Know-Evolve: Deep Temporal Reasoning for Dynamic Knowledge Graphs

Rakshit Trivedi [1] **Hanjun Dai** [1] **Yichen Wang** [1] **Le Song** [1]

Abstract

The availability of large scale event data with time stamps has given rise to *dynamically evolving* knowledge graphs that contain temporal information for each edge. Reasoning over time in such dynamic knowledge graphs is not yet well understood. To this end, we present **Know-Evolve**, a novel deep evolutionary knowledge network that learns non-linearly evolving entity representations over time. The occurrence of a fact (edge) is modeled as a multivariate point process whose intensity function is modulated by the score for that fact computed based on the learned entity embeddings. We demonstrate significantly improved performance over various relational learning approaches on two large scale real-world datasets. Further, our method effectively predicts occurrence or recurrence time of a fact which is novel compared to prior reasoning approaches in multi-relational setting.

1. Introduction

Reasoning is a key concept in artificial intelligence. A host of applications such as search engines, question-answering systems, conversational dialogue systems, and social networks require reasoning over underlying structured knowledge. Effective representation and learning over such knowledge has come to the fore as a very important task. In particular, Knowledge Graphs have gained much attention as an important model for studying complex multi-relational settings. Traditionally, knowledge graphs are considered to be static snapshot of multi-relational data. However, recent availability of large amount of event based interaction data that exhibits complex temporal dynamics in addition to its multi-relational nature has created the need for approaches that can characterize and reason over temporally evolving systems. For instance, GDELT (Leetaru & Schrodt, 2013) and ICEWS (Boschee et al., 2017) are two popular event based data repository that contains evolving knowledge about entity interactions across the globe.

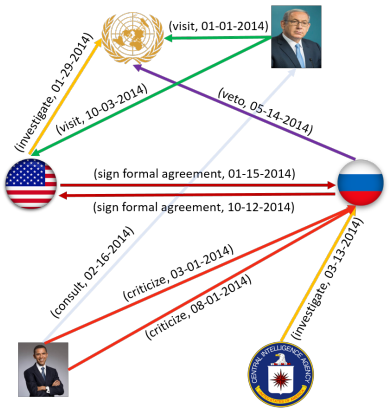


Figure 1. Sample temporal knowledge subgraph between persons, organizations and countries.

Thus traditional knowledge graphs need to be augmented into *Temporal Knowledge Graphs*, where facts occur, recur or evolve over time in these graphs, and each edge in the graphs have temporal information associated with it. Figure 1 shows a subgraph snapshot of such temporal knowledge graph. Static knowledge graphs suffer from incompleteness resulting in their limited reasoning ability. Most work on static graphs have therefore focussed on advancing entity-relationship representation learning to infer missing facts based on available knowledge. But these methods lack ability to use rich temporal dynamics available in underlying data represented by temporal knowledge graphs.

Effectively capturing temporal dependencies across facts in addition to the relational (structural) dependencies can help improve the understanding on behavior of entities and how they contribute to generation of facts over time. For example, one can precisely answer questions like:

- **Object prediction**. (Who) will Donald Trump mention next?
- **Subject prediction**. (Which country) will provide material support to US next month?
- **Time prediction**. (When) will Bob visit Burger King?

[1]College of Computing, Georgia Institute of Technology. Correspondence to: Rakshit Trivedi <rstrivedi@gatech.edu>, Le Song <lsong@cc.gatech.edu>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

"People (entities) change over time and so do relationships." When two entities forge a relationship, the newly formed edge drives their preferences and behavior. This change is effected by combination of their own historical factors (**temporal evolution**) and their compatibility with the historical factors of the other entity (**mutual evolution**).

For instance, if two countries have tense relationships, they are more likely to engage in conflicts. On the other hand, two countries forging an alliance are most likely to take confrontational stands against enemies of each other. Finally, time plays a vital role in this process. A country that was once peaceful may not have same characteristics 10 years in future due to various facts (events) that may occur during that period. Being able to capture this temporal and evolutionary effects can help us reason better about future relationship of an entity. We term this combined phenomenon of evolving entities and their dynamically changing relationships over time as **"knowledge evolution"**.

In this paper, we propose an elegant framework to model knowledge evolution and reason over complex non-linear interactions between entities in a multi-relational setting. The key idea of our work is to model the occurrence of a fact as multidimensional temporal point process whose conditional intensity function is modulated by the relationship score for that fact. The relationship score further depends on the dynamically evolving entity embeddings. Specifically, our work makes the following contributions:

- We propose a novel deep learning architecture that evolves over time based on availability of new facts. The dynamically evolving network will ingest the incoming new facts, learn from them and update the embeddings of involved entities based on their recent relationships and temporal behavior.
- Besides predicting the occurrence of a fact, our architecture has ability to predict time when the fact may potentially occur which is not possible by any prior relational learning approaches to the best of our knowledge.
- Our model supports *Open World Assumption* as missing links are not considered to be false and may potentially occur in future. It further supports prediction over unseen entities due to its novel dynamic embedding process.
- The large-scale experiments on two real world datasets show that our framework has consistently and significantly better performance for link prediction than state-of-arts that do not account for temporal and evolving non-linear dynamics.
- Our work aims to introduce the use of powerful mathematical tool of temporal point process framework for temporal reasoning over dynamically evolving knowledge graphs. It has potential to open a new research direction in reasoning over time for various multi-relational settings with underlying spatio-temporal dynamics.

