Recurrent Event Network for Reasoning over Temporal Knowledge Graphs

Woojeong Jin[†], Changlin Zhang[†], Pedro Szekely[‡], Xiang Ren[†‡]
[†]Department of Computer Science, University of Southern California
[‡]Information Sciences Institute, University of Southern California
{woojeong.jin, changlin.zhang, xiangren}@usc.edu, pszekely@isi.edu

Abstract

Recently, there has been a surge of interest in learning representation of graph-structured data that are dynamically evolving. However, current dynamic graph learning methods lack a principled way in modeling *temporal*, *multi-relational*, and *concurrent* interactions between nodes—a limitation that is especially problematic for the task of temporal knowledge graph reasoning, where the goal is to predict unseen entity relationships (i.e., events) over time. Here we present Recurrent Event Network (RE-NET)—an architecture for modeling complex event sequences—which consists of a *recurrent event encoder* and a *neighborhood aggregator*. The event encoder employs a RNN to capture (subject, relation)-specific patterns from historical entity interactions; while the neighborhood aggregator summarizes concurrent interactions within each time stamp. An output layer is designed for predicting forthcoming, multi-relational events. Experiments[1] on temporal link prediction over two knowledge graph datasets demonstrate the effectiveness of our method, especially on multi-step inference over time.

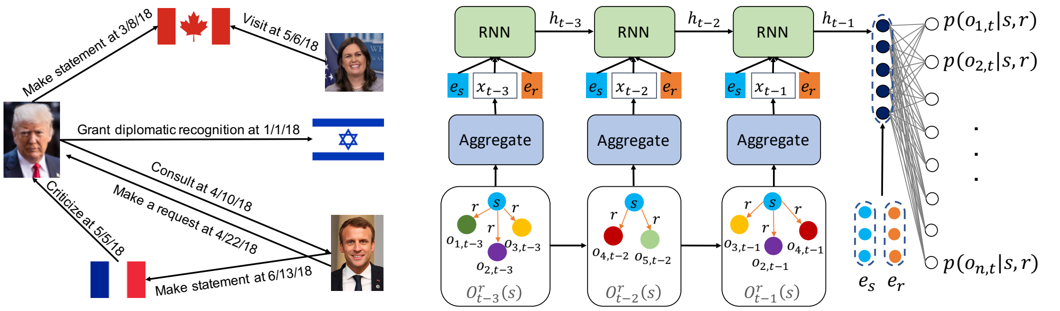
1 Introduction

Representation learning on graph-structured data that are dynamically evolving has emerged as an important machine learning task in a wide range of applications, such as social network analysis, question answering, and event forecasting. This task becomes particularly challenging when dealing with multi-relational graphs with complex interaction patterns between nodes—e.g., in reasoning over temporal knowledge graphs (TKGs). However, despite that there has been some recent studies on representation learning and reasoning over TKGs (Trivedi et al., 2017; García-Durán et al., 2018; Dasgupta et al., 2018; Leblay & Chekol, 2018), these methods either simply embed the associated time information into low-dimensional space while ignoring the *temporal dependencies between events* (García-Durán et al., 2018; Dasgupta et al., 2018; Leblay & Chekol, 2018), or lack of a principled way to consolidate *concurrent events* within the same time stamps (Trivedi et al., 2017).

In this paper, we propose a general neural architecture, called Recurrent Event Network (RE-NET), for modeling multi-relational event sequences. To address the above limitations, RE-NET introduces an *event sequence encoder* and a *neighborhood aggregation module*. The event sequence encoder captures temporal and multi-relation dynamics by utilizing the past interactions between entities (i.e., events). This encoder harness a recurrent neural network to encode the past entity interactions. The neighborhood aggregation module resolves multiple concurrent interactions at the same time stamp by consolidating neighborhood information via different ways. A classifier layer is designed to predict unseen entity relationships for the current time stamp, given prior encoder state, subject entity, and relation. We adopt multi-class cross entropy loss to learn the RE-NET model, and perform multi-step inference for predicting forthcoming events on the graph over time.

We evaluate our proposed method on temporal graph reasoning (i.e., link prediction) using two public temporal knowledge graph datasets, and test the performance of multi-step inference over time. Experiment results demonstrate the strengths of RE-NET on modeling temporal, multi-relational graph data with concurrent events, as compared to the state-of-the-art static and temporal graph reasoning methods. We further show that RE-NET can perform effective multi-step inference to predict unseen entity relationships (i.e., forthcoming events) in a distant future.

