Temporal-Relational Matching Network for Few-Shot Temporal Knowledge Graph Completion

Xing Gong[1], Jianyang Qin[1], Heyan Chai[1], Ye Ding[2], Yan Jia[1,3], and Qing Liao[1,3(✉)]

[1] School of Computer Science and Technology, Harbin Institute of Technology,Shenzhen, China
{gongxing,22b351005,chaiheyan}@stu.hit.edu.cn,
{jiayan2020,liaoqing}@hit.edu.cn
[2] School of Cyberspace Security, Dongguan University of Technology, Dongguan, China
dingye@dgut.edu.cn
[3] Peng Cheng Laboratory, Shenzhen, China

Abstract. Temporal knowledge graph completion (TKGC) is an important research task due to the incompleteness of temporal knowledge graphs. However, existing TKGC models face the following two issues: 1) these models cannot be directly applied to few-shot scenario where most relations have only few quadruples and new relations will be added; 2) these models cannot fully exploit the dynamic time and relation properties to generate discriminative embeddings of entities. In this paper, we propose a temporal-relational matching network, namely TR-Match, for few-shot temporal knowledge graph completion. Specifically, we design a multi-scale time-relation attention encoder to adaptively capture local and global information based on time and relation to tackle the dynamic properties problem. Then, we build a new matching processor to tackle the few-shot problem by mapping the query to few support quadruples in a relation-agnostic manner. Finally, we construct three new datasets for few-shot TKGC task based on benchmark datasets. Extensive experimental results demonstrate the superiority of our model over the state-of-the-art baselines.

Keywords: Temporal knowledge graph completion · Few-shot learning · Link prediction

1 Introduction

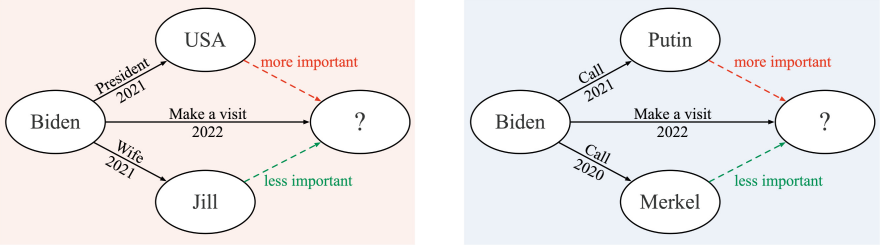
Knowledge graphs (KGs) have proved their powerful strength in various downstream tasks, such as recommender system [18], information retrieval [12], concept discovery [6], and question answering [25], etc. KGs represent every fact with a triplet (s, r, o), where s, o are the subject entity and the object entity, and r is the relation between s and o. For example, $(Biden,\ President,\ USA)$

X. Wang et al. (Eds.): DASFAA 2023, LNCS 13944, pp. 768–783, 2023.
https://doi.org/10.1007/978-3-031-30672-3_52

represents that Biden is the president of the USA. However, many facts are not static but highly ephemeral. For instance, $(Biden,\ President,\ USA)$ is true only after $(Trump,\ President,\ USA)$. Thus, by incorporating temporal information into KGs, temporal knowledge graphs (TKGs) represent each fact with a quadruple (s, r, o, t), where t represents a temporal constraint specifying the temporal validity of the fact. Due to the incompleteness of TKGs, temporal knowledge graph completion (TKGC) becomes an increasingly important research task. The task is to infer missing facts at specific timestamps based on the existing ones by answering queries such as $(Biden,\ President,\ ?,\ 2022)$.

Most current TKGC models perform the completion task by proposing a distance-based scoring function that incorporates the time representations. For example, TTransE [11] modifies the distance formula of TransE [2] by adding the temporal information, ATiSE [22] projects the representations of TKGs into the space of multi-dimensional Gaussian distributions, TeRo [21] represents time as a rotation in complex vector space, etc. However, these methods still do not perform well because they all face the few-shot relations problem and dynamic entity properties problem:

(1) **Few-shot relations.** Few-shot relations problem widely exists in real-world TKGs, bringing difficulties to complete temporal knowledge graph due to the following twofold. On the one hand, most relations in TKGs only have a small number of quadruples. This brings difficulties for conventional TKGC models to learn discriminative embeddings of entities, relations and timestamps from few quadruples, because these models usually require a lot of quadruples for training. As a result, they cannot accurately compute the distance between embeddings for completion task. On the other hand, some new relations will be added into TKGs consistently, so that conventional TKGC methods should finetune to update the learned embeddings to fit new relations. Recently, some studies, such as GMatching [19], MetaR [4], FSRL [24] and FAAN [15], have been proposed to tackle the few-shot problem, but these methods elaborated for static KGs cannot be applied to temporal KGs.



(a) Dynamic relation influence (b) Dynamic time influence

Fig. 1. Two examples to illustrate the influence of neighbors on the target entity varying dynamically with the (a) relation and (b) time when completing the knowledge graph.

(2) **Dynamic entity properties.** Dynamic entity properties mean that the influence of neighbors on entity varies with the time and relation

of different completion tasks. Figure 1 illustrates two examples of dynamic entity properties problem. Specifically, when we perform the completion task (*Biden*, *Make a visit*, ?, 2022), the influence of entity *USA* on *Biden* is greater than the influence of entity *Jill* on *Biden*, as shown in Fig. 1(a). This is because the relation *President* shares the same work property as the relation *Make a visit*, while the relation *Wife* reflects a different family property than the relation *Make a visit*. Meanwhile, as shown in Fig. 3(b), the timespan affects the weights of different neighbors too. Despite that the entities *Markel* and *Putin* share the same relation *Call*, entity *Biden* weights more on its neighbor *Putin* than *Markel* due to the smaller timespan between *Biden* and *Putin*. Existing few-shot KGC methods [4,15,19,24] ignore the dynamic properties of entities, resulting in inaccurate encoding of entities. Thus, how to jointly exploit the relation and time for TKGC remains a challenging problem.