2. Preliminaries

2.1. Temporal Point Process

A temporal point process (Cox & Lewis, 2006) is a random process whose realization consists of a list of events localized in time, $\{t_i\}$ with $t_i \in \mathbb{R}^+$. Equivalently, a given temporal point process can be represented as a counting process, $N(t)$, which records the number of events before time t.

An important way to characterize temporal point processes is via the conditional intensity function $\lambda(t)$, a stochastic model for the time of the next event given all the previous events. Formally, $\lambda(t)\mathrm{d}t$ is the conditional probability of observing an event in a small window $[t, t+\mathrm{d}t)$ given the history $\mathcal{T}(t) := \{t_k | t_k < t\}$ up to t, *i.e.*,

$$\begin{aligned}\lambda(t)\mathrm{d}t &:= \mathbb{P}\left\{\text{event in } [t, t+\mathrm{d}t) | \mathcal{T}(t)\right\} \\ &= \mathbb{E}[\mathrm{d}N(t) | \mathcal{T}(t)]\end{aligned} \tag{1}$$

where one typically assumes that only one event can happen in a small window of size $\mathrm{d}t$, *i.e.*, $\mathrm{d}N(t) \in \{0, 1\}$.

From the survival analysis theory (Aalen et al., 2008), given the history $\mathcal{T} = \{t_1, \ldots, t_n\}$, for any $t > t_n$, we characterize the conditional probability that no event happens during $[t_n, t)$ as $S(t|\mathcal{T}) = \exp\big(-\int_{t_n}^{t} \lambda(\tau)\,\mathrm{d}\tau\big)$. Moreover, the conditional density that an event occurs at time t is defined as :

$$f(t) = \lambda(t)\, S(t) \tag{2}$$

The functional form of the intensity $\lambda(t)$ is often designed to capture the phenomena of interests. Some Common forms include: Poisson Process, Hawkes processes (Hawkes, 1971), Self-Correcting Process (Isham & Westcott, 1979), Power Law and Rayleigh Process.

Rayleigh Process is a non-monotonic process and is well-adapted to modeling fads, where event likelihood drops rapidly after rising to a peak. Its intensity function is $\lambda(t) = \alpha \cdot (t)$, where $\alpha > 0$ is the weight parameter, and the log survival function is $\log S(t|\alpha) = -\alpha \cdot (t)^2/2$.

2.2. Temporal Knowledge Graph representation

We define a *Temporal Knowledge Graph (TKG)* as a multi-relational directed graph with timestamped edges between any pair of nodes. In a *TKG*, each edge between two nodes represent an event in the real world and edge type (relationship) represent the corresponding event type. Further an edge may be available multiple times (recurrence). We do not allow duplicate edges and self-loops in graph. Hence, all recurrent edges will have different time points and every edge will have distinct subject and object entities.

Given n_e entities and n_r relationships, we extend traditional triplet representation for knowledge graphs to introduce time dimension and represent each fact in *TKG* as a quadruplet (e^s, r, e^o, t), where $e^s, e^o \in \{1, \ldots, n_e\}$, $e^s \neq e^o$,

$r \in \{1, \ldots, n_r\}$, $t \in \mathbb{R}^+$. It represents the creation of relationship edge r between subject entity e^s, and object entity e^o at time t. The complete TKG can therefore be represented as an $n_e \times n_e \times n_r \times \mathcal{T}$ - dimensional tensor where $\mathcal{T}$ is the total number of available time points. Consider a TKG comprising of N edges and denote the globally ordered set of corresponding N observed events as $\mathcal{D} = \{(e^s, r, e^o, t)_n\}_{n=1}^N$, where $0 \leq t_1 \leq t_2 \ldots \leq T$.

3. Evolutionary Knowledge Network

We present our unified knowledge evolution framework (Know-Evolve) for reasoning over temporal knowledge graphs. The reasoning power of Know-Evolve stems from the following three major components:

1. A powerful mathematical tool of temporal point process that models occurrence of a fact.
2. A bilinear relationship score that captures multi-relational interactions between entities and modulates the intensity function of above point process.
3. A novel deep recurrent network that learns non-linearly and mutually evolving latent representations of entities based on their interactions with other entities in multi-relational space over time.