[1] Code and data are released at https://github.com/INK-USC/RENet.



(a) An example of temporal KG

(b) Overview of the RE-NET architecture

Figure 1: Illustration of (a) temporal knowledge graph and (b) the Recurrent Event Network (RE-NET) architecture. RE-NET employs a RNN to capture (s, r)-specific interactions with object entities $O^r_t(s)$ (modeled by a neighborhood aggregator) at different time t, and adopts the hidden state of RNN h_t and (subject, relation) embeddings (e_s, e_r) for predicting forthcoming entity interactions.

2 Proposed Method

2.1 Temporal Knowledge Graph Reasoning

A *temporal knowledge graph* (TKG) is a multi-relational, directed graph with time-stamped edges (relationships) between the nodes (entities). An *event* is defined as a time-stamped edge (subject entity, relation, object entity, time) in a TKG, and is denoted by the quadruple (s, r, o, t). A TKG is built upon a set of event quadruples $\{(s_i, r_i, o_i, t_i)\}_i$[2], where each time-stamped edge has a direction pointing from the subject entity to the object entity. The task of **reasoning over TKGs** (or temporal link prediction) aims to predict unseen relationships with object entities given $(s, r, ?, t)$, or to predict relationships with subject entities given $(?, r, o, t)$, based on the observed events in the TKG.

2.2 Recurrent Event Network

Predicting unseen entity relationships requires the ability of learning temporal dependency patterns across historical events. We propose RE-NET to capture the temporal dynamics for predicting forthcoming events and to summarize the concurrent events within the same time stamps. Our architecture consists of a Recurrent Neural Network (RNN) as an event sequence encoder and a neighborhood aggregation module to collect entities each time. Here we only describe our object prediction model and subject prediction can be obtained by reversing subjects and objects.

Event Sequence Encoder. We first define a conditional probability of an object o_t at time t given a subject s and a relation r, and history of objects interacted with subject s under relation r.

$$p(o_t|s, r, \{O^r_{t-k-1}(s), ..., O^r_{t-1}(s)\}) = f(e_s, e_r, h_{t-1}(s, r)), \tag{1}$$

where $O^r_{t-1}(s)$ is a set of objects interacted with s under r at $t-1$, $e_s, e_r \in \mathbb{R}^d$ are representation of subject s and relation r, and $h_{t-1}(s, r)$ is a history vector which includes information from the past m object set sequence $\{O^r_{t-m-1}(s), ..., O^r_{t-1}(s)\}$. In our implementation, f is a one-layer fully-connected network with softmax activation function to output class (entity) probability.

We assume that the next set of objects can be predicted with a previous object history under the same relation. To track the history of interactions, we introduce an event sequence encoder based on RNN as follows

$$h_t(s, r) = \text{RNN}(e_s, e_r, g(O^r_t(s)), h_{t-1}(s, r)). \tag{2}$$

In each time step, besides the history $h_{t-1}(s, r)$, we add the aggregation of neighbour representation $g(O^r_t(s))$. We also use a subject embedding and a relation embedding as well as aggregation of objects as the input of RNN to make the RNN subject-relation specific.

Neighborhood Aggregation. A subject entity can make interactions with multiple objects under relation r at the same time stamp. To encode the entity neighborhood information to a fixed-length input for our RNN encoder, we define an aggregation module $g(\{e_o : o \in O^r_t(s)\})$ to collect information from relation-specific neighbors.

[2] The same triple (s, r, o) may occur multiple times in different time stamps, yielding different event quadruples.

Algorithm 1: Training RE-NET

```
Input: Events $E = \{(s_i, r_i, o_i, t_i)\}_i$
Output: A trained classifier $f : (s, r; \{O^r_{t-k-1}(s), ..., O^r_{t-1}(s)\}) \to p(o_{i,t}|s, r)$
while Parameters in RE-NET does not change do
    foreach $(s_i, r_i, o_i, t_i)$ in $E$ do
        Get history $\{O^r_{t-k-1}(s_i), ..., O^r_{t-1}(s_i)\}$
        set $h(s_i, r_i) \leftarrow 0$
        for $i = t - k - 1$ to $t - 1$ do
            Aggregate neighborhood embeddings $x_i \leftarrow g(O^r_i(s))$
            Update history vector $h(s_i, r_i) \leftarrow \text{RNN}(e_{s_i}, e_{r_i}, x_i, h(s_i, r_i))$
        Compute $\hat{y} = f(e_{s_i}, e_{r_i}, h(s_i, r_i))$
        $\hat{y} \leftarrow softmax(\hat{y})$
        Compute cross entropy loss based on equation 3.
    Update model parameters.
```

2.3 Aggregator Architectures

Here we discuss different choices for the aggregate function $g(\cdot)$, which capture different kinds of neighborhood information for each subject entity and relation, i.e., (s, r).

