Modeling Precursors for Temporal Knowledge Graph Reasoning via Auto-encoder Structure

Yifu Gao¹, Linhui Feng¹, Zhigang Kan¹, Yi Han², Linbo Qiao¹* and Dongsheng Li¹*
¹Science and Technology on Parallel and Distributed Processing Laboratory, National University of Defense Technology, Changsha, China
²College of Meteorology and Oceanography, National University of Defense Technology
{gaoyifu, linhufeng, kanzhigang13, hanyi12, qiao.linbo, dsl}@nudt.edu.cn

Abstract

Temporal knowledge graph (TKG) reasoning that infers missing facts in the future is an essential and challenging task. When predicting a future event, there must be a narrative evolutionary process composed of closely related historical facts to support the event’s occurrence, namely fact precursors. However, most existing models employ a sequential reasoning process in an auto-regressive manner, which cannot capture precursor information. This paper proposes a novel auto-encoder architecture that introduces a relation-aware graph attention layer into transformer (rGalT) to accommodate inference over the TKG. Specifically, we first calculate the correlation between historical and predicted facts through multiple attention mechanisms along intra-graph and inter-graph dimensions, then constitute these mutually related facts into diverse fact segments. Next, we borrow the translation generation idea to decode in parallel the precursor information associated with the given query, which enables our model to infer future unknown facts by progressively generating graph structures. Experimental results on four benchmark datasets demonstrate that our model outperforms other state-of-the-art methods, and precursor identification provides supporting evidence for prediction.

1 Introduction

The knowledge graph is crucial to many artificial intelligence (AI) applications. Since most knowledge graphs are far from complete, knowledge graph reasoning becomes a critical task and has been extensively studied on static graphs [Yang et al., 2015]. However, the rapidly growing facts on the knowledge graph show complex dynamic characteristics, which creates the need for introducing the concept of temporal knowledge graph (TKG) to infer missing facts. Unlike the static KG, the fact on TKG is composed of a quadruple, i.e., (subject, relation, object, timestamp). Due to the incompleteness and dynamic changes in a graph structure, TKG reasoning is a complex task worth researching.

*Corresponding Author

Figure 1: An example of fact precursors related to the query on the ICEWS14 dataset.

Given a TKG from $t_0$ to $t_T$, there are two main settings for TKG reasoning, interpolation and extrapolation [Jin et al., 2020]. The former setting aims to learn dynamic representations and infer missing facts for timestamp $t_0<t<t_T$, while the latter attempts to predict future facts for timestamp $t>t_T$, even the corresponding graph snapshots within the time interval are unavailable. A general solution to the extrapolation reasoning problem is to learn the dynamic representation of all historical entities and adopt a temporal point process to predict future facts [Trivedi et al., 2017; Trivedi et al., 2019]. However, they can not model concurrent events occurring within the same time window.

Some recent attempts capture the structural dependencies among concurrent facts and sequential patterns across temporally adjacent facts in an auto-regressive manner. Specifically, RE-NET [Jin et al., 2020] adopts RGCN [Schlichtkrull et al., 2018] to aggregate concurrent events within the same timestamp and summarizes information of the past event sequences with GRU [Cho et al., 2014]. HIP [He et al., 2021] comprehensively transmits historical information in terms of time, structure, and repetition through three novel scoring functions. Instead of encoding related facts for the given query, RE-GCN [Li et al., 2021b] treats TKG as a KG sequence and encodes all historical facts into entity and relation evolutionary representations.

Another line of method argues that only a few historical facts are useful for a specific prediction, which is the most related work with us. CyGNet [Zhu et al., 2021] uses a copy-generation mechanism to model the repetitive historical facts. Nevertheless, only considering 1-hop repetitive paths will lose massive useful query-relevant information. Thus, CluSTeR [Li et al., 2021a] extracts both 1-hop repetitive and
non-repetitive paths related to the query and performs temporal reasoning on the sequence of subgraphs constructed from clues. Furthermore, xERTE [Han et al., 2021] employs a sequential reasoning process over local inference subgraphs and provides interpretability for their predictions. However, they both use heuristic rules to focus on single independent causes close to the current timestamp, making it challenging to discover latent connections between relevant historical facts.