3.1. Temporal Process

Large scale temporal knowledge graphs exhibit highly heterogeneous temporal patterns of events between entities. Discrete epoch based methods to model such temporal behavior fail to capture the underlying intricate temporal dependencies. We therefore model time as a random variable and use temporal point process to model occurrence of fact.

More concretely, given a set of observed events $\mathcal{O}$ corresponding to a *TKG*, we construct a relationship-modulated multidimensional point process to model occurrence of these events. We characterize this point process with the following conditional intensity function:

$$\lambda_r^{e^s,e^o}(t|\bar{t}) = f(g_r^{e^s,e^o}(\bar{t})) * (t - \bar{t}) \tag{3}$$

where $t > \bar{t}$, t is the time of the current event and $\bar{t} = max(t^{e^s}-, t^{e^o}-)$ is the most recent time point when either subject or object entity was involved in an event before time t. Thus, $\lambda_r^{e^s,e^o}(t|\bar{t})$ represents intensity of event involving triplet (e^s, r, e^j) at time t given previous time point $\bar{t}$ when either e^s or e^o was involved in an event. This modulates the intensity of current event based on most recent activity on either entities' timeline and allows to capture scenarios like non-periodic events and previously unseen events. $f(\cdot) = \exp(\cdot)$ ensures that intensity is positive and well defined.

3.2. Relational Score Function

The first term in (3) modulates the intensity function by the relational compatibility score between the involved entities in that specific relationship. Specifically, for an event $(e^s, r, e^o, t) \in \mathcal{D}$ occurring at time t, the score term $g_r^{e^s,e^o}$ is computed using a bilinear formulation as follows:

$$g_r^{e^s,e^o}(t) = \mathbf{v}^{\mathbf{e^s}}(t-)^T \cdot \mathbf{R_r} \cdot \mathbf{v}^{\mathbf{e^o}}(t-) \tag{4}$$

where $\mathbf{v}^{\mathbf{e^s}}, \mathbf{v}^{\mathbf{e^s}} \in \mathbb{R}^d$ represent latent feature embeddings of entities appearing in subject and object position respectively. $\mathbf{R_r} \in \mathbb{R}^{d \times d}$ represents relationship weight matrix which attempts to capture interaction between two entities in the specific relationship space r. This matrix is unique for each relation in dataset and is learned during training. t is time of current event and $t-$ represent time point just before time t. $\mathbf{v}^{\mathbf{e^s}}(t-)$ and $\mathbf{v}^{\mathbf{e^o}}(t-)$, therefore represent most recently updated vector embeddings of subject and object entities respectively before time t. As these entity embeddings evolve and update over time, $g_r^{e^s,e^o}(t)$ is able to capture cumulative knowledge learned about the entities over the history of events that have affected their embeddings.

3.3. Dynamically Evolving Entity Representations

We represent latent feature embedding of an entity e at time t with a low-dimensional vector $\mathbf{v}^{\mathbf{e}}(t)$. We add superscript $\mathbf{s}$ and $\mathbf{o}$ as shown in Eq. (4) to indicate if the embedding corresponds to entity in subject or object position respectively. We also use relationship-specific low-dimensional representation for each relation type.

The latent representations of entities change over time as entities forge relationships with each other. We design novel deep recurrent neural network based update functions to capture mutually evolving and nonlinear dynamics of entities in their vector space representations. We consider an event $m = (e^s, r, e^o, t)_m \in \mathcal{D}$ occurring at time t. Also, consider that event m is entity e^s's p-th event while it is entity e^o's q-th event. As entities participate in events in a heterogeneous pattern, it is less likely that $p = q$ although not impossible. Having observed this event, we update the embeddings of two involved entities as follows:

Subject Embedding:

$$\begin{aligned}\mathbf{v}^{\mathbf{e^s}}(t_p) &= \sigma(\mathbf{W_t^s}(t_p - t_{p-1}) + \mathbf{W^{hh}} \cdot \mathbf{h}^{\mathbf{e^s}}(t_p-)) \\ \mathbf{h}^{\mathbf{e^s}}(t_p-) &= \sigma(\mathbf{W^h} \cdot [\mathbf{v}^{\mathbf{e^s}}(t_{p-1}) \oplus \mathbf{v}^{\mathbf{e^o}}(t_p-) \oplus \mathbf{r_{p-1}^{e^s}}])\end{aligned} \tag{5}$$

Object Embedding:

$$\begin{aligned}\mathbf{v}^{\mathbf{e^o}}(t_q) &= \sigma(\mathbf{W_t^o}(t_q - t_{q-1}) + \mathbf{W^{hh}} \cdot \mathbf{h}^{\mathbf{e^o}}(t_q-)) \\ \mathbf{h}^{\mathbf{e^o}}(t_q-) &= \sigma(\mathbf{W^h} \cdot [\mathbf{v}^{\mathbf{e^o}}(t_{q-1}) \oplus \mathbf{v}^{\mathbf{e^s}}(t_q-) \oplus \mathbf{r_{q-1}^{e^o}}])\end{aligned} \tag{6}$$

where, $\mathbf{v}^{\mathbf{e^s}}, \mathbf{v}^{\mathbf{e^o}} \in \mathbb{R}^d$. $t_p = t_q = t_m$ is the time of observed event. For subject embedding update in Eq. (5), t_{p-1} is the time point of the previous event in which entity e^s was