Mean Aggregator. The baseline method is to simply take the element-wise mean of the vectors in $\{e_o : o \in O^r_t(s)\}$. But the mean aggregator treats all neighboring objects equally, and thus ignores the different importance of each neighbour entity.

Attentive Aggregator. We define an attentive aggregator based on the additive attention introduced in (Bahdanau et al., 2015). The aggregator function is defined as $g(O^r_t(s)) = \sum_{o \in O^r_t(s)} \alpha_o e_o$ where $\alpha_o = \text{softmax}(v^\top \tanh(W(e_s; e_r; e_o)))$. $v \in \mathbb{R}^d$ and $W \in \mathbb{R}^{d \times 3d}$ are trainable weight matrices. By adding attention function of the subject and the relation, the weight can determine how relevant each object entity is to the subject and relation.

Graph Convolutional Aggregator. Based on the graph convolutional operation in (Kipf & Welling, 2016), we designed an aggregator with GCN message passing mechanism, which takes the form as $g(O^r_t(s)) = \sigma(W \cdot \frac{1}{|O^r_t(s)|} \sum_{o \in O^r_t(s)} e_o)$, where $W \in \mathbb{R}^d$ is a trainable weight matrix.

2.4 Inference and Learning of RE-Net

Multi-step Inference over Time. At inference time, given the subject entity s and relation r, RE-NET performs multi-step inference to predict forthcoming entities. For example, reasoning for c time steps from last time stamp t yields entity prediction $\{O^r_{t+1}(s), O^r_{t+2}(s), \ldots, O^r_{t+c}(s)\}$. During multi-step inference, the encoder state is updated based on current predictions, and will be used for making next predictions. That is, for each time step we rank the candidate entities and select top-m entities as current predictions. We maintain the history as a sliding window of length k, so the oldest interaction set will be detached and new predicted entity set will be added to the history.

Model Learning via Entity Prediction. The (object) entity prediction can be viewed as a multi-class classification task, where each class corresponds to one object entity. To learn weights and representations for entities and relations, we adopt a multi-class cross entropy loss to the model's output.The loss function for the predicted o_t is defined as:

$$\mathcal{L} = - \sum_{(s,r,o,t) \in E} \sum_{c=1}^{M} y_c \log(p(o = c|s, r)), \tag{3}$$

where E is set of events, and y_c is a binary indicator (0 or 1) if class label c is the correct classification for prediction o. $p(o = c|s, r)$ is the probability that o is in class c. We use the softmax function on equation 1 to get the probability.

Algorithm 1 describes the training for RE-NET.

Table 1: Performance comparison on link prediction (average metrics over the entire test set) on the two public datasets. RE-Net with mean aggregator outperforms all other baseline methods.

	Method	**ICEWS18 - filtered**				**GDELT - filtered**			
		MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
Static	TransE	17.56	2.48	26.95	43.87	16.05	0.00	26.10	42.29
	DisMult	22.16	12.13	26.00	42.18	18.71	11.59	20.05	32.55
	ComplEx	30.09	21.88	34.15	45.96	22.77	15.77	24.05	36.33
	R-GCN	23.19	16.36	25.34	36.48	23.31	17.24	24.94	34.36
	ConvE	37.67	29.91	40.80	51.69	36.99	28.05	40.32	51.44
Temporal	Know-Evolve*	3.27	3.23	3.23	3.26	2.43	2.33	2.35	2.41
	HyTE	7.31	3.10	7.50	14.95	6.37	0.00	6.72	18.63
	TTransE	8.36	1.94	8.71	21.93	5.52	0.47	5.01	15.27
	TA-TransE	12.85	0.00	19.04	37.53	16.62	0.00	27.65	42.53
	TA-DistMult	28.53	20.30	31.57	44.96	29.35	22.11	31.56	41.39
	RE-Net (Mean)	**42.38**	**35.80**	**44.99**	**54.90**	**39.15**	**30.84**	**43.07**	**53.48**
	RE-Net (Attn)	41.46	34.67	44.19	54.44	38.07	29.44	42.26	52.93
	RE-Net (GC)	41.35	34.54	44.05	54.35	37.99	30.05	41.40	52.18

Table 2: Performance comparison on ICEWS and GDELT dataset with raw metrics. We observe our method outperforms all other methods.

	Method	**ICEWS18 - raw**				**GDELT - raw**			
		MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
Static	TransE	12.37	1.51	15.99	34.65	7.84	0.00	8.92	23.30
	DisMult	13.86	5.61	15.22	31.26	8.61	3.91	8.27	17.04
	ComplEx	15.45	8.04	17.19	30.73	9.84	5.17	9.58	18.23
	R-GCN	15.05	8.13	16.49	29.00	12.17	7.40	12.37	20.63
	ConvE	22.81	13.63	25.83	41.43	18.37	11.29	19.36	32.13
Temporal	Know-Evolve*	0.11	0.00	0.00	0.47	0.11	0.00	0.02	0.10
	HyTE	7.41	3.10	7.33	16.01	6.69	0.01	7.57	19.06
	TTransE	8.44	1.85	8.95	22.38	5.53	0.46	4.97	15.37
	TA-TransE	8.02	0.00	9.53	24.44	8.84	0.00	11.69	25.32
	TA-DistMult	15.62	7.63	17.09	32.21	10.34	4.44	10.44	21.63
	RE-Net (Mean)	**26.07**	**16.55**	**29.70**	**44.77**	**19.02**	**11.74**	**20.20**	**33.34**
	RE-Net (Attn)	25.77	16.34	29.42	44.47	18.60	11.39	19.68	32.96
	RE-Net (GC)	25.78	16.35	29.35	44.44	18.53	11.41	19.63	32.53

3 Experiments

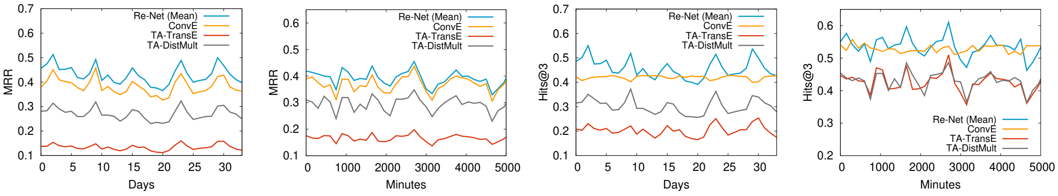
We evaluate the proposed method with other static and temporal baselines on the task of link prediction. Our goal is to predict future entities given the past interactions. Furthermore, we examine the method in a multi-step prediction setting.

Datasets. We use two datasets, Integrated Crisis Early Warning System (ICEWS18) (Boschee et al., 2015) and Global Database of Events, Language, and Tone (GDELT) (Leetaru & Schrodt, 2013). ICEWS is collected from 1/1/2018 to 10/31/2018,and GDELT is from 1/1/2018 to 1/31/2018.

Experimental Setup. We use Gated Recurrent Unit (Cho et al., 2014) as our event sequence encoder, where the length of history is set as $k = 10$. We use a 1-layer fully connected layer for f in equation 1 At inference time, RE-Net performs multi-step prediction across the time stamps in dev and test sets. For each dataset, we split it into three subsets, i.e., train(80%)/valid(10%)/test(10%), by time stamps. We report Mean Reciprocal Ranks (MRR) and Hits@$1/3/10$, using the filtered version of the datasets as described in (Bordes et al., 2013).

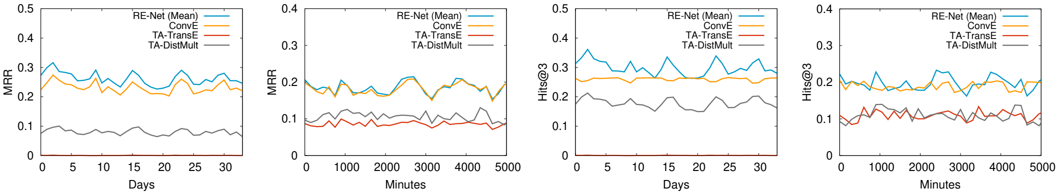
Baseline Methods. We compare our approach to baselines for static graphs and temporal graphs: **(1)** ***Static Methods.*** By ignoring the edge time stamps, we construct a static, cumulative graph for all the training events, and apply multi-relational graph representation learning methods including TransE (Bordes et al., 2013), DisMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), R-GCN (Schlichtkrull et al., 2018), and ConvE (Dettmers et al., 2018). **(2)** ***Temporal Reasoning Methods.*** We also compare with state-of-the-art temporal reasoning for knowledge graphs, including Know-Evolve[3] (Trivedi et al., 2017), TA-TransE/DistMult (García-Durán et al., 2018),

[3] *: We found a problematic formulation in Know-Evolve when dealing with concurrent events (Eq. (3) in its paper) and a flaw in its evaluation code. The performance dramatically drops after fixing the code. Details are discussed in Section D of supplementary materials.



(a) ICEWS18 (MRR) (b) GDELT (MRR) (c) ICEWS18 (H@3) (d) GDELT (H@3)

Figure 2: Performance comparison of link prediction over different time stamps (i.e., temporal reasoning) on the test sets of ICEWS18 and GDELT datasets with **filtered** metrics.



(a) ICEWS18 (MRR) (b) GDELT (MRR) (c) ICEWS18 (H@3) (d) GDELT (H@3)

Figure 3: Performance comparison of link prediction over different time stamps (i.e., temporal reasoning) on the test sets of ICEWS18 and GDELT datasets with **raw** metrics.

HyTE (Dasgupta et al., 2018), and TTransE (Leblay & Chekol, 2018). Experiment settings and implementation details of baseline methods are described in Section C of supplementary materials.

3.1 Results

Overall Performance Comparison. Tables 1 and 2 summarize performance comparison results on averaged metrics over the entire test sets. RE-Net (Mean), RE-Net (Attn) and RE-Net (GC) denote our method with mean, attentive, and graph convolutional aggregators, respectively. Overall, our proposed method, RE-Net, outperforms all other baselines on both datasets. Among all variants of aggregation functions, the mean aggregator shows the best performances while the attentive and graph convolutional aggregator little lower performances. In particular, the temporal baselines do not always show the better performances than static methods. This is because their methods try to model the temporal information regardless of the past interactions. For example, TA-TransE designed time-aware representations for relation embeddings which only see the current time stamp, and thus it cannot generalize to unseen time stamps. However, our proposed method capture the temporal dependencies between entities and can predict future entities even in the unseen time stamps.

Performance over Time. Figs. 2 and 3 show the performance comparisons over different time stamps for the two datasets with filtered metrics and raw metrics, respectively. RE-Net outperforms other baselines with the MRR metric on both datasets. However, RE-Net and ConvE compete each other with the Hits@3 metric. We notice that RE-Net is getting lower and the difference between RE-Net and ConvE is getting smaller as shown in Fig. 3b. This is because RE-Net predicts entities based on the history, but the history is updated with predicted entities at inference, which is why RE-Net's performance is getting lower. We also note that performance gaps on the GDELT dataset is smaller than on the ICEWS18 dataset. GDELT has finer granularity of time than ICEWS18, and thus it requires longer history for prediction.

4 Conclusion and Future Work

In this work, we study the task of temporal reasoning over dynamic knowledge graphs, and propose Recurrent Event Network (RE-Net) to model temporal, multi-relational, and concurrent interactions between entities. We show the effectiveness of RE-Net on predicting unseen relationships over time on two TKG datasets. Interesting future work includes aggregating multi-hop neighborhood information for event modeling, and in-depth study of different aggregator functions.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, 2013.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. Icews coded event data. *Harvard Dataverse*, 12, 2015.

Kyunghyun Cho, Bart van Merrienboer, aglar Gülehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*, 2014.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *EMNLP*, 2018.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*, 2018.

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. Learning sequence encoders for temporal knowledge graph completion. In *EMNLP*, 2018.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

Julien Leblay and Melisachew Wudage Chekol. Deriving validity time in knowledge graph. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pp. 1771–1776. International World Wide Web Conferences Steering Committee, 2018.

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

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, 2018.

Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In *ICML*, 2017.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, 2016.

Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *CoRR*, abs/1412.6575, 2015.