To address the above problems, we propose a **T**emporal-**R**elational **Match**ing network for few-shot temporal knowledge graph completion (TR-Match). Specifically, we firstly follow the few-shot settings [14,17] to split and generate each task with support and query quadruples based on relation. Secondly, we propose a multi-scale time-relation attention encoder to learn the representations of support quadruples with dynamic properties. The encoder adaptively aggregates local neighbor information based on time and relation to obtain quadruple representations, and interacts the representations with global relational information among all support quadruples via a multi-head attention mechanism. Thirdly, we introduce a matching processor to deal with the few quadruples and unseen relations existing in few-shot scenario. The processor utilizes an attention-based LSTM to generate the informative representation of each query quadruple, and maps the query to few support quadruples in a relation-agnostic manner to deal with new relation by ranking the similarity between quadruples. Main contributions of this paper are summarized as follow:

- We propose a novel few-shot TKGC model, namely TR-Match, to deal with the dynamic few-shot problem.
- In TR-Match, the multi-scale time-relation attention encoder can dynamically encode quadruples based on the time and relation. Furthermore, the matching processor can map the query quadruples to support quadruples in a relation-agnostic manner to achieve few-shot TKCG task.
- We create three few-shot TKG datasets and conduct extensive experiments to demonstrate the superiority of our model over state-of-the-art baselines.

2 Related Work

2.1 Temporal Knowledge Graph Completion Methods

Recently, many temporal knowledge graph embedding models have been proposed, which encode time information in their embeddings. TTransE [11] modifies the distance formula of TransE [2] to complete the temporal knowledge graph by adding the projection of temporal information and carrying out vector

calculation. ATiSE [22] considers the temporal uncertainty during the evolution of entity/relation representations over time and projects the representations of TKGs into the space of multi-dimensional Gaussian distributions. TeRo [21] proposes scoring functions which incorporate time representations into a distance-based score function. DE-SimplE [5] uses diachronic entity embeddings to represent entities at different time steps and exploit the same score function as SimplE [8] to score the plausibility of a quadruple. Based on ComplEx [16], TComplEx [10] and TNTComplEx [10] analogously factorize the input TKG, which both models represent as a 4th-order tensor. TeLM [20] performs 4th-order tensor factorization of a TKG, using the asymmetric geometric product instead of complex Hermitian operator. These conventional TKGC methods do not consider the few-shot relations problem. Although FTMF [1] takes into account the few-shot relations problem, it focuses on temporal knowledge graph reasoning.

2.2 Few-shot Knowledge Graph Completion Methods

Recently, few-shot knowledge graph completion has attracted more and more research attention. GMatching [19] is the first one-shot knowledge graph completion model which consists an entity encoder to average ground aggregation of heterogeneous neighbors and a matching processor to measure the similarity between the support triple and the query triple. Based on GMatching, FSRL [24] uses an attention mechanism to aggregate neighbor information and a LSTM-based encoder to represent few-shot relations by support entity pairs. FAAN [15] is the first to propose a dynamic attention mechanism for one-hop neighbors adapting to the different relations which connect them. MetaR [4] focuses on transferring relation-specific meta to represent and fast update few-shot relations. MetaP [7] extracts the patterns effectively through a convolutional pattern learner and measures the validity of triples accurately by matching query patterns with reference patterns. GANA [13] puts more emphasis on neighbor information and accordingly proposes a gated and attentive neighbor aggregator. However, these models developed for static knowledge graphs cannot be applied to temporal knowledge graphs.

3 Preliminaries

In this section, we first present the notations of the temporal knowledge graph, then introduce the few-shot learning settings of our model, and finally define the few-shot temporal knowledge graph completion task in this work.

3.1 Temporal Knowledge Graph

Let $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ represent a finite set of entities, relations and timestamps, respectively. A TKG is a collection of facts represented as a set of quadruples $G = \{(s, r, o, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$ in which $s \in \mathcal{E}$ and $o \in \mathcal{E}$ are subject entity and object entity respectively, $r \in \mathcal{R}$ is the relation and $t \in \mathcal{T}$ denotes the happened time of these facts.

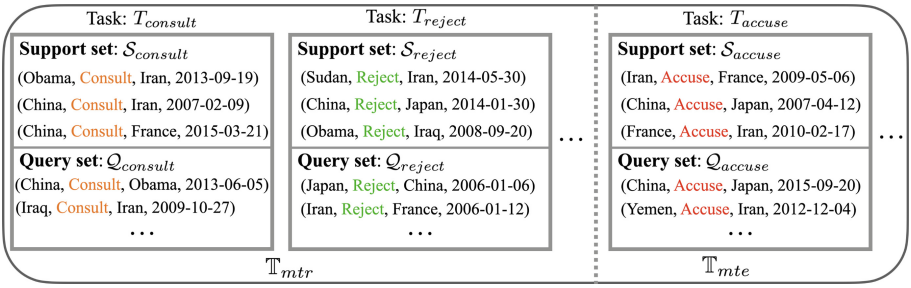


Fig. 2. A few-shot learning settings example, where few-shot size is 3.

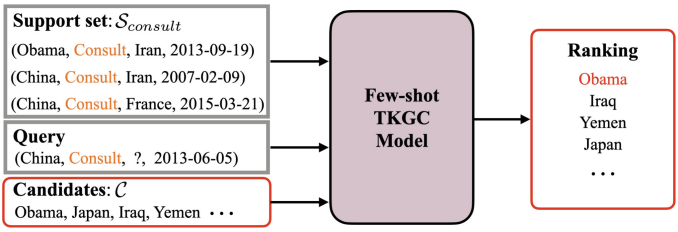


Fig. 3. A few-shot TKGC task $T_{consult}$ example.