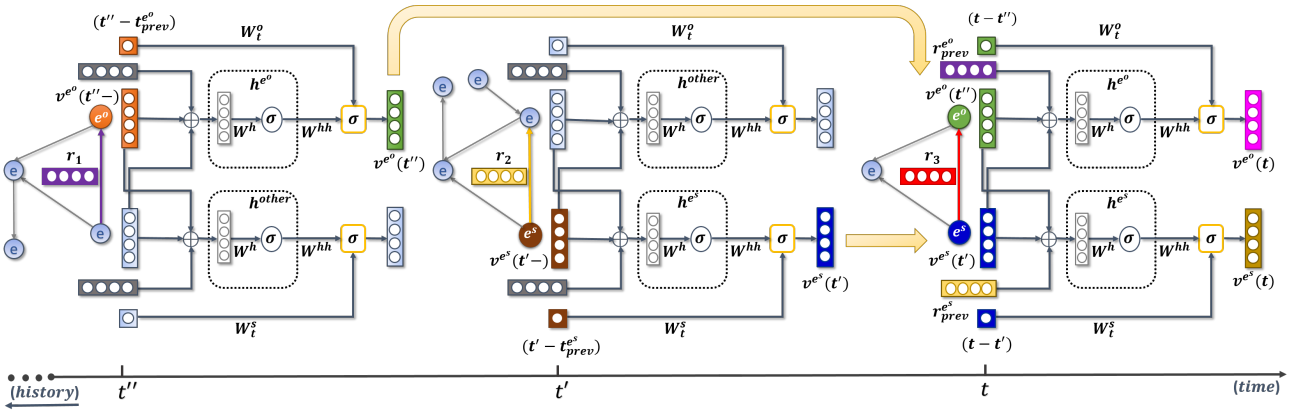
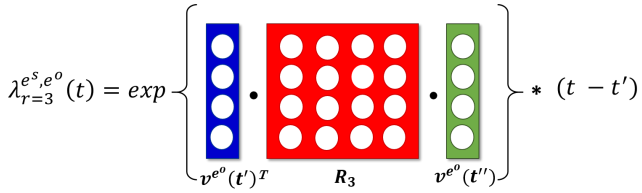
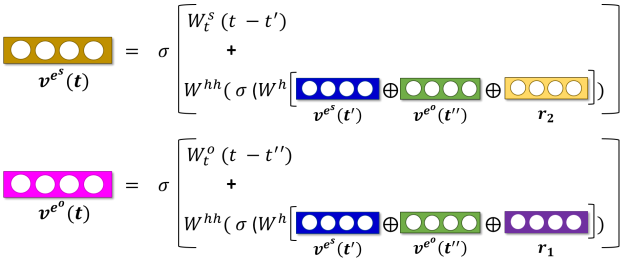


Figure 2. Realization of Evolutionary Knowledge Network Architecture over a timeline. Here t'', t' and t may or may not be consecutive time points. We focus on the event at time point t and show how previous events affected the embeddings of entities involved in this event. From Eq. (5) and (6), $t_{p-1} = t'$ and $t_{q-1} = t''$ respectively. $t^{e^s}_{prev}$, $t^{e^o}_{prev}$ represent previous time points in history before t', t''. $\mathbf{h^{other}}$ stands for hidden layer for the entities (other than the ones in focus) involved in events at t' and t''. $r^{e^s}_{prev} = r_2$ and $r^{e^o}_{prev} = r_1$. All other notations mean exactly as defined in text. We only label nodes, edges and embeddings directly relevant to event at time t for clarity.



(a) Intensity Computation at time t



(c) Entity Embedding update after event observed at time t

Figure 3. One step visualization of Know-Evolve computations done in Figure 2 after observing an event at time t. (Best viewed in color)

involved. t_p- is the timepoint just before time t_p. Hence, $\mathbf{v^{e^s}}(t_{p-1})$ represents latest embedding for entity e^s that was updated after $(p-1)$-th event for that entity. $\mathbf{v^{e^o}}(t_p-)$ represents latest embedding for entity e^o that was updated any time just before $t_p = t_m$. This accounts for the fact that entity e^o may have been involved in some other event during the interval between current (p) and previous $(p-1)$ event of entity e^s. $\mathbf{r^{e^s}_{p-1}} \in \mathbb{R}^c$ represent relationship embedding that corresponds to relationship type of the $(p-1)$-th event of entity e^s. Note that the relationship vectors are static and do not evolve over time. $\mathbf{h^{e^s}}(t_p-) \in \mathbb{R}^d$ is the hidden layer. The semantics of notations apply similarly to object embedding update in Eq. (6).