In fact, there must have been an evolutionary process associated with the predicted fact along history to suggest the event’s occurrence. For example, as shown in Figure 1, when facing the query (Congress (United States), Impose sanctions, ?, 2014-12-01), several historical facts (red dashed lines) involved by the subject entity of given query constitute a narrative fact segment, which helps people capture the evolutionary law behind the Impose sanctions event and find the answer Iran to the query. We discover this underlying phenomenon and introduce the concept of fact precursors to describe these query-relevant historical facts with either continuous or discontinuous timestamps. However, previous models adopt a sequential reasoning structure, which cannot capture the close connections between historical facts. Recently transformer [Vaswani et al., 2017] has been widely used in various fields, such as computer vision and machine translation, thanks to its ability to compute the correlation between arbitrary positions in a sequence. Inspired by this mechanism, we treat the TKG as a static KG sequence and first propose an auto-encoder architecture, rGalT, which employs transformer based on a relation-aware graph attention layer (RGAL) to fully explore the precursor information related to the predicted fact. More specifically, in the intra-graph component, we use RGAL to calculate query-dependent attention scores by considering the entity and relation factors of the given query and guide the model to aggregate neighbor information of the query’s interest within the same timestamp. As for the inter-graph part, we adopt a transformer encoder to discover diverse fact segments within the graph snapshot sequence. Finally, the generative method in the inference component enables our model to decode precursor information related to the given query and infer future missing facts.

Overall, this paper makes the following contributions: 1) We introduce the concept of fact precursors in the TKG reasoning task to reveal the underlying evolutionary patterns behind the event. Fact precursors can be viewed as a narrative fact segment composed of several query-relevant historical facts with continuous or discontinuous timestamps. 2) To the best of our knowledge, we are the first to propose an auto-encoder structure, rGalT, to accommodate extrapolation inference over the TKG. Compared to traditional auto-regressive methods, rGalT can more fully capture interactions between fact precursors. 3) Our experiments achieve state-of-the-art performance in four widely used datasets, thus demonstrating the validity of our model. Precursor identification can provide supporting evidence for the results.

2 Related Work

Static KG reasoning. In recent years, we have witnessed increasing interest in knowledge reasoning. Furthermore, there are three main categories: KG embedding-based reasoning, rule-based reasoning, and relation path-based reasoning. KG embedding aims to map the entities and relations into continuous vector space and score the probabilities with embeddings, such as the translating model [Bordes et al., 2013] and the semantic matching model [Yang et al., 2015; Trouillon et al., 2016]. Rule-based methods utilize the instantiation rules [Galárraga et al., 2015] or inject the rules into models [Guo et al., 2018; Zhang et al., 2019], to improve reasoning accuracy. Moreover, relation path-based reasoning finds the target answer on the graph with reinforcement learning [Xiong et al., 2017]. However, static KG reasoning methods ignore the temporal information, which makes them not applicable to temporal knowledge graph reasoning.

Temporal KG reasoning. There are two main settings for TKG reasoning, interpolation and extrapolation. The former setting [Jiang et al., 2016; Dasgupta et al., 2018; García-Durán et al., 2018] aims to learn dynamic representations and infer missing facts for the historical timestamps. In contrast, this paper focuses more on the latter setting, which attempts to predict facts in the future. Previous models [Trivedi et al., 2017; Trivedi et al., 2019] consider all historical facts and adopt a temporal point process to infer the occurrence of future facts. However, they can not model concurrent facts at the same timestamps. Accordingly, some earlier attempts consider structural and temporal patterns over the TKG. RE-NET [Jin et al., 2020] uses an aggregator and GRU to transmit information of the past event sequences sequentially. HIP [He et al., 2021] passes historical information selectively from temporal, structural, and repetitive perspectives. Instead of facing each query, RE-GCN [Li et al., 2021b] encodes all historical facts into entity and relation evolutionary representation for future fact prediction. Besides, some recent attempts extract some related historical information for each query. CyGNet [Zhu et al., 2021] proposes a copy-generation method to model repetitive facts with the same entity and relation as each query. CluSTEr [Li et al., 2021a] uses reinforcement learning to search for 1-hop repetitive and non-repetitive facts related to the query and constructs them into temporal subgraphs for reasoning. Moreover, xERTE [Han et al., 2021] conducts a sequential reasoning process on query-relevant subgraphs that are dynamically expanded by an iterative sampling of temporal neighbors and temporal relational attention propagation. The inference graph can be seen as a graphical explanation. However, most of the above models use an auto-regressive approach, which makes it challenging to capture the underlying evolutionary process behind the occurrence of events.

3 Our Model

In this section we start with the notation and task definition, then we provide an overview in Section 3.2 and explain each module from Section 3.3 to 3.5.

3.1 Notations and Task Definition

A temporal knowledge graph (TKG) $G$ can be viewed as a multi-relational, directed graph with timestamped edges between nodes (entities), which can be formalized as a sequence
of knowledge graph snapshots ordered by timestamp, i.e., \( G = \{G_1, G_2, \ldots, G_t, \ldots\} \). Each event (fact) in the snapshot \( G_t \) can be represented as a quadruple \( (s, r, o, t) \) or \( (s, t, o, t) \), corresponding to subject entity \( s \in \mathcal{E} \), relation type \( r \in \mathcal{R} \), object entity \( o \in \mathcal{E} \) and timestamp \( t \in \mathcal{T} \), where \( \mathcal{E} \), \( \mathcal{R} \) and \( \mathcal{T} \) represent the sets of entities, relationships and timestamps, respectively.

The purpose of the extrapolation reasoning task over TKG is to infer the missing object entity \( o \) given the query \( (s, r, ?, t + \Delta t) \), or missing subject entity \( s \) of the query \( (?, r, o, t + \Delta t) \). It is worth noting that the missing entity \( s \) or \( o \) on \( G_{t+\Delta t} \) is inferred using the historical facts of \( \{G_1, G_2, \ldots, G_t\} \), while the event information in the time period \( \Delta t \) is unknown. Specifically, following \cite{Jin2020}, we only use the ground truth of the training set when making inferences on the validation and test sets.

### 3.2 Model Overview

As shown in Figure 2, our model consists of three parts: Intra-graph Component, Inter-graph Component, and Inference Component.

At the intra-graph component, we use the relation-aware graph attention layer to compute an attention score for each query-relevant entity and then perform weighted aggregation of these neighboring entities. As for the inter-graph component, we employ the transformer encoder to interact with facts at different timestamps, with the aim of capturing multiple fact fragments from the graph sequence. Finally, at the inference component, we adopt the masked self-attention layer and encoder-decoder attention layer of the transformer decoder to identify fact segments with high relevance to the given query, called fact precursors. Then we decode in parallel the relation and object entity associated with the given query to predict what happens next.

### 3.3 Intra-graph Component

In this section, we propose a relation-aware graph attention layer (RGAL) to calculate the query-dependent score for each historical fact that the subject entity of the given query participates in. Unlike RGCN \cite{Schlichtkrull2018}, RGAL assigns different correlation weights to neighboring entities within the same timestamp by considering the entity and relation factors of the given query.

In addition, considering that the fact (event) meanings vary with the relationship, we incorporate the relation embedding factors of the given query. The purpose of the extrapolation reasoning task over TKG is to infer the missing object entity \( o \) corresponding to subject entity \( s \), relation \( r \), and timestamp \( t \) as shown below:

\[
 u_{so} = \alpha^T \sigma \left( W \left[ e_s | e_r | e_o \right] \right),
\]

where \( u_{so} \) is the attention score of the fact \( (s, r, o) \) in graph snapshot \( G \) relative to the query \( (s, r, ?, t + 1) \); \( e_s, e_r, e_o \in \mathbb{R}^d \) are embeddings corresponding to subject entity and object entity, and \( e_r, e_{oy} \in \mathbb{R}^d \) are the relation embeddings; \( \| \) is the concatenation operation and \( W \in \mathbb{R}^{d \times 4d} \) is a weight matrix applied to each node in the graph; \( \alpha \in \mathbb{R}^d \) is used as a feed-forward layer to parameterize the attention function and \( \sigma \) is a tanh non-linearity function to compute attention weights. We compute the normalized attention score using the softmax function and obtain the final output \( x_s \) as follows:

\[
 \alpha_{so} = \frac{\exp(u_{so})}{\sum_{w \in N_s} \exp(u_{sw})}, \quad x_s = \sum_{o \in N_s} \alpha_{so} e_o,
\]

where \( N_s \) is the set of immediate neighbors of entity node \( s \), and \( \alpha_{so} \) denotes the correlation fraction of \( o \) with respect to \( s \) in snapshot \( G \). The output representation \( x_s \in \mathbb{R}^d \) aggregates factual information about graph snapshot \( G \) associated with the given query.
3.4 Inter-graph Component

We find that there exists a tighter connection called the fact segment within the historical fact sequence. To capture various fact segments, we use the encoder part of the transformer to dynamically interact with graph aggregation information at different timestamps. Specifically, for the entity \( s \) of given query, the encoder maps an input sequence of aggregated representations \( \{x_{s,t,m+1}, x_{s,t,m+2}, \ldots, x_{s,t} \} \), \( x_s \in \mathbb{R}^d \), to a sequence of continuous representations \( \{h_{s,t,m+1}, h_{s,t,m+2}, \ldots, h_{s,t} \} \), \( h_s \in \mathbb{R}^d \), where \( x_s \) for each timestamp is obtained by (2), \( m \) represents the previous time steps related to the entity \( s \). We denote the input and output sequences as \( X_s, H_s \in \mathbb{R}^{m \times d} \) respectively.

As shown in Figure 2(b), each layer of the transformer encoder is identical and divided into two sub-layers, a multi-head self-attention layer and a feed-forward network. For the first sub-layer, we set up the self-attention mechanism of \( K \) heads to capture richer position information. And we use three independent linear transformation matrices \( \mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_o \in \mathbb{R}^{d \times d} \) to transform the input representations \( X_s \) into queries, keys, and values of the \( i \)-th scaled dot-product attention head \( (i = 1, 2, \ldots, K) \). The specific model details are shown below:

\[
\mathbf{Z}_s^i = \text{Softmax}(\frac{(\mathbf{X}_s \mathbf{W}_q^i)^T (\mathbf{X}_s \mathbf{W}_k^i)^T}{\sqrt{d}}) (\mathbf{X}_s \mathbf{W}_o^i),
\]

where \( \mathbf{Z}_s^i \in \mathbb{R}^{m \times d} \) is the output representation of corresponding attention head, \( d^i = \frac{d}{K} \). When we get the output sequence of all attention heads in parallel, we cascade them to get the final output \( \mathbf{Z}_s \in \mathbb{R}^{m \times d} \).

\[
\mathbf{Z}_s = \text{Concat}(\mathbf{Z}_s^1, \mathbf{Z}_s^2, \ldots, \mathbf{Z}_s^K).
\]

The second sub-layer is a fully connected feed-forward network, which consists of two linear transformations with a ReLU activation in between:

\[
\mathbf{H}_s = \text{ReLU}((\mathbf{Z}_s \mathbf{W}_1 + \mathbf{b}_1) \mathbf{W}_2 + \mathbf{b}_2),
\]

where \( \mathbf{W}_1 \in \mathbb{R}^{d \times d}, \mathbf{b}_1 \in \mathbb{R}^d \) are the parameters of the first linear layer, and \( \mathbf{W}_2 \in \mathbb{R}^{d \times d}, \mathbf{b}_2 \in \mathbb{R}^d \) are the parameters of second. Although the model network is the same for different time steps related to the entity \( s \), we use the positional encoding to learn the sequential temporal information at different time steps.

3.5 Inference Component

In order to find the fact precursors related to the given query \((s, r, ?, t + 1)\), we use the first two sub-layers of decoder, masked multi-head self-attention, and encoder-decoder self-attention, to model the sequence of hidden states \( \mathbf{H}_s \) from the previous module. Then we decode the precursor information in parallel by MLP to get the probabilities of relations and object entities, respectively.

First, we take the embeddings of the entity \( e_s \in \mathbb{R}^d \) and the relation \( e_r \in \mathbb{R}^d \) as the input sequence \( Q \in \mathbb{R}^{n \times d} \) of the masked multi-head self-attention layer, which differs from Eq. 3 by adding the mask matrix \( \mathbf{M} \in \mathbb{R}^{n \times n} \). The specific function is shown below:

\[
\mathbf{Z}_q = \text{Softmax}(\frac{(\mathbf{QW}_q^i)(\mathbf{QW}_k^i)^T}{\sqrt{d}} + \mathbf{M}(\mathbf{QW}_o^i),
\]

where \( \mathbf{Z}_q \in \mathbb{R}^{n \times d} \) is the final output of this layer after cascading the outputs of each head using Eq. 4, the meanings of the parameters are consistent with those of the self-attention mechanism in the above section. Primarily, \( \mathbf{M} \) is represented as follows:

\[
\mathbf{M}[a, b] = \begin{cases} 0, & \text{if } a = b, \\ \infty, & \text{others}. \end{cases}
\]

and we also get the concatenation form \( \mathbf{H}_q \in \mathbb{R}^{n \times d} \) of all heads by Eq. 4. This process matches the encoder’s output \( \mathbf{H}_s^{(l)} \) to the decoder’s input \( \mathbf{Z}_q \), allowing the subject entity and relation of the predicted fact to decide which fact segment is relevant to themselves. Hence, \( \mathbf{h}_{q,s}, \mathbf{h}_{q,e} \in \mathbb{R}^d \) are decoupled hidden states contained in the \( \mathbf{H}_q \), corresponding to the outputs of subject entity \( s \) and relation \( r \) at this sub-layer. Given a query \((s, r, ?, t + 1)\), we feed the above representation at the \( l \)-th layer into a multi-layer perceptron (MLP) decoder parameterized by \( \mathbf{w}_o \) to predict the occurrence probabilities of all entities at timestamp \( t + 1 \), i.e.,

\[
p(o \mid s, r, G_{t-m+1:t}) = \text{Softmax}(\langle e_s : h_{s,o}^{(l)} : e_r \rangle^T \cdot \mathbf{w}_o),
\]

where \( \mathbf{w}_o \in \mathbb{R}^{3d \times |E|}, e_s, e_r \) are learnable embedding vectors. \( h_{s,o}^{(l)} \) is the object entity representation of precursor information. Similarly, we define probabilities of all relations associated with the subject entity \( s \) as follows:

\[
p(r \mid s, G_{t-m+1:t}) = \text{Softmax}(\langle e_s : h_{s,r}^{(l)} \rangle^T \cdot \mathbf{w}_r),
\]

where \( \mathbf{w}_r \in \mathbb{R}^{2d \times |R|}, h_{s,r}^{(l)} \) is the relation representation of precursor information.

With the above three modules, we can infer the missing entity \( o \) of the given query \((s, r, ?, t + 1)\), provided that the graph snapshot sequence \( \{G_1, G_2, \ldots, G_t\} \) is known. While the extrapolation reasoning task is to infer the missing entity at the future moment \( t + \Delta t \), even if the ground truth within \( \Delta t \) is unknown. Therefore we propose a subgraph generation algorithm, which can generate new graphs sequentially within \( \Delta t \) to handle the multi-step reasoning problem. The exact procedure of the algorithm is described in Appendix A.
### 3.6 Training Objective

Giving a query \((s, r, ?, t')\) to infer missing object entity \(o\) can be considered as a multi-class classification task, where each class corresponds to each entity. Similarly, relation prediction of a given subject entity \(s\) can also be viewed as a multi-class classification task where categories correspond to each relation. Here the loss function can be defined as follows:

\[
    \mathcal{L} = \sum_{(s, r, o, t') \in G_{14}} \log p(o | (s, r, o')) + \lambda \log p(t' | s, r'),
\]

where \((s, r, o, t') \in G_{14}\) represents the known facts in the training set, \(p(s)\) are the probability scores obtained from Eq.9 and Eq.10, and \(\lambda\) is a hyperparameter that regulates the importance of the two classification tasks.

### 4 Experiments

#### 4.1 Experimental Setting

**Datasets and Metrics.** There are five typical datasets commonly used in previous studies [Li et al., 2021a; Li et al., 2021b], namely, ICEWS14, ICEWS05-15, ICEWS18 [Boschee et al., 2015], GDELT [Leetaru and Schrodt, 2013] and YAGO [Mahdisoltani et al., 2015]. More details about datasets are shown in Table 4 of Appendix B. We split all datasets into three sets, with a proportion of train(80%), valid(10%) and test(10%) by timestamps following [Li et al., 2021b]. Mean Reciprocal Rank (MRR) and Hits@\([1, 3, 10]\) are employed as the metrics in our experiments. We compare the experimental results under the raw setting [Li et al., 2021a; Li et al., 2021b] rather than the filtered setting used in [Jin et al., 2020; Zhu et al., 2021; He et al., 2021]. A detailed explanation is provided in Appendix B.

**Baselines.** Our model is compared with two categories of models, i.e., static KG reasoning models and TKG reasoning models. For static KG models, we select DisMult [Yang et al., 2015], R-GCN [Schlichtkrull et al., 2018], ConvE [Dettmers et al., 2018], RotatE [Sun et al., 2019]. For temporal models under the interpolation setting, we choose HyTE [Dasgupta et al., 2018], TTransE [Jiang et al., 2016] and TA-DistMult [García-Durán et al., 2018]. Then, we compared most of the methods under the extrapolation setting, including Know-Evolve [Trivedi et al., 2017], RGCN [Jin et al., 2020], CyGNet [Zhu et al., 2021], RE-NET [Jin et al., 2020], RE-GCN [Li et al., 2021b], xERTE [Han et al., 2021] and CluStEr [Li et al., 2021a]. Note that xERTE and CluStEr use previous all ground truth when inferencing in the validation and test sets, while other methods are only given a training set as ground truth to conduct the multi-step inference.

### 4.2 Results on TKG

Without loss of fairness, we will compare the experimental results separately, as shown in Tables 1 and 2. Table 1 illustrates that rGalT consistently outperforms the baselines on the three ICEWS datasets and GDELT. Especially on the ICEWS14 dataset, rGalT achieves the improvements of 4.2% in Hits@1, 2.9% in Hits@3 over the best baseline.

Specifically, rGalT performs much better than all static methods (the first block) and temporal models under the interpolation setting (the second block) since our model considers temporal factors and learns historical evolution patterns. For those temporal models under the extrapolation setting (the third block), Know-Evolve performs the least well because it ignores the mutual interactions of concurrent events. The performance of RGCN is worse than other models except for Know-Evolve because it models all history from several latest timestamps, while others consider a more extended history sequence. CyGNet and RE-NET achieve relatively good results for the inference task, on the grounds that CyGNet can learn repetitive facts of a given query, and RE-NET propagates the local and global structural information related to the query sequentially over the temporal pattern. There is no doubt that rGalT achieves better results because it can find more useful historical facts related to the prediction by modeling fact precursors. Note that rGalT outperforms RE-GCN on all datasets except YAGO. Considering evolutionary representations of all historical facts introduces noise to affect performance, which may cause RE-GCN to be less effective than rGalT on most datasets. Besides, the time interval (1 year) of YAGO is much larger than the other datasets resulting in more structural dependencies of the KG at each timestamp, which is friendly to some models with static properties, such as RE-GCN. Overall, rGalT is more capable of handling datasets with obvious dynamic features.

As shown in Table 2, the effects of rGalT w.GT and CluS-TeR on these three datasets exceed xERTE, probably because
the inference subgraph of xERTE focuses more on direct causes leading to the predicted event, while the precursor information and the clues contain more query-related factual information. Despite the CluSTeR achieving better results on two ICEWS datasets, it does not follow the assumptions of the extrapolation reasoning task and was unable to make inferences only using the ground truth of the training set. In addition, rGalT’s best performance on the GDELT dataset verifies our motivation that rGalT model can discover relevant historical facts at farther time steps apart because frequent changes of historical facts in the GDELT dataset (15-minute time intervals) may cause related facts to be far away in the time window.

4.3 Ablation Studies

To facilitate generalized conclusions, we conduct ablation experiments on ICEWS14 and YAGO datasets with different time intervals, 24 hours and one year, respectively. We discuss the effects of each component as follows:

Impact of Intra-graph Component. As shown in Table 3, we remove the relation-aware graph attention layer, denoted as - RGAL. It can be observed that - RGAL has a more significant impact on the final results, suggesting that introducing RGAL makes the transformer capable of capturing query-relevant fact information inside the graph. +RGCN indicates that we replace RGAL method with a relational graph convolutional network (RGCN). The result performance implies our method has the advantage of exploring neighboring entities related to the query.

Impact of Inter-graph Component. +GRU indicates that the GRU method is used instead of the transformer encoder to learn the temporal patterns among different graphs. It can be observed that the performance of +GRU decreases a lot in two datasets, demonstrating the transformer encoder’s importance in our model. There are several facts helpful for future prediction occurring at more distant time steps. Compared to the traditional auto-regressive model, the transformer will allow each fact within a time window to focus on other time-stamped facts and form narrative fact segments with highly correlated facts to avoid forgetting parts of the information during the time shift.

Impact of Inference Component. +average in Table 3 denotes a variant of rGalT directly using the average representations of transformer encoder outputs as the input of MLP. It can observe that the performance of +average decreases rapidly, which sufficiently indicates the necessity of self-attention mechanism. Attention methods help rGalT capture query-related precursor information from various fact segments. - multi-step represents rGalT keeps its last history in the training set and does not update history during inference. It can be seen that the performance is worse than the initial rGalT, which proves the effectiveness of our subgraph generation algorithm.

4.4 Case Study

rGalT provides an explanation for model predictions through the discovery of precursor information. Precursor information can be viewed as query-relevant narrative fact segments that reveal the evolutionary process behind the occurrence of an event. We study the query selected from the ICEWS14 test set (China, Express intent to settle dispute, ?, 364), and Figure 3 shows how rGalT finds the fact precursors of this query for prediction. Here, the previous m time steps refer to the timestamp from 354 to 363.

The upper part of Figure 3 shows the subgraphs composed of partial facts related to the query at different timestamps. The weights of edges in the subgraphs are computed by RGAL, representing the correlation between facts and entity China. Encoder Self-attention aims to discover fact segments within the historical sequence. The timestamps of fact segments are extracted as follows according to the self-attention matrix in the lower left part of Figure 3: ⟨356,360,361⟩, ⟨357,360,361⟩, ⟨358,360,361,363⟩. With our observations, the facts at ⟨360,361⟩ play a key role in this prediction because they involve multiple fact segments. Encoder-Decoder self-attention aims to capture the correlation between query and fact segments. We can see that the given query has a significant link with the fact segment containing the timestamp ⟨363⟩ from the lower right part. Thus, the facts with high query-dependent scores at timestamp ⟨358,360,361,363⟩ can be considered as precursors to the query (triples with solid lines). Fact precursors suggest that there is a strong willingness to cooperate between China and Japan, which can be regarded as antecedent events for (China, Express intent to settle dispute, ?, 364).

5 Conclusion

In this paper, we are the first to propose a novel precursor-aware transformer, rGalT, which employs transformer based on a relation-aware graph attention layer to model precursor information over the TKG. We use multiple attention mechanisms to capture diverse fact segments from the graph se-
Algorithm 1 Subgraph generation algorithm

Input:
- Known graph sequence \( \{G_1, G_2, \ldots, G_t\} \);
- Given query \((s, r, ?, t + \Delta t)\) with missing object entity at timestamp \(t + \Delta t\).

Output:
\[
\{\hat{G}_{t+1}, \hat{G}_{t+2}, \ldots, \hat{G}_{t+\Delta t-1}\}.
\]

1: \(t' \leftarrow t + 1\)
2: while \(t' < t + \Delta t\) do
3:     for every \(r \in R\) do
4:         Get \(p(r|s, G_1:t, \hat{G}_{t+1:t'-1})\) of relation \(r\) with subject entity \(s\) by Eq. 10
5:     for every \(o \in E\) do
6:         Get \(p(o|s, r, G_1:t, \hat{G}_{t+1:t'-1})\) of object entity \(o\) with subject entity \(s\) and relation \(r\) by Eq. 9
7:     Calculate the probability of triple \((s, r, o)\) by Eq. 12.
8: end for
9: end for
10: Pick top-\(k\) triples at \(t'\)
\[
\{(s_1, r_1, o_1, t'), \ldots, \{s_{k}, r_{k}, o_{k}, t'\}\}
\]
\[
\hat{G}_{t'} \leftarrow \{(s_1, r_1, o_1, t'), \ldots, \{s_{k}, r_{k}, o_{k}, t'\}\}
\]
\[
t' \leftarrow t' + 1
\]
11: end while
12: return \(\{\hat{G}_{t+1}, \ldots, \hat{G}_{t+\Delta t-1}\}\)

sequence and find the fact segment most closely related to the query through join-self-attention methods. Besides, we decode precursor information in parallel to infer missing facts and construct a subgraph generation algorithm to handle the extrapolation task. Experimental results on five datasets demonstrate the effectiveness of our method, and precursor discovery provides interpretability for reasoning results.

Appendices

A Subgraph Generation Algorithm

As shown in Algorithm 1, given all the past facts \(G_{1:t}\), we first calculate the conditional probability of each relation relative to the query subject entity(line 3-4), then calculate the conditional probability of each object entity close to the relation and subject entity(line 5-6), and finally obtain the likelihood of the triple \((s, r, o)\) by the following formula:

\[
p(s, r, o|G_{1:t}) = p(o|s, r, G_{1:t}) \cdot p(r|s, G_{1:t}) \cdot p(s|G_{1:t}),
\]

(12)

the subgraph we inferred is composed of facts related to \(s\), so \(p(s|G_{1:t})\) can be regarded as an inevitable probability. Then we pick the \(k\) triples with the highest probability to form the subgraph at time \(t + 1\) (line 10), and treat the subgraph as ground truth for inferring future. Finally we estimate all the unknown graph sequence \(\{G_{t+1}, \ldots, G_{t+\Delta t-1}\}\) in the time period \(\Delta t\), and we can obtain the probability of missing entity \(o\) at \(t + \Delta t\) by Eq.9.

B Detailed Experimental Settings

We use five TKG datasets in our experiments. Dataset statistics are described in Table 4.
Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No. 61932001.

References


[Han et al., 2021] Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In ICLR, 2021.