3.2 Few-Shot Learning Settings

In this work, we classify the relations in TKG into two categories, frequent relations $\mathcal{R}_{freq}$ and sparse relations $\mathcal{R}_{sp}$, based on the frequency of their occurrence. Following GMamtching [19], the quadruples with the relations in $\mathcal{R}_{freq}$ construct background knowledge graph G' to get neighbor information. Each sparse relation corresponds to a few-shot relation. Following the standard few-shot learning settings [14,17], we consider the completion problem of quadruples with sparse relation $r \in \mathcal{R}_{sp}$ as a task, so as to access a set of tasks. In our problem, each task T_r corresponds to a sparse relation $r \in \mathcal{R}_{sp}$, and has its own support/query set: $T_r = \{\mathcal{S}_r, \mathcal{Q}_r\}$. Each support set $\mathcal{S}_r$ only contains few support quadruples $\{(s_1, r, o_1, t_1), (s_2, r, o_2, t_2), ..., (s_k, r, o_k, t_k)\}$, and $|\mathcal{S}_r| = k$ denotes the few-shot size. Besides, query set $\mathcal{Q}_r$ contains all query quadruples of relation r, including positive query quadruples $\mathcal{Q}_r^+ = \{(s_i, r, o_i^+, t_i) | (s_i, r, o_i^+, t_i) \in G, o_i^+ \in \mathcal{C}\}$ and corresponding negative query quadruples $\mathcal{Q}_r^- = \{(s_i, r, o_i^-, t_i) | (s_i, r, o_i^-, t_i) \notin G, o_i^- \in \mathcal{C}\}$. $\mathcal{C}$ is the candidate entity set, and the candidate entity set in this work is composed of all entities, i.e. $\mathcal{C} = \mathcal{E}$.

Moreover, we divide all the tasks into two sets, meta-train set $\mathbb{T}_{mtr}$ and meta-test set $\mathbb{T}_{mte}$. Notably, the relations in $\mathbb{T}_{mte}$ does not appear in $\mathbb{T}_{mtr}$. And we leave out a subset of relations in $\mathbb{T}_{mtr}$ as the meta-validation set $\mathbb{T}_{mtv}$. Figure 2 illustrates a few-shot learning settings example.

3.3 Few-Shot Temporal Knowledge Graph Completion

In this work, our purpose is to predict the object entity o given the subject entity, relation and timestamp: $(s, r, ?, t)$. In contrast to previous conventional TKGC methods that usually assume enough quadruples are available for training, this work studies the case where only few training quadruples are available. To be more specific, the goal is to rank the true object entity higher than other candidate entities in candidate entity set $\mathcal{C}$ to complete the quadruples in query set, given only few support quadruples. Figure 3 is an example of task $T_{consult}$.

Define $\ell_\Theta(\mathcal{Q}_r|\mathcal{S}_r)$ as the ranking loss of task T_r, Θ is the set of model parameters, the probabilistic optimization objective for this problem is given as:

$$\mathcal{L} = \arg\max_{\Theta} \mathbb{E}_{r\sim\mathcal{R}} \left[\ell_\Theta(\mathcal{Q}_r|\mathcal{S}_r)\right]. \tag{1}$$

4 Proposed Model

In this section, we give an introduction to our model in detail. This work aims to compute a similarity score $\mathcal{P}_\Theta((s_q, r, o_q, t_q), \mathcal{S}_r)$ for each query (s_q, r, o_q, t_q) given the support set. To achieve this purpose, we propose a completion network including an encoding and matching step, as shown in Fig. 4. Specifically, the encoding step utilizes multi-scale time-relation attention encoder to dynamically encode entity and time to obtain the representations of support and query quadruples, then the matching step uses matching processor to calculate the similarity between support and query quadruples for TKGC.

4.1 Multi-scale Time-Relation Attention Encoder

As shown in Fig. 4, multi-scale time-relation attention encoder is designed to obtain the temporal-relational representations of support and query quadruples. In this module, we design an adaptive neighbor aggregator and utilize a multi-head attention to capture the local and global information in TKGs, respectively.

Adaptive Neighbor Aggregator. The influence of neighbors on one entity keeps changing based on the relevance of relation and the length of timespan. The neighbor with similar relation and smaller timespan put a higher weight on certain entity. However, existing few-shot KGC methods [15,19,24] cannot simultaneously consider the relation and timespan to obtain discriminative entity representations.

To tackle the above issue, we design a time-relation attention mechanism to dynamically assign neighbor weights. For every entity e, our model constructs the neighbors of e, i.e., $\mathcal{N}_e = \{(e_i^{'}, r_i^{'}, t_i^{'})|, (e, r_i^{'}, e_i^{'}, t_i^{'}) \in G^{'}\}$, by searching for the quadruples in background knowledge graph $G^{'}$ whose subject entity is e. $e_i^{'}$ is the object entity which is regarded as a neighbor of e, $r_i^{'}$ is the relation between

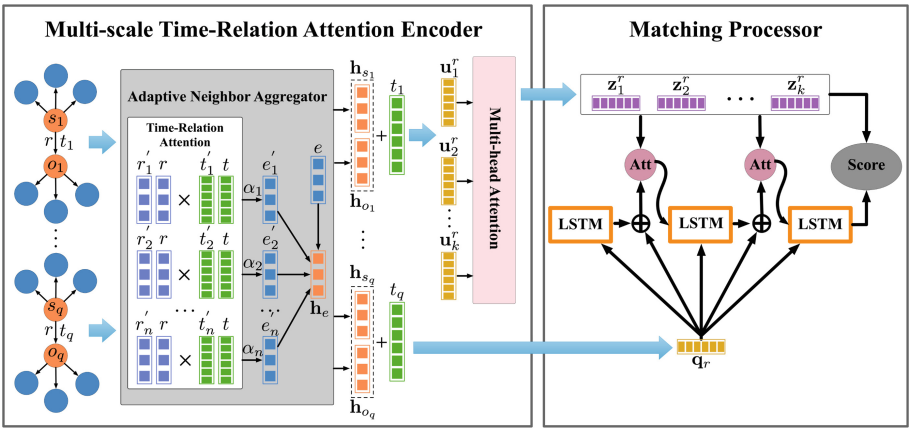


Fig. 4. The framework of TR-Match: it first obtains representations of the support and query quadruples by multi-scale time-relation attention encoder, then captures the similarity score between support and query quadruples by matching processor.