$\mathbf{W^s_t}, \mathbf{W^o_t} \in \mathbb{R}^{d\times 1}$, $\mathbf{W^{hh}} \in \mathbb{R}^{d\times l}$ and $\mathbf{W^h} \in \mathbb{R}^{l\times(2d+c)}$ are weight parameters in network learned during training. $\mathbf{W^s_t}, \mathbf{W^o_t}$ captures variation in temporal drift for subject and object respectively. $\mathbf{W^{hh}}$ is shared parameter that captures recurrent participation effect for each entity. $\mathbf{W^h}$ is a shared projection matrix applied to consider the compatibility of entities in their previous relationships. $\oplus$ represent simple concatenation operator. $\sigma(\cdot)$ denotes nonlinear activation function ($tanh$ in our case). Our formulations use simple RNN units but it can be replaced with more expressive units like LSTM or GRU in straightforward manner. In our experiments, we choose $d = l$ and $d \neq c$ but they can be chosen differently. Below we explain the rationales of our deep recurrent architecture that captures nonlinear evolutionary dynamics of entities over time.

Reasoning Based on Structural Dependency: The hidden layer ($\mathbf{h^{e^s}}$) reasons for an event by capturing the compatibility of most recent subject embedding with most recent object embedding in previous relationship of subject entity. This accounts for the behavior that within a short period of time, entities tend to form relationships with other entities that have similar recent actions and goals. This layer thereby uses historical information of the two nodes involved in current event and the edges they both created before this event. This holds symmetrically for hidden layer ($\mathbf{h^{e^o}}$).

Reasoning based on Temporal Dependency: The recurrent layer uses hidden layer information to model the intertwined evolution of entity embeddings over time. Specifically this layer has two main components:

- **Drift over time:** The first term captures the temporal difference between consecutive events on respective dimension of each entity. This captures the external influences

that entities may have experienced between events and allows to smoothly drift their features over time. This term will not contribute anything in case when multiple events happen for an entity at same time point (e.g. within a day in our dataset). While $t_p - t_{p-1}$ may exhibit high variation, the corresponding weight parameter will capture these variations and along with the second recurrent term, it will prevent $\mathbf{v}^{\mathbf{e^s}}(t_p)$ to collapse.

- **Relation-specific Mutual Evolution:** The latent features of both subject and object entities influence each other. In multi-relational setting, this is further affected by the relationship they form. Recurrent update to entity embedding with the information from the hidden layer allows to capture the intricate non-linear and evolutionary dynamics of an entity with respect to itself and the other entity in a specific relationship space.

3.4. Understanding Unified View of Know-Evolve

Figure (2) and Figure (3) shows the architecture of knowledge evolution framework and one step of our model.

The updates to the entity representations in Eq. (5) and (6) are driven by the events involving those entities which makes the embeddings piecewise constant i.e. an entity embedding remains unchanged in the duration between two events involving that entity and updates only when an event happens on its dimension. This is justifiable as an entity's features may update only when it forges a relationship with other entity within the graph. Note that the first term in Eq. (5) and (6) already accounts for any external influences.

Having observed an event at time t, Know-Evolve considers it as an incoming fact that brings new knowledge about the entities involved in that event. It computes the intensity of that event in Eq. (3) which is based on relational compatibility score in Eq. (4) between most recent latent embeddings of involved entities. As these embeddings are piecewise constant, we use time interval term $(t - \bar{t})$ in Eq. (3) to make the overall intensity piecewise linear which is standard mathematical choice for efficient computation in point process framework. This formulation naturally leads to Rayleigh distribution which models time interval between current event and most recent event on either entities' dimension. Rayleigh distribution has an added benefit of having a simple analytic form of likelihood which can be further used to find entity for which the likelihood reaches maximum value and thereby make precise entity predictions.

4. Efficient Training Procedure

The complete parameter space for the above model is: $\mathbf{\Omega} = \{\{\mathbf{V^e}\}_{e=1:n_e}, \{\mathbf{R_r}\}_{r=1:n_r}, \mathbf{W_e}, \mathbf{W_t^s}, \mathbf{W_t^o}, \mathbf{W^h}, \mathbf{W^{hh}}, \mathbf{W_r}\}$. Although Know-Evolve gains expressive power from deep architecture, Table 4 (Appendix D) shows that the memory footprint of our model is comparable to simpler relational models. The intensity function in (3) allows to use maximum likelihood estimation over all the facts as our objective function. Concretely, given a collection of facts recorded in a temporal window $[0, T)$, we learn the model by minimizing the joint negative log likelihood of intensity function (Daley & Vere-Jones, 2007) written as:

$$\mathcal{L} = \underbrace{-\sum_{p=1}^{N} \log\left(\lambda_r^{e^s,e^o}(t_p|\bar{t_p})\right)}_{\text{happened events}} + \underbrace{\sum_{r=1}^{n_r}\sum_{e^s=1}^{n_e}\sum_{e^o=1}^{n_e}\int_0^T \lambda_r^{e^s,e^o}(\tau|\bar{\tau})\,d\tau}_{\text{survival term}} \tag{7}$$

The first term maximizes the probability of specific type of event between two entities; the second term penalizes non-presence of all possible types of events between all possible entity pairs in a given observation window. We use Back Propagation Through Time (BPTT) algorithm to train our model. Previous techniques (Du et al., 2016; Hidasi et al., 2016) that use BPTT algorithm decompose data into independent sequences and train on mini-batches of those sequences. But there exists intricate relational and temporal dependencies between data points in our setting which limits our ability to efficiently train by decomposing events into independent sequences. To address this challenge, we design an efficient Global BPTT algorithm (Algorithm 2, Appendix A) that creates mini-batches of events over global timeline in sliding window fashion and allows to capture dependencies across batches while retaining efficiency.

Intractable Survival Term. To compute the second survival term in (7), since our intensity function is modulated by relation-specific parameter, for each relationship we need to compute survival probability over all pairs of entities. Next, given a relation r and entity pair (e^s, e^o), we denote $P_{(e^s,e^o)}$ as total number of events of type r involving either e^s or e^o in window $[T_0, T)$. As our intensity function is piecewise-linear, we can decompose the integration term $-\int_{T_0}^{T} \lambda_r^{e^s,e^o}(\tau|\bar{\tau})d\tau$ into multiple time intervals where intensity is constant:

$$\begin{aligned}
&\int_{T_0}^{T} \lambda_r^{e^s,e^o}(\tau|\bar{\tau})d\tau \\
&= \sum_{p=1}^{P_{(e^s,e^o)}-1} \int_{t_p}^{t_{p+1}} \lambda_r^{e^s,e^o}(\tau|\bar{\tau})d\tau \\
&= \sum_{p=1}^{P_{(e^s,e^o)}-1} (t_{p+1}^2 - t_p^2)\cdot exp(\mathbf{v^{e^s}}(t_p)^T \cdot \mathbf{R_r} \cdot \mathbf{v^{e^o}}(t_p))
\end{aligned} \tag{8}$$

The integral calculations in (8) for all possible triplets requires $\mathcal{O}(n^2 r)$ computations (n is number of entities and r is the number of relations). This is computationally intractable

Algorithm 1 Survival Loss Computation in mini-batch

Input: Minibatch $\mathcal{E}$, size s, Batch Entity List bl
$loss = 0.0$
for $p = 0$ **to** $s - 1$ **do**
  subj_feat = $\mathcal{E}_p \rightarrow \mathbf{v}^{\mathbf{e^s}}(t-)$
  obj_feat = $\mathcal{E}_p \rightarrow \mathbf{v}^{\mathbf{e^o}}(t-)$
  rel_weight = $\mathcal{E}_p \rightarrow \mathbf{R_r}$
  t_end = $\mathcal{E}_p \rightarrow t$
  $subj\_surv = 0, obj\_surv = 0, total\_surv = 0$
  for $i = 0$ **to** $bl.size$ **do**
    obj_other = $bl[i]$
    if obj_other == $\mathcal{E}_p \rightarrow e^s$ **then**
      continue
    end if
    $\bar{t} = max(t^{e^s}-, t^{e^o}-)$
    $subj\_surv \mathrel{+}= (t\_end^2 - \bar{t}^2) \cdot exp(subj\_feat^T \cdot rel\_weight \cdot obj\_other\_feat)$
  end for
  for $j = 0$ **to** $bl.size$ **do**
    subj_other = $bl[i]$
    if subj_other == $\mathcal{E}_p \rightarrow e^o$ **then**
      continue
    end if
    $\bar{t} = max(t^{e^s}-, t^{e^o}-)$
    $obj\_surv \mathrel{+}= (t\_end^2 - \bar{t}^2) \cdot exp(subj\_other\_feat^T \cdot rel\_weight \cdot obj\_feat)$
  end for
  $loss \mathrel{+}= subj\_surv + obj\_surv$
end for

and also unnecessary. Knowledge tensors are inherently sparse and hence it is plausible to approximate the survival loss in a stochastic setting. We take inspiration from techniques like noise contrastive (Gutmann & Hyvärinen, 2012) estimation and adopt a random sampling strategy to compute survival loss: Given a mini-batch of events, for each relation in the mini-batch, we compute dyadic survival term across all entities in that batch. Algorithm 1 presents the survival loss computation procedure. While this procedure may randomly avoid penalizing some dimensions in a relationship, it still includes all dimensions that had events on them. The computational complexity for this procedure will be $\mathcal{O}(2n'r'm)$ where m is size of mini-batch and n' and r' represent number of entities and relations in the mini-batch.

5. Experiments

5.1. Temporal Knowledge Graph Data

We use two datasets: Global Database of Events, Language, and Tone (GDELT) (Leetaru & Schrodt, 2013) and Integrated Crisis Early Warning System (ICEWS) (Boschee et al., 2017) which has recently gained attention in learning community (Schein et al., 2016) as useful temporal KGs. GDELT data is collected from April 1, 2015 to Mar 31, 2016 (temporal granularity of 15 mins). ICEWS dataset is collected from Jan 1, 2014 to Dec 31, 2014 (temporal granularity of 24 hrs). Both datasets contain records of events that include two actors, action type and timestamp of event. We use different hierarchy of actions in two datasets - (top level 20 relations for GDELT while last level 260 relations for ICEWS) - to test on variety of knowledge tensor configurations. Note that this does not filter any record from the dataset. We process both datasets to remove any duplicate quadruples, any mono-actor events (*i.e.*, we use only dyadic events), and self-loops. We report our main results on full versions of each dataset. We create smaller version of both datasets for exploration purposes. Table 1 (Appendix B) provide statistics about the data and Table 2 (Appendix B) demonstrates the sparsity of knowledge tensor.

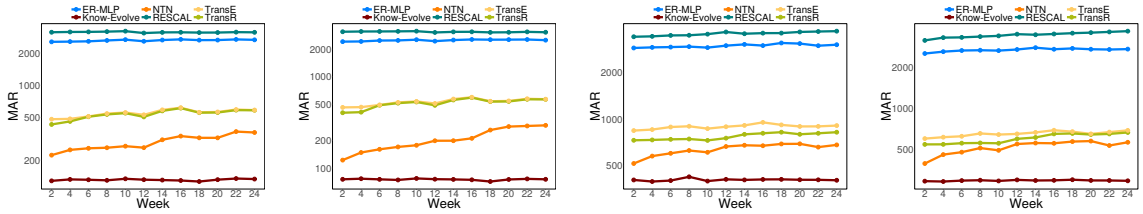
5.2. Competitors

We compare the performance of our method with following relational learning methods: RESCAL, Neural Tensor Network (NTN), Multiway Neural Network (ER-MLP), TransE and TransR. To the best of our knowledge, there are no existing relational learning approaches that can predict time for a new fact. Hence we devised two baseline methods for evaluating time prediction performance — *(i) Multi-dimensional Hawkes process (MHP):* We model dyadic entity interactions as multi-dimensional Hawkes process similar to (Du et al., 2015). Here, an entity pair constitutes a dimension and for each pair we collect sequence of events on its dimension and train and test on that sequence. Relationship is not modeled in this setup. *(ii) Recurrent Temporal Point Process (RTPP):* We implement a simplified version of RMTPP (Du et al., 2016) where we do not predict the marker. For training, we concatenate static entity and relationship embeddings and augment the resulting vector with temporal feature. This augmented unit is used as input to global RNN which produces output vector $\mathbf{h_t}$. During test time, for a given triplet, we use this vector $\mathbf{h_t}$ to compute conditional intensity of the event given history which is further used to predict next event time. Appendix C provides implementation details of our method and competitors.

5.3. Evaluation Protocol

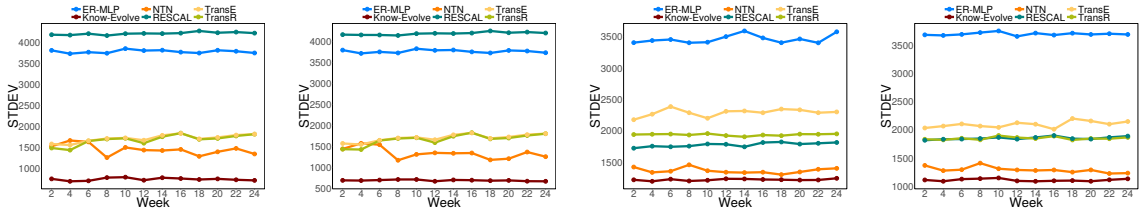
We report experimental results on two tasks: *Link prediction* and *Time prediction*.

Link prediction: Given a test quadruplet (e^s, r, e^o, t), we replace e^o with all the entities in the dataset and compute the conditional density $d_r^{e^s,e^o} = \lambda_r^{e^s,e^o}(t) S_r^{e^s,e^o}(t)$ for the resulting quadruplets including the ground truth. We then sort all the quadruplets in the descending order of this density to rank the correct entity for object position. We also conduct testing after applying the filtering techniques described in (Bordes et al., 2013) - we only rank against the entities that do not generate a true triplet (seen in train) when it replaces



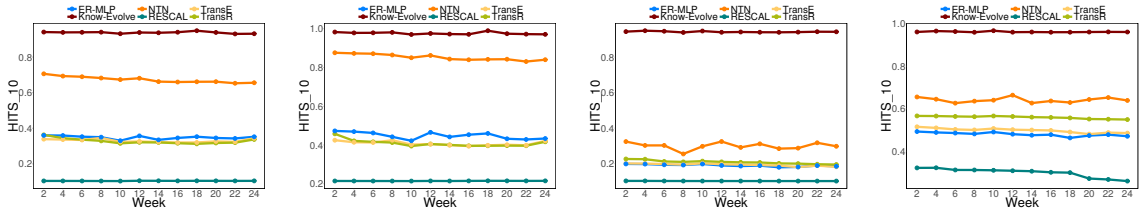
(a) ICEWS-raw (b) ICEWS-filtered (c) GDELT-raw (d) GDELT-filtered

Figure 4. Mean Average Rank (MAR) for Entity Prediction on both datasets.



(a) ICEWS-raw (b) ICEWS-filtered (c) GDELT-raw (d) GDELT-filtered

Figure 5. Standard Deviation (STD) in MAR for Entity Prediction on both datasets.



(a) ICEWS-raw (b) ICEWS-filtered (c) GDELT-raw (d) GDELT-filtered

Figure 6. HITS@10 for Entity Prediction on both datasets.

ground truth object. We report Mean Absolute Rank (MAR), Standard Deviation for MAR and HITS@10 (correct entity in top 10 predictions) for both Raw and Filtered Versions.

Time prediction: Give a test triplet (e^s, r, e^o), we predict the expected value of next time the fact (e^s, r, e^o) can occur. This expectation is defined by: $\mathbb{E}_r^{e^s,e^o}(t) = \sqrt{\frac{\pi}{2\exp\left(g_r^{e^s,e^o}(t)\right)}}$, where $g_r^{e^s,e^o}(t)$ is computed using equation (4). We report Mean Absolute Error (MAE) between the predicted time and true time in hours.

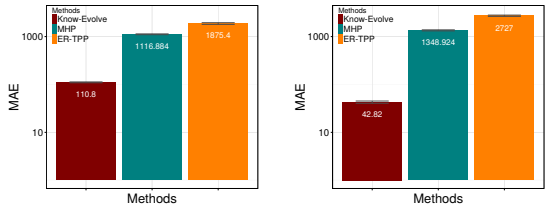
Sliding Window Evaluation. As our work concentrates on temporal knowledge graphs, it is more interesting to see the performance of methods over time span of test set as compared to single rank value. This evaluation method can help to realize the effect of modeling temporal and evolutionary knowledge. We therefore partition our test set in 12 different slides and report results in each window. For both datasets, each slide included 2 weeks of time.

5.4. Quantitative Analysis

Link Prediction Results. Figure (4, 5, 6) demonstrate link prediction performance comparison on both datasets. Know-Evolve significantly and consistently outperforms all competitors in terms of prediction rank without any deterioration over time. Neural Tensor Network's second best performance compared to other baselines demonstrate its rich expressive power but it fails to capture the evolving dynamics of intricate dependencies over time. This is further substantiated by its decreasing performance as we move test window further in time.

The second row represents deviation error for MAR across samples in a given test window. Our method achieves significantly low deviation error compared to competitors making it most stable. Finally, high performance on HITS@10 metric demonstrates extensive discriminative ability of Know-Evolve. For instance, GDELT has only 20 relations but 32M events where many entities interact with each other in multiple relationships. In this complex setting, other methods depend only on static entity embeddings to perform prediction unlike our method which does effectively infers new knowledge using powerful evolutionary network and provides accurate prediction results.

Time Prediction Results. Figure 7 demonstrates that Know-Evolve performs significantly better than other point process based methods for predicting time. MHP uses a specific parametric form of the intensity function which limits its expressiveness. Further, each entity pair interaction is modeled as an independent dimension and does not take



(a) GDELT-500 (b) ICEWS-500

Figure 7. Time prediction performance (Unit is hours).

into account relational feature which fails to capture the intricate influence of different entities on each other. On the other hand, RTPP uses relational features as part of input, but it sees all events globally and cannot model the intricate evolutionary dependencies on past events. We observe that our method effectively captures such non-linear relational and temporal dynamics.

In addition to the superior quantitative performance, we demonstrate the effectiveness of our method by providing extensive exploratory analysis in Appendix E.

6. Related Work

In this section, we discuss relevant works in relational learning and temporal modeling techniques.

6.1. Relational Learning

Among various relational learning techniques, neural embedding models that focus on learning low-dimensional representations of entities and relations have shown state-of-the-art performance. These methods compute a score for the fact based on different operations on these latent representations. Such models can be mainly categorized into two variants:

Compositional Models. RESCAL (Nickel et al., 2011) uses a relation specific weight matrix to explain triplets via pairwise interactions of latent features. Neural Tensor Network (NTN) (Socher et al., 2013) is more expressive model as it combines a standard NN layer with a bilinear tensor layer. (Dong et al., 2014) employs a concatenation-projection method to project entities and relations to lower dimensional space. