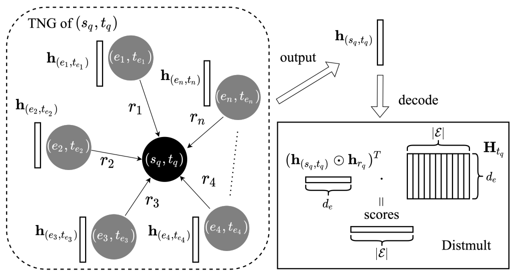

Figure 2: Inference process of TARGCN + Distmult. $\mathbf{H}_{t_q}$ is the embedding matrix containing time-aware representations of all candidates at t_q. For a TKGC query $(s_q, r_q, ?, t_q)$, we first sample a TNG rooting from (s_q, t_q) (marked with dashed line square). Then we employ TARGCN encoder to compute an aggregated representation $\mathbf{h}_{(s_q,t_q)}$ for (s_q, t_q). We provide Distmult with time-aware representations of all candidates for score computation. The candidate producing the highest score is selected as the predicted answer.

3.3 Learning and Inference

Figure 2 illustrates how TARGCN, together with a KG score function, i.e., Distmult (Yang et al., 2015), predicts the ground truth missing object for the TKGC query $(s_q, r_q, ?, t_q)$. Given s_q, we use the sampling strategy and our time-aware relational graph encoder to compute a time dependent node representation for (s_q, t_q). Then we use a simple KG score function Distmult to compute the plausibility of every candidate entity. We choose Distmult because it does not introduce additional parameters, which encounters our flavor of building a simple and parameter-efficient TKGC model. Note that for the candidate entities, we do not sample TNG for them to avoid huge time consumption during inference. Instead, for every candidate entity o', we simply derive its time-aware representation by computing $\mathbf{h}_{(o',t_q)} = f(\mathbf{h}_{o'} \| \mathbf{\Phi}(t_q, t_q))$. The temporal encoder $\mathbf{\Phi}(\cdot,\cdot)$ will also return a unique representation when time difference equals zero.

We employ cross-entropy loss for parameter learning:

$$\mathcal{L} = \sum_{(s,r,o,t)\in\mathcal{G}} -\log\left(\frac{score(\mathbf{h}_{(s,t)},\mathbf{h}_r,\mathbf{h}_{(o,t)})}{\sum_{o'\in\mathcal{E}} score(\mathbf{h}_{(s,t)},\mathbf{h}_r,\mathbf{h}_{(o',t)})}\right), \tag{3}$$

where o' denotes all candidate entities and we sum over all *observed* quadruples in $\mathcal{G}$. Note that our TARGCN encoder can be equipped with any KG score functions since our encoder returns time-aware representations for entities. In our work, $score(\mathbf{h}_{(s,t)}, \mathbf{h}_r, \mathbf{h}_{(o',t)}) =< \mathbf{h}_{(s,t)}, \mathbf{h}_r, \mathbf{h}_{(o',t)} >$.

4 Experiments

We evaluate our model on three TKGC benchmark datasets. We compare our model with several existing TKGC methods. To further show the parameter efficiency of our model, we do an analysis of parameter usage on TARGCN, compared with two recently proposed powerful TKGC models TeMP (Wu et al., 2020) and T-GAP (Jung et al., 2021). We then prove the robustness of TARGCN and present ablation studies in Section 4.4 and Section 4.5.

4.1 Experimental Setup

4.1.1 Datasets

We perform evaluation on three TKGC benchmark datasets: (1) ICEWS14 (García-Durán et al., 2018) (2) ICEWS05-15 (García-Durán et al., 2018) (3) GDELT (Leetaru and Schrodt, 2013). ICEWS14 and ICEWS05-15 are two subsets of *Integrated Crisis Early Warning System* (ICEWS) database. ICEWS14 contains timestamped political facts happening in 2014, while the timestamps of factual events in ICEWS05-15 span from 2005 to 2015. We follow (Wu et al., 2020) and use the GDELT subset proposed by (Trivedi et al., 2017). It contains global social facts from April 1, 2015 to March 31, 2016. The detailed dataset statistics are presented in Table 5 in Appendix B.

4.1.2 Evaluation Metrics

We employ two evaluation metrics for all experiments, i.e., Hits@1/3/10 and *Mean Reciprocal Rank* (MRR). For every test fact $(s_q, r_q, o_q, t_q) \in \bar{\mathcal{G}}$ ($(s_q, r_q, o_q, t_q) \notin \mathcal{G}$), we derive an associated TKGC query $(s_q, r_q, ?, t_q)$. We let models compute the rank of the ground truth entity o_q among all the candidates. Hits@1/3/10 are the proportions of the test facts where ground truth entities are ranked as top 1, top 3, top 10, respectively. MRR computes the mean of the reciprocal ranks of ground truth entities. We follow the *filtered* setting proposed by (Bordes et al., 2013) to achieve fairer evaluation.

4.1.3 Baseline Methods

We take fifteen methods as baseline models. The first four baselines are static KG reasoning methods, i.e., TransE (Bordes et al., 2013), Distmult (Yang et al., 2015), ComplEx (Trouillon et al., 2016) and SimplE (Kazemi and Poole, 2018). The other methods are developed to solve TKGC, including TTransE (Leblay and Chekol, 2018), TA-Distmult (García-Durán et al., 2018), HyTE (Dasgupta et al., 2018), DE-SimplE (Goel et al., 2020), ATiSE (Xu et al., 2020a), TNTComplEx (Lacroix et al., 2020), ChronoR (Sadeghian et al., 2021), TeLM (Xu et al., 2021), BoxTE (Messner et al., 2021), TeMP (Wu et al., 2020) and T-GAP (Jung et al., 2021). Among all baselines, only TeMP and T-GAP employ GNNs as graph encoders, similar to our TARGCN setting. Therefore, we further compare the parameter efficiency among them.

4.2 Experimental Results

Table 1 reports the experimental results of all methods on three benchmark datasets. We can observe that TARGCN outperforms all baselines on all datasets. The margin is particularly huge on the GDELT dataset. TARGCN achieves an over 46% relative improvement on MRR (0.163 absolute improvement) compared with the strongest baseline BoxTE. TARGCN also leads in Hits metrics greatly. It improves Hits@1/3/10 by 57.25%, 47.75%, and 34.83%, respectively. On ICEWS datasets, though TARGCN does not take a huge step forward, it still achieves the best results on MRR. TARGCN also shows particularly strong performance on Hits@1, which can be taken as the main contribution to its superior results on MRR.

We argue that the performance gap varies because of the characteristics of different datasets. While ICEWS datasets are sparse, GDELT is much denser. As discussed in (Wu et al., 2020; Messner et al., 2021), the temporal sparsity issue on ICEWS is much more severe than it on GDELT. This implies that GDELT contains substantially more temporal patterns, while ICEWS datasets are more prone to be biased by a large number of isolated events which are mainly dominated by sparse entities and relations. Hence, we argue that reason-

Datasets	ICEWS14				ICEWS05-15				GDELT			
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
TransE [▼]	0.326	0.154	0.43	0.644	0.330	0.152	0.440	0.660	0.155	0.060	0.178	0.335
Distmult ▼	0.441	0.325	0.498	0.668	0.457	0.338	0.515	0.691	0.210	0.133	0.224	0.365
ComplEx [▼]	0.442	0.400	0.430	0.664	0.464	0.347	0.524	0.696	0.213	0.133	0.225	0.366
SimplE [▼]	0.458	0.341	0.516	0.687	0.478	0.359	0.539	0.708	0.206	0.124	0.220	0.366
TTransE [▼]	0.255	0.074	-	0.601	0.271	0.084	-	0.616	0.115	0.000	0.160	0.318
TA-DistMult [▼]	0.477	0.363	-	0.686	0.474	0.346	-	0.728	0.206	0.124	0.219	0.365
HyTE [▼]	0.297	0.108	0.416	0.655	0.316	0.116	0.445	0.681	0.118	0.000	0.165	0.326
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
ATiSE [▼]	0.571	0.465	0.643	0.755	0.484	0.350	0.558	0.749	-	-	-	-
TNTComplEx [▼]	0.620	0.520	0.660	0.760	0.670	0.590	0.710	0.810	-	-	-	-
ChronoR [★]	0.625	0.547	0.669	0.773	0.675	0.596	0.723	0.820	-	-	-	-
TeLM [★]	0.625	0.545	0.673	0.774	0.678	0.599	0.728	0.823	-	-	-	-
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
TeMP-GRU [▼]	0.601	0.478	0.681	0.828	0.691	0.566	**0.782**	**0.917**	0.275	0.191	0.297	0.437
TeMP-SA [▼]	0.607	0.484	**0.684**	**0.840**	0.680	0.553	0.769	0.913	0.232	0.152	0.245	0.377
T-GAP [♡]	0.610	0.509	0.677	0.790	0.670	0.568	0.743	0.845	-	-	-	-
TARGCN	**0.636**	**0.576**	0.672	0.746	**0.702**	**0.635**	0.743	0.823	**0.515**	**0.423**	**0.557**	**0.689**

Table 1: Temporal knowledge graph completion results on three benchmark datasets. Evaluation metrics are filtered MRR and Hits@1/3/10. The best results are marked in bold. Results marked with [▼], [♡], [★] are taken from (Wu et al., 2020), (Jung et al., 2021), (Messner et al., 2021), respectively.

ing on GDELT requires much stronger techniques. And this can also be deduced by the performance of TKGC models. From Table 1, we can observe that for prior methods, though several TKGC methods outperform static methods on GDELT, the improvements are not substantial. However, on GDELT, TARGCN achieves a more than 141% relative improvement on MRR, compared with the strongest static KG baseline ComplEx. This shows the superior effectiveness of our graph encoder in capturing various temporal patterns. For ICEWS datasets, our model can also achieve state-of-the-art performance. This demonstrates its strong ability in capturing the temporal KG information brought by sparse entities and relations.

4.3 Parameter Efficiency Analysis

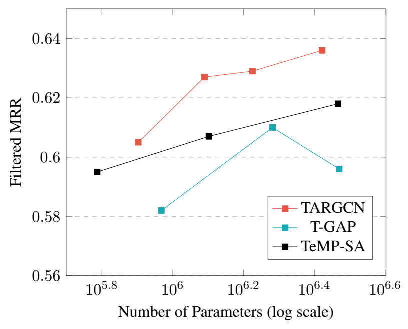

Figure 3: Filtered MRR on ICEWS14 achieved by TARGCN, T-GAP and TeMP-SA, with varied number of parameters. More details in Appendix G.

While TARGCN serves as a strong TKGC model, it also keeps a quite low parameter cost. In this section, we compare the parameter efficiency among TARGCN and two recently proposed GNN-based TKGC models, i.e., TeMP and T-GAP.

On ICEWS14, for all three models, we adjust the embedding size of both entities and relations to adjust the number of parameters. We do not change model structures and other optimal hyperparameter settings. In Figure 3, we show that TARGCN performs better as we increase model parameters. More importantly, even with much fewer parameters, TARGCN still outperforms TeMP and T-GAP.

For ICEWS05-15 and GDELT, we summarize the number of parameters as well as performance difference in Table 2. We compare across the models with parameter settings that lead to the experimental results shown in Table 1. On ICEWS05-15, T-GAP uses 30.89% more parameters than our model, but its performance drops by 4.56%. TeMP-GRU achieves almost the same result as TARGCN on ICEWS05-15, however, it uses 18.45% more parameters than our model. Fewer parameters are used in TeMP-SA, but it also leads to worse performance. On GDELT, we observe that though TeMP-GRU employs 50.07% more parameters than TARGCN, its performance is 46.60% lower than our model. TeMP-SA shows the worst performance (with a 54.95% performance drop), although it has even 5.19% fewer parameters than TARGCN. To this end, we argue that our model is extremely parameter-efficient.

We attribute such high parameter efficiency to

Datasets	ICEWS05-15				GDELT			
Model	# Parameters	MRR	Parameter ↑	MRR ↓	# Parameters	MRR	Parameter ↑	MRR ↓
TARGCN	2359200	0.702	-	-	269200	0.515	-	-
T-GAP	3088000	0.670	30.89%	4.56%	-	-	-	-
TeMP-SA	2645760	0.680	12.15%	3.13%	255232	0.232	-5.19%	54.95%
TeMP-GRU	2794528	0.691	18.45%	1.57%	404000	0.275	50.07 %	46.60%

Table 2: Parameter efficiency comparison on ICEWS05-15 and GDELT. We adopt relative change to define the increase in parameter numbers and the drop in MRR. Due to memory problem, we cannot train T-GAP on GDELT even when batch size equals 1.

our simple but powerful time-aware relational graph encoder. Note that in the TNG sampling process, we explicitly force our model to choose the temporal neighbors who are nearer to the source node (s_q, t_q) on the time axis, by assigning higher sampling probabilities to them. This can also be interpreted as a "hard-coded attentional process". Models like TeMP and T-GAP explicitly employ self-attention modules to let models choose their attention themselves through parameter learning. We argue that even if such modules are powerful, they can be simplified in the context of TKGC. In our model, we force our TNG sampler to focus on the facts happening at the timestamps that are closer to the query timestamp, i.e., pay more "attention" to the nearer facts. Our TNG sampling process does not include any additional parameters, while self-attention modules normally increase parameters and bring heavier burdens for parameter optimization. Another crucial point is that, compared with TeMP who encodes temporal information only from a fixed short time span of 2τ, our TNG sampling range spans across the whole timeline. This means that even if a temporal neighbor is derived from a sparse entity and it appears only at faraway timestamps from the query timestamp, our sampler still has the ability to include it into the TNG and enables information aggregation. Similar to TARGCN, T-GAP, with the help of its *Preliminary GNN* (PGNN), is able to find any temporal associated quadruples related to any entity appearing at any time. However, in its PGNN, it employs three weight matrices, i.e., $\mathbf{W}_{past}$, $\mathbf{W}_{present}$, $\mathbf{W}_{future}$, together with discretized time displacement embeddings $\mathbf{h}_{|\Delta t|}$ to fully express the supporting information coming from the past, the present and the future. We find it redundant to model time difference in this way. In TARGCN, we do not use separate weight matrices during aggregation since our functional time encoder naturally distinguishes the sign of time difference itself. Besides, instead of learning different discretized embeddings to represent different $|\Delta t|$, our model computes the representation of any time difference with shared parameters, thus cutting parameter consumption.

4.4 Generalization to Unseen Timestamps and Irregular Timestamped Data

To prove the robustness of our model, we follow (Goel et al., 2020) to test its ability to predict the links at unseen timestamps. We exclude every quadruple appearing on the 5th, 15th, and 25th day of each month in ICEWS14 to construct a new training set. We randomly split the excluded quadruples into validation and test sets. We compare TARGCN with several recently proposed baselines on this new dataset ICEWS14-unseen, and the results (Table 3) indicate the strong robustness of our model on timestamp generalization. TARGCN greatly outperforms all baseline methods, especially in Hits@1. We attribute this to our strong temporal neighborhood searching mechanism which extensively utilizes information from the whole timeline.

Besides, we construct another new dataset ICEWS14-irregular to validate whether TKGC models can generalize well to the TKG data collected at irregular-spaced timestamps. We randomly sample the snapshots in ICEWS14 and keep the time interval between every two of the sampled neighboring snapshots not greater than 4. We perform TKGC on ICEWS14-irregular and experimental results in Table 3 show that TARGCN is superior in handling data with irregular timestamps. Modeling temporal data with time differences enables TARGCN to distinguish irregular time intervals.

Datasets	ICEWS14-unseen				ICEWS14-irregular			
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
TComplEx	0.461	0.365	0.513	0.644	0.509	0.421	0.558	0.678
TNTComplEx	0.474	0.373	0.524	0.665	0.512	0.429	0.558	0.665
TeMP-SA [1]	-	-	-	-	0.521	0.408	0.583	**0.741**
T-GAP	0.474	0.362	0.532	0.689	0.526	0.428	**0.588**	0.719
TARGCN	**0.578**	**0.518**	**0.607**	**0.692**	**0.552**	**0.496**	0.583	0.667

Table 3: Performance comparison of generalization to unseen timestamps and irregular timestamped data. The best results are marked in bold. Dataset creation process and more discussions are presented in Appendix E.

4.5 Ablation Study

To validate the effectiveness of different model components, we conduct several ablation studies on ICEWS14 and GDELT. We first change the time difference encoding module into an absolute time encoder, e.g., for a (s_q, t_q) and a temporal neighbor (e, t), we learn a representation for t instead

[1] TeMP-SA can not generalize to unseen timestamps.

of $t - t_q$. From Table 4, we observe performance drops on both datasets. This proves the effectiveness of time difference modeling. Next, we adopt random sample in TNG sampling process, which means we do not impose higher probabilities on the temporal neighbors nearer to (s_q, t_q) on the time axis. The performance drops on both datasets (especially on GDELT), indicating that by sampling more neighbors nearer in the temporal context, our model benefits more in learning better representations. Additionally, we conduct another experiment by including all temporal neighbors instead of sampled ones during aggregation. We observe huge performance drops on both datasets, which proves that our sampling strategy helps to exclude noisy information from less-concerned neighbors.

Apart from the first three experiments, we further study how the performance is affected if we constrain the search range of our model in the temporal neighbor sampling process. We constrain the search range to 15, same as the optimal length of temporal snapshots τ used in (Wu et al., 2020) for encoding, and allow our model only to sample temporal neighbors who appear not farther than 15 snapshots away. The performance drops on both datasets (the drop is considerably huge on ICEWS14), and this concludes that our model's performance is closely connected to the amount of temporal information it can utilize. Further details of ablation studies are discussed in Appendix H

Datasets	ICEWS14				GDELT			
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
Absolute Time	0.622	0.556	0.660	0.739	0.502	0.408	0.545	0.678
Random Sample	0.618	0.551	0.656	0.735	0.433	0.312	0.502	0.640
Whole Neighborhood	0.481	0.433	0.501	0.568	0.431	0.312	0.497	0.633
Constrain Search Range	0.420	0.374	0.448	0.498	0.496	0.401	0.541	0.675
TARGCN	**0.636**	**0.576**	**0.672**	**0.746**	**0.515**	**0.423**	**0.557**	**0.689**

Table 4: Ablation studies of TARGCN variants on ICEWS14 and GDELT. The best results are marked in bold.

4.5.1 Temporal Neighborhood Exploration

From Table 4, we observe that if we constrain the search range of TARGCN in the temporal neighborhood sampling process, the performance is strongly affected. Therefore, we further conduct an experiment to study how TARGCN performs while the search range varies. We report our model's performance on ICEWS14 with different search range, namely, 15, 50, 100, 200, 300, and 365 (whole timeline), in Figure 4. For all the metrics, our model's performance improves greatly and constantly as the search range increases. This proves that the effectiveness of TARGCN mainly comes from its superiority in utilizing temporal information. The amount of available temporal information is decisive for our simple-structured model. Compared with the models that only make use of graph snapshots near to the query timestamp t_q, e.g., TeMP, we simplify the model structure but take advantage of as much temporal information as we can. Experimental results in Table 1 show that it is more beneficial in TKGC to utilize temporal information more extensively, instead of designing complex modules.

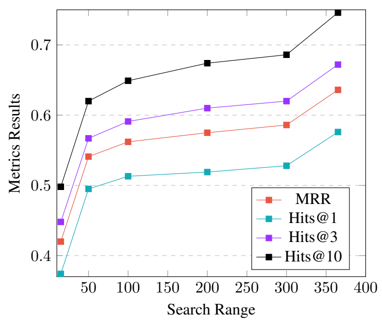

Figure 4: Temporal neighborhood exploration analysis on ICEWS14. Evaluation metrics are filtered MRR and Hits@1/3/10.

5 Conclusion

We propose a simple but powerful time-aware relational graph encoder TARGCN for *Temporal Knowledge Graph Completion* (TKGC). TARGCN employs a *Temporal Neighboring Graph* (TNG) sampling strategy, which enables it to extensively utilize the information from the whole temporal context. Experimental results show that TARGCN achieves state-of-the-art performance on three benchmark TKGC datasets. Besides, TARGCN enjoys an extremely high parameter efficiency. It beats two recently proposed strong GNN-based TKGC methods, i.e., TeMP and T-GAP, with much fewer parameters. Thanks to its time difference learning module and temporal neighbor sampler, TARGCN also shows strong robustness to inferring links on irregular timestamped data or at unseen timestamps. We find that it is not always necessary to incorporate complex modules, e.g., Transformers, into TKG reasoning models. Instead, developing methods to capture more extensive temporal information is more beneficial.

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*.

Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5184–5193. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. 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 2787–2795.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha P. Talukdar. 2018. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2001–2011. Association for Computational Linguistics.

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 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.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2021. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jaehun Jung, Jinhong Jung, and U Kang. 2021. Learning to walk across time for interpretable temporal knowledge graph completion. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 786–795. ACM.

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.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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.

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.

Yunpu Ma, Volker Tresp, and Erik A. Daxberger. 2019. Embedding models for episodic knowledge graphs. *J. Web Semant.*, 59.

Johannes Messner, Ralph Abboud, and İsmail İlkan Ceylan. 2021. Temporal knowledge graph completion using box embeddings. *CoRR*, abs/2109.08970.

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.

Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. 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 2014–2023. JMLR.org.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. 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 8024–8035.

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.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

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.

Volker Tresp, Yunpu Ma, Stephan Baier, and Yinchong Yang. 2017. Embedding learning for declarative memories. In *The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I*, volume 10249 of *Lecture Notes in Computer Science*, pages 202–216.

Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. 2017. Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3462–3471. PMLR.

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.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable reasoning over knowledge graphs for recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5329–5336. AAAI Press.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L. Hamilton. 2020. Temp: Temporal message passing for temporal knowledge graph completion. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5730–5746. Association for Computational Linguistics.

Yiran Xing, Zai Shi, Zhao Meng, Gerhard Lakemeyer, Yunpu Ma, and Roger Wattenhofer. 2021. KM-BART: knowledge enhanced multimodal BART for visual commonsense generation. 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 525–535. Association for Computational Linguistics.

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.

Chenjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020a. Temporal knowledge graph completion based on time series gaussian embedding. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I*, volume 12506 of *Lecture Notes in Computer Science*, pages 654–671. Springer.

Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. 2020b. Inductive representation learning on temporal graphs. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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*.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J. Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 6069–6076. AAAI Press.

Appendix

A Example of Temporal Neighborhood

Figure 5 shows an example of the temporal neighborhood of (s_q, t_q), generated from a TKGC query $(s_q, r_q, ?, t_q)$. We can represent it as $\mathcal{N}_{(s_q,t_q)} = \{(e_1, t_q - 1), (e_2, t_q + 1), (e_3, t_q - 3), (e_4, t_1), (e_5, t_T - 1)\}$. The probability of each temporal neighbor being sampled into (s_q, t_q)'s TNG is determined according to the time difference between t_q and the timestamp of this temporal neighbor (the darker the temporal neighbor shows, the higher the probability).

B Dataset Statistics

Table 5 contains the dataset statistics of all three benchmark datasets and two newly created datasets, i.e., ICEWS14-unseen and ICEWS14-irregular. The data creation process of ICEWS14-unseen and ICEWS14-irregular is discussed in Appendix E.

Dataset	N_{train}	N_{valid}	N_{test}	$\|\mathcal{E}\|$	$\|\mathcal{R}\|$	$\|\mathcal{T}\|$
ICEWS14	$72,826$	$8,941$	$8,963$	$7,128$	230	365
ICEWS05-15	$386,962$	$46,275$	$46,092$	$10,488$	251	$4,017$
GDELT	$2,735,685$	$341,961$	$341,961$	500	20	366
ICEWS14-unseen	$65,679$	$3,420$	$3,420$	$6,601$	230	365
ICEWS14-irregular	$29,102$	$3,555$	$3,607$	$5,093$	210	146

Table 5: Dataset statistics. N_{train}, N_{valid}, N_{test} represent the number of quadruples in the training set, validation set, and test set, respectively. $|\mathcal{T}|$ denotes the number of timestamps, where we take a snapshot of a TKG at each timestamp. All facts in all datasets are denoted in English.

C Implementation Details

We implement all experiments with PyTorch (Paszke et al., 2019) and use a single NVIDIA Tesla T4 for computation. We allow TARGCN to search for neighbors along the whole timeline. The hyperparameter searching strategies are reported in Table 6 and the hyperparameter settings producing the reported experimental results (in Table 1) are presented in Table 7. We use the official implementation of TComplEx, TNTComplEx [2], TeMP [3] and T-GAP [4]. We find that T-GAP has an extremely high memory demand. Training GDELT with T-GAP on a 16GB NVIDIA Tesla T4 causes out-of-memory error even when we set batch size to 1. This is due to its PGNN which constructs a huge temporal associative graph for every entity in training examples.

The training time and the memory usage of TARGCN are reported in Table 8. The training time of TARGCN scales with the number of training quadruples in each dataset. Sampling temporal neighbors for every query subject requires relatively long computation time. This may cause timeout problems during the training process when TARGCN is used to train large-scale datasets (even much larger than GDELT). However, the memory usage of our model remains quite low, which enables training on smaller GPUs.

Table 6: Hyperparameter searching strategy.

Datasets	**ICEWS14**	**ICEWS05-15**	**GDELT**
Hyperparameter			
Embedding Size	{150, 200, 300}	{150, 200, 300}	{150, 200, 300}
# Aggregation Step	{1, 2}	{1, 2}	{1, 2}
Activation Function	{Tanh, ReLU}	{Tanh, ReLU}	{Tanh, ReLU}
Search Range	{15, 100, 200, 300, 365}	{100, 500, 1000, 4017}	{100, 200, 366}
# Temporal Neighbor	{50, 100, 500}	{50, 100, 500}	{50, 100, 500}

Table 7: Best hyperparameter settings on each dataset.

Datasets	**ICEWS14**	**ICEWS05-15**	**GDELT**
Hyperparameter			
Embedding Size	300	200	200
# Aggregation Step	1	1	1
Activation Function	Tanh	Tanh	Tanh
Search Range	365	4017	366
# Temporal Neighbor	100	100	100

[2] https://github.com/facebookresearch/tkbc
[3] https://github.com/JiapengWu/TeMP
[4] https://github.com/sharkmir1/T-GAP

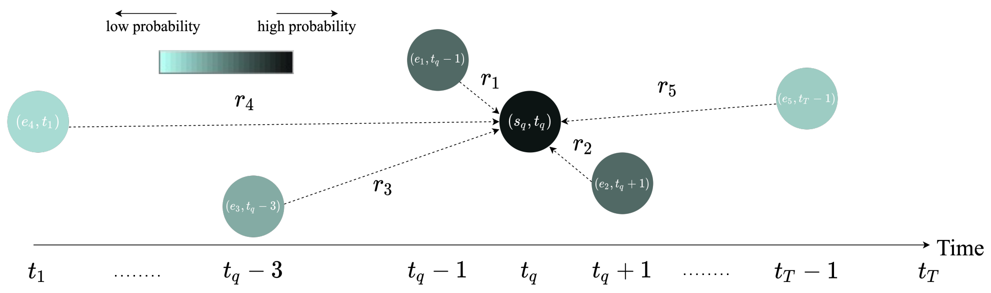

Figure 5: Temporal neighborhood of (s_q, t_q) derived from an object prediction query $(s_q, r_q, ?, t_q)$. We use a dashed line (labeled with relation type) to denote a temporal associated link connecting s_q with its temporal neighbor, e.g., the dashed line labeled with r_4 corresponds to the temporal associated quadruple (e_4, r_4, s_q, t_1). A temporal neighbor with darker color is assigned a higher probability to be sampled into (s_q, t_q)'s TNG. Since (e_1, t_q-1) and (e_2, t_q+1) has the same temporal distance from t_q, they are assigned with the same sampling probability (denoted with the same color darkness).

Table 8: Computational budget of TARGCN on benchmark datasets.

Datasets	ICEWS14	ICEWS05-15	GDELT
GPU Memory Usage (MB)	1,375	1,385	1,261
Train Time/ Epoch (s)	405	9,900	145,200
# Train Epochs	100	100	10

D Validation Results

In Table 9, we report the experimental results of TARGCN on validation sets on all three benchmark datasets. The results are produced by the same trained models reported in Table 1.

E Further Details of Generalization to Unseen Timestamps and Irregular Timestamped Data

We choose four strong baselines to compare with TARGCN, namely, TComplEx (Lacroix et al., 2020), TNTComplEx (Lacroix et al., 2020), TeMP-SA (Wu et al., 2020), and T-GAP (Jung et al., 2021). We choose TeMP-SA since it is reported with better results on ICEWS14 (newly created datasets are based on ICEWS14). We can not perform unseen timestamps generalization with TeMP-SA since it requires the unavailable KG snapshot $\mathcal{G}(t_q)$ for every link prediction query $(s_q, r_q, ?, t_q)$.

E.1 Unseen Timestamps Generalization

We do not use the same unseen timestamps generalization datasets proposed in (Goel et al., 2020) and (Jung et al., 2021), since they do not release their dataset. We follow (Goel et al., 2020) and create ICEWS14-unseen by ourselves. We exclude every quadruple appearing on the 5th, 15th, and 25th day of each month in ICEWS14 to construct a new training set. We randomly split the excluded quadruples into validation and test sets. We make sure that every entity appearing in the validation and test sets is seen in the training set.

By comparing the results in Table 1 and Table 3, we observe that the performance improvement of TARGCN becomes even much larger on ICEWS14-unseen than on the original dataset. TARGCN achieves a relative improvement of 21.94% on MRR compared with T-GAP and TNTComplEx. More surprisingly, it also achieves a relative improvement of 43.09% on Hits@1 compared with the strongest baseline T-GAP on unseen timestamps generalization. This proves the extremely strong robustness of our model to link inference at unseen timestamps.

E.2 Performance on Irregular Timestamped Data

We sample the KG snapshots from the original ICEWS14 dataset. The value of the time interval between every two neighboring snapshots can be randomly assigned either to 1, 2, 3, or 4. In this way, we create a dataset simulating that the TKG data is observed and collected at irregular-spaced timestamps.

TARGCN enlarges the performance gap between itself and other baselines, compared with the results regarding TKGC on the original dataset reported in Table 1. Besides, we observe that TeMP-SA and T-GAP outperform TNTComplEx on ICEWS14-irregular, while they perform worse on the original dataset. This is due to their time displacement temporal encoders which learn different tempo-

Datasets	ICEWS14				ICEWS05-15				GDELT			
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
TARGCN	0.647	0.591	0.679	0.748	0.705	0.641	0.742	0.821	0.510	0.418	0.552	0.685

Table 9: Temporal knowledge graph completion results on the validation sets of three benchmark datasets. Evaluation metrics are filtered MRR and Hits@1/3/10.

ral embeddings for different time intervals. For TARGCN, it employs a time difference temporal encoder that maps time-aware entity representations with the explicit value of time differences, thus being able to capture accurate temporal information provided by irregular timestamped data.

F Functional Time Encoder Details

For a TKGC query $(s_q, r_q, ?, t_q)$, to compute the time-aware representation of a sampled temporal neighbor (e, t), we employ the time encoder following (Xu et al., 2020b) to generate the time difference representation $\mathbf{h}_{t-t_q}$:

$$\begin{aligned}\mathbf{h}_{t-t_q} =& \mathbf{\Phi}(t, t_q)\\ =& \sqrt{\frac{1}{d_t}}[cos(\omega_1(t-t_q)+\phi_1), ...,\\ & cos(\omega_{d_t}(t-t_q)+\phi_{d_t}))],\end{aligned} \tag{4}$$

where ω_1 to ω_{d_t} are frequency components, ϕ_1 to ϕ_{d_t} are phase components, and d_t denotes the embedding size of time difference representations. The frequency components and phase components are learnable and shared in representation computation for all time differences. For more details please refer to (Xu et al., 2020b).

G Parameter Efficiency Analysis Details

Datasets	ICEWS14			
Model	# Parameters	MRR	Parameter ↑	MRR ↓
TARGCN	1229100	0.627	-	-
T-GAP	1912350	0.610	55.59%	2.71%
TeMP-SA	1264640	0.607	2.89%	3.19%
TeMP-GRU	1413408	0.601	15.00%	4.15%

Table 10: Parameter efficiency comparison on ICEWS14. We adopt relative change to define the increase in parameter numbers and the drop in MRR.

Similar to Table 2, Table 10 summarizes the number of parameters as well as performance difference on ICEWS14. For TARGCN, the model producing results in Table 1 has more parameters than T-GAP and TeMP. Therefore, we decrease the embedding size of TARGCN to 150 so that its parameter number becomes the smallest among all models. We keep T-GAP and TeMP with their optimal parameter settings and compare them with TARGCN. From Table 1 and Table 10, we observe that even when we decrease the embedding size of TARGCN from 300 to 150, our model still performs well (MRR drops from 0.635 to 0.627), and it still outperforms T-GAP and TeMP on ICEWS14. TeMP variants show the worst performance, even when they have more parameters than TARGCN. T-GAP performs better than TeMP variants. However, it uses 55.59% more parameters than TARGCN, while it is beaten with a 2.71% performance drop.

All the points in Figure 3 are based on the results in Table 11. Note that we control the number of parameters only by changing embedding size, without changing any other hyperparameters or model structures.

Datasets	ICEWS14		
Model	Embedding Size	# Parameters	MRR
TARGCN	100	799400	0.605
	150	1229100	0.627
	200	1678800	0.629
	<u>300</u>	<u>2638200</u>	<u>0.636</u>
T-GAP	50	928675	0.582
	<u>100</u>	<u>1912350</u>	<u>0.610</u>
	200	2951025	0.596
TeMP-SA	64	611840	0.595
	<u>128</u>	<u>1264640</u>	<u>0.607</u>
	256	2928640	0.618

Table 11: Experimental results as well as the number of parameters that lead to Figure 3. Underlined results are taken from Table 1.

H Ablation Study Details

Although in all ablation studies, both ICEWS14 and GDELT witness the same tendency in performance drops, the sensitivity of different model components are different. We find that the absolute time encoder and the whole temporal neighborhood aggregation bring similar influences on the performance on both datasets. However, random sample in TNG imposes a much larger impact on the performance on GDELT than it on ICEWS14, while constraining search range affects ICEWS14 more greatly. This can also be explained with the dataset characteristics discussed in Section 4.2. ICEWS14 is much sparser, which implies that in a small search range around the query timestamp, we might not be able to find most supporting temporal neighbors. However, for GDELT, its dense nature leads to the fact that almost all supporting temporal neighbors can be found in this small search range. And if we allow the model to randomly sample temporal neighbors from the whole timeline, the gain brought by the increasing number of meaningful temporal neighbors helps the model to perform much better on ICEWS14. On the contrary, for GDELT, sampling too many temporal neighbors far from the query timestamp incurs huge noise, thus leading to degenerated performance.