e and $e_i^{'}$, and $t_i^{'}$ is the time of fact $(e, r_i^{'}, e_i^{'})$. Then, the weight α_i of the neighbor $e_i^{'}$ on entity e can be calculated as follows:

$$\alpha_i = \frac{\exp\left(\left(\mathbf{v}_r^{\top}\mathbf{W}_1\mathbf{v}_{r_i^{'}}\right) \times \langle \Phi(t), \Phi(t_i^{'})\rangle\right)}{\sum_{(e_j^{'}, r_j^{'}, t_j^{'})\in\mathcal{N}_e} \exp\left(\left(\mathbf{v}_r^{\top}\mathbf{W}_1\mathbf{v}_{r_j^{'}}\right) \times \langle \Phi(t), \Phi(t_j^{'})\rangle\right)}. \tag{2}$$

It can be seen from Eq. (2) that we assign the weight α_i by jointly considering the relevance of different relations and the length of timespan. Specifically, the relevance of different relations can be calculated through $\mathbf{v}_r^T\mathbf{W}_1\mathbf{v}_{r_i^{'}}$. $\mathbf{v}_r$ denotes the embedding of current task relation r, which can be randomly initialized as a d-dimension vector, i.e. $\mathbf{v}_r \in \mathbb{R}^d$. $\mathbf{W}_1 \in \mathbb{R}^{d\times d}$ is a learnable parameter which is the similarity matrix to calculate the relevance between relations. The length of timespan can be measured via the inner product of paired time encoding, i.e., $\langle \Phi(t), \Phi(t_i^{'})\rangle$. To ensure that the neighbors with smaller timespan have relatively higher weights, we refer to [23] to encode the time t as follows,

$$\Phi(t) = \sqrt{\frac{1}{d}}\left[\cos(\omega_1 t), \sin(\omega_1 t), \ldots, \cos(\omega_d t), \sin(\omega_d t)\right], \tag{3}$$

where $\{\omega_1, \omega_2, ..., \omega_d\}$ is a set of learnable parameters. With this encoding function, the neighbors with smaller timespan can receive higher weights on time factor, i.e., if $|t - t_i^{'}| < |t - t_j^{'}|$, then $\langle \Phi(t), \Phi(t_i^{'})\rangle > \langle \Phi(t), \Phi(t_j^{'})\rangle$.

Having obtained the weights of neighbors, we can learn entity e's representation by adaptively aggregating neighbor information and its own information.

$$\mathbf{h}_e = \sigma(\mathbf{W_2} \sum_{(e_i', r_i', t_i') \in \mathcal{N}_e} \alpha_i \mathbf{v}_{e_i'} + \mathbf{W_3} \mathbf{v}_e), \tag{4}$$

where $\mathbf{h}_e$ denotes e's entity representation obtained from the adaptive neighbor aggregator. $\mathbf{v}_e$ is the embedding of entity e, which can be randomly initialized as a d-dimension vector, i.e. $\mathbf{v}_e \in \mathbb{R}^d$. $\sigma(\cdot)$ denotes activation function of Relu. $\mathbf{W_2}, \mathbf{W_3} \in \mathbb{R}^{d \times d}$ are trade-off parameters learned by MLP, which balance the importance between neighbor information and entity itself information.

Multi-head Attention for Support Set. The entity representations obtained above consider only local neighbor information, but ignore global relational information in support set. Due to this, we use a multi-head attention module to generate informative representations of support set by refining the relational information of support set. We use the combination of the entity pair representations and time embedding as the input $\mathbf{U}_r$ of multi-head attention to better capture the temporal dependence between support quadruples:

$$\mathbf{u}_i^r = (\mathbf{h}_{s_i} \| \mathbf{h}_{o_i}) + \Phi(t_i), \ \mathbf{U}_r = [\mathbf{u}_1^r, \mathbf{u}_2^r, ..., \mathbf{u}_k^r], \tag{5}$$

where $\mathbf{u}_i^r$ is the initial representation of support quadruple (s_i, r, o_i, t_i). $\mathbf{h}_{s_i}$ and $\mathbf{h}_{o_i}$ are the representations of entities s_i and o_i, respectively, obtained by the adaptive neighbor aggregator. $\|$ denotes concatenation operation to gather paired representations $\mathbf{h}_{s_i}$ and $\mathbf{h}_{o_i}$. Moreover, we take the time embedding $\Phi(t_i)$ as positional encoding to capture the temporal dependence of support quadruples. After that, we can obtain the support quadruple representations via the following multi-head attention layer,

$$head_i = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^\top}{\sqrt{2d}}\right) \mathbf{V}_i \tag{6}$$

$$\mathbf{Z}_r = [head_1, ..., head_N] \mathbf{W}^O \tag{7}$$

where $\mathbf{Z}_r = [\mathbf{z}_1^r, \mathbf{z}_2^r, ..., \mathbf{z}_k^r] \in \mathbb{R}^{k \times 2d}$ is the support quadruple representations. $\mathbf{W}^O \in \mathbb{R}^{2Nd \times 2d}$ is parameter matrix and N is the number of heads. $\mathbf{Q}_i = \mathbf{U}_r \mathbf{W}_i^Q$, $\mathbf{K}_i = \mathbf{U}_r \mathbf{W}_i^K$, $\mathbf{V}_i = \mathbf{U}_r \mathbf{W}_i^V$ are the 'queries', 'keys' and 'values' of the ith head attention. $\mathbf{W}_i^Q, \mathbf{W}_i^K, \mathbf{W}_i^V \in \mathbb{R}^{2d \times 2d}$ are the projection matrices of the ith head attention. The multi-head attention mechanism fully interacts with the global relational information of the support set.

4.2 Matching Processor

Conventional TKGC models are mainly encountered with two issues: (1)They require a large number of quadruples to train usually learn poor embeddings

under few-shot scenario that most relations only have a few quadruples. (2)They should be finetuned to adapt newly added relations since the learned embeddings are unavailable to update after training. To this end, we propose a matching processor that includes an embedding step and a matching step to address the poor embeddings and new relation problems, respectively.

In the embedding step, we choose LSTM to improve the quality of query embedding because LSTM can selectively capture the feature impact of support set on query [17]. Moreover, we argue that different support quadruples have different weights for a query because of the semantic divergence existing in support quadruples. Therefore, we design an **Att** module based on attention mechanism to dynamically aggregate support quadruples. Finally, we combine LSTM and **Att** module to generate informative representation h_i of query quadruple as follows:

$$h_i = \hat{h}_i + \mathbf{q}_r, \ \hat{h}_i, c_i = \mathrm{LSTM}\left(\mathbf{q}_r, p_{i-1}, c_{i-1}\right), \tag{8}$$

where $\mathrm{LSTM}(\cdot)$ is a standard LSTM cell with input $\mathbf{q}_r$, hidden state p_{i-1} and cell state c_{i-1}, and $\mathbf{q}_r = (\mathbf{h}_{s_q} \| \mathbf{h}_{o_q}) + \Phi(t_q)$ is the initial representation of query (s_q, r, o_q, t_q). The hidden state p_{i-1} regarded as a hidden representation of query (s_q, r, o_q, t_q) can be calculated via the **Att** module as follows:

$$p_{i-1} = h_{i-1} + \sum_{\mathbf{z}_j^r \in \mathbf{Z}_r} \beta_j \mathbf{z}_j^r, \tag{9}$$

$$\beta_j = \frac{\exp(h_{i-1}^\top \mathbf{z}_j^r)}{\sum_{\mathbf{z}_m^r \in \mathbf{Z}_r} \exp(h_{i-1}^\top \mathbf{z}_m^r)}, \tag{10}$$

where β_j is the weight of support quadruple (s_j, r, o_j, t_j) and $\mathbf{z}_j^r$ is the representation of support quadruple (s_j, r, o_j, t_j) obtained by Eq. (7). After l layer LSTM, we can obtain the representation h_l of query (s_q, r, o_q, t_q).

In the matching step, our goal is to rank the query quadruples to find the best candidate object entity with respect to a certain relation, so as to complete the missing quadruples. Considering that the new relation will be added to TKGs, it is not suitable to complete quadruples by calculating the distance between subject entity, relation and object entity. This is because new relation unseen in the training process cannot be presented in testing process without finetuning. To address this, we aim to directly achieve the completion task in an end-to-end network via a relation-agnostic matching. Specifically, we rank the query quadruples via the similarity score between query and support set which can be defined as the sum of inner product between quadruple representations:

$$Score(\mathbf{q}_r, \mathbf{Z}_r) = \sum_{\mathbf{z}_j^r \in \mathbf{Z}_r} h_l^\top \mathbf{z}_j^r, \tag{11}$$

where h_l and $\mathbf{z}_j^r$ are query and support quadruple representations, respectively. It is worth noting that, the representations of query quadruples learned from Eq. (8) do not embed relation directly, meaning that each query representation is relation-agnostic. Thus, we match the quadruples only based on the similarity

of entities and timespans between query and support quadruples. As a result, given a new relation, our matching process can find the best candidate object entity by selecting a query quadruple similar to support quadruples with new relation.

4.3 Loss Function and Training

We train the model on meta-training task set $\mathbb{T}_{mtr}$. We encourage high similarity scores for positive pairs and low similarity scores for negative pairs. The objective function is a hinge loss defined as follow:

$$\mathcal{L} = \sum_{r}^{\mathcal{R}} \sum_{q_r^+ \in \mathcal{Q}_r^+, q_r^- \in \mathcal{Q}_r^-} \left[\lambda + Score(\mathbf{q}_r^-, \mathbf{Z}_r) - Score(\mathbf{q}_r^+, \mathbf{Z}_r)\right]_+, \tag{12}$$

where $[x]_+ = max(0, x)$ is standard hinge loss, and λ is a hyperparameter represents safety margin distance. The detail of the training process is shown in Algorithm 1.

Algorithm 1: TR-Match Training

Input: Meta-training task set $\mathbb{T}_{mtr}$; background knowledge graph $G^{'}$; randomly initial TKG embeddings; initial model parameters Θ.

```
for epoch=0:M-1 do
    Shuffle the tasks in T_mtr;
    for T_r in T_mtr do
        Sample k quadruples as support set S_r;
        Sample a batch of positive query quadruples Q_r^+;
        Pollute the object entity in Q_r^+ to get Q_r^-;
        Obtain the representations of entities in S_r, Q_r^+ and Q_r^- by Eq. (2)-(4);
        Obtain the representations of support quadruples in S_r by Eq. (5)-(7);
        Calculate the matching score for query in Q_r^+ and Q_r^- by Eq. (8)-(11);
        Calculate the batch loss L of task T_r by Eq. (12);
        Update parameters Θ by Adam optimizer;
    end
end
return Optimal model parameters Θ
```

5 Experiments

In this section, we begin with an introduction about how to construct datasets for few-shot settings. Then, we provide an overview of baselines and implement details. Finally, we conduct a series of experiments and provide an analysis of the experimental results.

Table 1. Dataset details of ICEWS14-few, ICEWS05-15-few and ICEWS18-few. $|\mathcal{E}|$, $|\mathcal{T}|$ and $|\mathcal{R}_{sp}|$ are the number of the entities, timestamps, and sparse relations, respectively. #Tasks is the number of tasks of $\mathbb{T}_{mtr}/\mathbb{T}_{mtv}/\ \mathbb{T}_{mte}$. #Frequency is the frequency interval of sparse relations $\mathcal{R}_{sp}$.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{T}\rvert$	$\lvert\mathcal{R}_{sp}\rvert$	#Tasks	#Frequency
ICEWS14-few	7121	365	114	92/11/11	(10,200)
ICEWS05-15-few	10471	4017	99	81/9/9	(50,500)
ICEWS18-few	24572	365	99	81/9/9	(50,500)

5.1 Datasets

Since conventional temporal knowledge graph datasets, such as ICEWS14, ICEW S05-15 and ICEWS18 [3], are not suitable for few-shot settings, we construct three new datasets ICEWS14-few, ICEWS05-15-few and ICEWS18-few based on conventional datasets for few-shot TKGC. The details of each dataset are illustrated in Table 1. Following GMatching [19], we construct each dataset by selecting appropriate number of sparse relations based on the size of corresponding conventional dataset as follows:

ICEWS14-few. To construct ICEWS14-few dataset, we select the relations with number less than 200 but greater than 10 quadruples as sparse relations $\mathcal{R}_{sp}$, and the relations with number more than 200 are considered as frequent relations $\mathcal{R}_{freq}$ from ICEWS14 dataset. Then, we use 92/11/11 task relations for $\mathbb{T}_{mtr}/\mathbb{T}_{mtv}/\ \mathbb{T}_{mte}$.

ICEWS05-15-few. To construct ICEWS05-15-few dataset, we select the relations with less than 500 but more than 50 quadruples as the sparse relations $\mathcal{R}_{sp}$ from ICEWS05-15 dataset, and the relations in greater than 500 as frequent relations $\mathcal{R}_{freq}$ from ICEWS05-15 dataset. Then, we use 81/9/9 task relations for $\mathbb{T}_{mtr}/\mathbb{T}_{mtv}/\ \mathbb{T}_{mte}$.

ICEWS18-few. ICEWS18-few dataset is constructed in the same way as ICEWS05-15-few. The frequency interval of $\mathcal{R}_{sp}$ and $\mathcal{R}_{freq}$ in ICEWS18-few is the same as ICEWS05-15-few. We use 81/9/9 task relations for $\mathbb{T}_{mtr}/\mathbb{T}_{mtv}/\ \mathbb{T}_{mte}$.

5.2 Baselines

Since there is no few-shot TKGC model focusing on the completion task for comparison, we select two kinds of baseline models for comparison in this experiment: few-shot KGC models and conventional TKGC models. (1) As for few-shot KGC models, we adopt the following state-of-the-art models as baselines: GMatching [19], MetaR [4], FSRL [24] and FAAN [15]. Since all few-shot KGC models are static and cannot be generalized into dynamic scenario, we provide these models with all the quadruples in the original datasets and neglect time information, i.e., neglecting t in (s, r, o, t). (2) As for conventional TKGC models, we adopt the following state-of-the-art models for comparison: DE-SimpIE [5],

TNTComplEx [10], ATISE [22], TeRo [21] and TeLM [20]. Following GMatching, when evaluating these conventional TKGC models, we use the quadruples in background knowledge graph $G^{'}$, the quadruples in $\mathbb{T}_{mtr}$ and the quadruples in support set of $\mathbb{T}_{mtv}$ and $\mathbb{T}_{mte}$ as the train set.

5.3 Implementation

In our model, all entities and relations embeddings are initialized randomly with dimension of 100. The few-shot size k is set to 3 for the following experiments. We select the best hyperparameters that can achieve the highest MRR in validation set. The maximum number of local neighbors in adaptive neighbor aggregator is set to 50 for all datasets. In addition, we use LSTM in matching processor, the dimension of hidden state is 200 in our experiments. The number of layers of LSTM is set to 4 for ICEWS14-few and ICEWS05-15-few, and 5 for ICEWS18-few. The margin distance λ is set to 10. We implement all experiments with PyTorch and use Adam optimizer [9] to optimize model parameters with a learning rate of 0.001.

For models Gmatching, MateR, FSRL and FAAN, the few-shot size k is the same as our model, and the other parameters use the optimal parameters from the original papers. For models DE-SimpIE, TNTComplEx, ATiSE, TeRo, TeLM, we refer to the best hyperparameter settings of baseline methods reported in their original papers. We report two standard evaluation metrics: MRR and Hit@N. MRR is the mean reciprocal rank and Hits@N is the proportion of correct entities ranked in the top N, with N = 1,5,10.

5.4 Performance Comparison

Experimental Comparison with Baselines. We compare TR-Match with nine baselines on ICEWS14-few, ICEWS05-15-few and ICEWS18-few datasets, respectively, to evaluate the effectiveness of TR-Match. The performances of all models are reported in Table 2, where the best results are highlighted in bold, and the best performance of the two kinds of baselines on different datasets is underlined. It can be seen that our model outperforms all the baselines by achieving a higher MRR and Hits@1/5/10.

Compared to the few-shot KGC baselines, TR-Match consistently outperforms the best few-shot KGC baseline, i.e., FSRL, by achieving 3.0/13.4/18.1% improvement of MRR metric on ICEWS14-few/ICEWS05-15-few/ICEWS18-few datasets, respectively. This is because, compared to few-shot KGC models ignoring the temporal information, TR-Match can jointly and adaptively take the relation and time into consideration to aggregate local information, resulting in more accurate entity representations. Moreover, few-shot KGC models assume that different support quadruples are of equivalent importance to each query, while TR-Match can adaptively assign weights to support quadruples via matching process to capture the discriminative information.

Compared to the conventional TKGC baselines, TR-Match achieves significant improvements over the best results of conventional TKGC baselines by

Table 2. The overall results of all methods. The best results are highlighted in **bold**, and the best performance of the two kinds of baseline are marked as underline.

Model	ICEWS14-few				ICEWS05-15-few				ICEWS18-few			
	MRR	H@1	H@5	H@10	MRR	H@1	H@5	H@10	MRR	H@1	H@5	H@10
GMatching [19]	.213	.132	.286	.381	.222	.118	.318	.438	.184	.097	.269	.364
MetaR [4]	.224	.104	.352	.444	.215	.074	.333	.455	.132	.031	.240	.334
FSRL [24]	.306	.205	.403	.490	.246	.149	.363	.467	.199	.113	.287	.376
FAAN [15]	.241	.147	.343	.418	.200	.109	.291	.394	.155	.111	.177	.264
DE-SimpIE [5]	.265	.163	.364	.464	.208	.124	.278	.382	.192	.109	.263	.371
TNTComplEx [10]	.218	.130	.317	.402	.097	.045	.138	.199	.138	.070	.195	.283
ATiSE [22]	.259	.153	.377	.479	.179	.087	.271	.378	.097	.049	.142	.196
TeRo [21]	.236	.131	.355	.469	.187	.086	.292	.408	.165	.087	.240	.336
TeLM [20]	.270	.166	.364	.481	.198	.108	.282	.383	.203	.115	.280	.385
TR-Match(Ours)	**.315**	**.220**	**.431**	**.529**	**.279**	**.176**	**.385**	**.497**	**.235**	**.150**	**.324**	**.408**

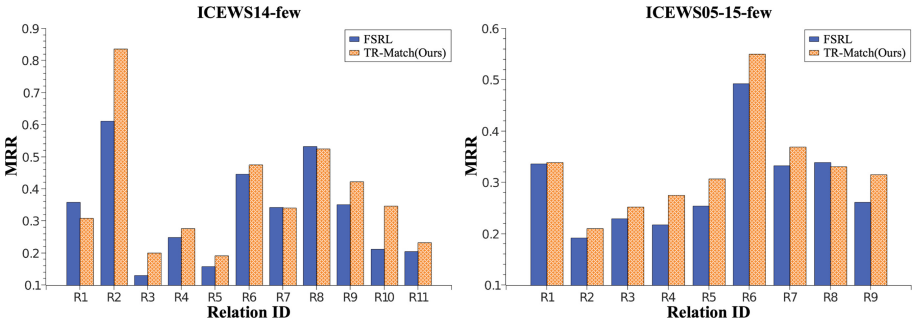


Fig. 5. The MMR of TR-Math and FSRL for each relation on ICEWS14-few and ICEWS05-15-few.

obtaining 16.7/34.13/15.8% gains of MRR metric on ICEWS14-few/ICEWS05-15-few/ICEWS18-few datasets, respectively. This is because conventional TKGC models requiring large-scale training data fail to achieve satisfying performance on such three datasets with few-shot relations. By contrast, our model is powerful to handle few-shot data by calculating the similarity between quadruples in matching processor.

Comparison over Different Relations. To demonstrate the superiority of our model in more detail, we set up comparative experiments on ICEWS14-few and ICEWS05-15-few with different relations. In this experiment, we compare our proposed TR-Match with the best few-shot static KGC baseline FSRL. The experimental results are shown in Fig. 5, where Relation ID represents a class of relation. On ICEWS14-few, our TR-Match outperforms FSRL in MRR metric with 8 out of 11 relations. On ICEWS05-15-few, our TR-Match outperforms FSRL in MRR metric with 8 out of 9 relations. Experimental results indicate that

Table 3. The results of ablation experiment.

Model	ICEWS14-few				ICEWS05-15-few				ICEWS18-few			
	MRR	H@1	H@5	H@10	MRR	H@1	H@5	H@10	MRR	H@1	H@5	H@10
TR-Match(v1)	.291	.197	.394	.497	.245	.135	.369	.483	.208	.129	.286	.362
TR-Match(v2)	.197	.104	.315	.373	.241	.145	.339	.434	.166	.102	.225	.304
TR-Match(v3)	.297	.203	.401	.488	.261	.152	.379	.486	.193	.114	.273	.363
TR-Match(Ours)	**.315**	**.220**	**.431**	**.529**	**.279**	**.176**	**.385**	**.497**	**.235**	**.150**	**.324**	**.408**

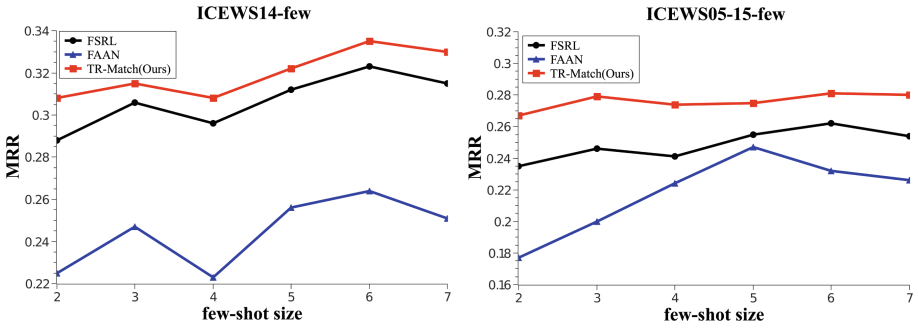


Fig. 6. Impact of few-shot size on ICEWS14-few and ICEWS05-15-few.

TR-Match is more powerful to learn discriminative quadruple representations by taking the advantage of time encoding in both encoding step and matching step.

5.5 Ablation Study

We perform experiments on all the datasets with several variants of our proposed model to provide a better understanding of the contribution of each module to our proposed model. The ablative results are shown in Table 3. In TR-Match(v1), we use the neighbor encoder proposed by GMatching [19] instead of our proposed adaptive neighbor aggregator to encode entities. Experiments demonstrate that dynamically aggregating neighbors based on relation and time can improve the model performance compared to aggregating neighbors with fixed weights. In TR-Match(v2), we remove multi-head attention from our model. The experimental results demonstrate that multi-head attention can stably boost the performance of our model by capturing the global relational information for support set. In TR-Match(v3), we replace **Att** in matching processor with mean-pooling. Experimental results show that **Att** can improve our model performance by adaptively aggregating support features compared to fixed support weights.

5.6 Impact of Few-shot Size

In this subsection, we study the impact of few-shot size k. We perform experiments on TR-Match, FSRL [24], and FAAN [15] models on ICEWS14-few and ICEWS05-15-few datasets, and set different k values from a subset

$\{2, 3, 4, 5, 6, 7\}$. Experimental results in Fig. 6 demonstrate that: (1) The performance of TR-Match is always better than the comparative models, indicating the capability of our proposed method in few-shot TKGC. (2) TR-Match obtains relatively stable boosts compared to FSRL and FAAN, which shows the robustness of TR-Match to few-shot size.

6 Conclusion

In this paper, we propose a new few-shot temporal knowledge graph completion model, i.e., TR-Match, which consists of an encoding step and a matching step. In the encoding step, we can dynamically aggregate the local and global information to generate temporal-relational representations, so as to capture the dynamic properties in completion task. In the matching step, we can map the query to few support quadruples in a relation-agnostic manner to overcome the few-shot problem. Additionally, we construct three datasets suitable for few-shot learning based on public datasets. The experimental results show the superiority of our model and the effectiveness of each component in our model.

Acknowledgements. This work was partially supported by the National Natural Science Foundation of China: 61976051, U19A2067, and the Major Key Project of PCL: PCL2022A03

References

1. Bai, L., Zhang, M., Zhang, H., Zhang, H.: FTMF: Few-shot temporal knowledge graph completion based on meta-optimization and fault-tolerant mechanism. In: World Wide Web, pp. 1–28 (2022)
2. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NeurIPS, pp. 2787–2795 (2013)
3. Boschee, E., Lautenschlager, J., O'Brien, S., Shellman, S., Starz, J., Ward, M.: ICEWS Coded Event Data (2015). https://doi.org/10.7910/DVN/28075
4. Chen, M., Zhang, W., Zhang, W., Chen, Q., Chen, H.: Meta relational learning for few-shot link prediction in knowledge graphs. In: EMNLP-IJCNLP, pp. 4217–4226 (2019)
5. Goel, R., Kazemi, S.M., Brubaker, M.A., Poupart, P.: Diachronic embedding for temporal knowledge graph completion. In: AAAI, pp. 3988–3995 (2020)
6. Jeon, I., Papalexakis, E.E., Faloutsos, C., Sael, L., Kang, U.: Mining billion-scale tensors: Algorithms and discoveries. VLDB J. **25**(4), 519–544 (2016). https://doi.org/10.1007/s00778-016-0427-4
7. Jiang, Z., Gao, J., Lv, X.: Metap: Meta pattern learning for one-shot knowledge graph completion. In: SIGIR, pp. 2232–2236 (2021)
8. Kazemi, S.M., Poole, D.: Simple embedding for link prediction in knowledge graphs. NeurIPS, pp. 4289–4300 (2018)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR (2015)
10. Lacroix, T., Obozinski, G., Usunier, N.: Tensor decompositions for temporal knowledge base completion. In: ICLR (2019)

11. Leblay, J., Chekol, M.W.: Deriving validity time in knowledge graph. WWW, pp. 1771–1776 (2018)
12. Liu, Z., Xiong, C., Sun, M., Liu, Z.: Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval. In: ACL, pp. 2395–2405 (2018)
13. Niu, G., et al.: Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion. In: SIGIR, pp. 213–222 (2021)
14. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: ICLR (2017)
15. Sheng, J., et al.: Adaptive attentional network for few-shot knowledge graph completion. In: EMNLP, pp. 1681–1691 (2020)
16. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML, pp. 2071–2080 (2016)
17. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. NeurIPS, pp. 3637–3645 (2016)
18. Wang, X., Wang, D., Xu, C., He, X., Cao, Y., Chua, T.S.: Explainable reasoning over knowledge graphs for recommendation. In: AAAI, pp. 5329–5336 (2019)
19. Xiong, W., Yu, M., Chang, S., Guo, X., Wang, W.Y.: One-shot relational learning for knowledge graphs. In: EMNLP, pp. 1980–1990 (2018)
20. Xu, C., Chen, Y.Y., Nayyeri, M., Lehmann, J.: Temporal knowledge graph completion using a linear temporal regularizer and multivector embeddings. In: NAACL, pp. 2569–2578 (2021)
21. Xu, C., Nayyeri, M., Alkhoury, F., Yazdi, H.S., Lehmann, J.: Tero: A time-aware knowledge graph embedding via temporal rotation. In: COLING, pp. 1583–1593 (2020)
22. Xu, C., Nayyeri, M., Alkhoury, F., Yazdi, H., Lehmann, J.: Temporal knowledge graph completion based on time series gaussian embedding. In: ISWC, pp. 654–671 (2020)
23. Xu, D., Ruan, C., Körpeoglu, E., Kumar, S., Achan, K.: Inductive representation learning on temporal graphs. ArXiv (2020)
24. Zhang, C., Yao, H., Huang, C., Jiang, M., Li, Z.J., Chawla, N.: Few-shot knowledge graph completion. In: AAAI, pp. 3041–3048 (2020)
25. Zhang, Y., Dai, H., Kozareva, Z., Smola, A.J., Song, L.: Variational reasoning for question answering with knowledge graph. In: AAAI, pp. 1–8 (2018)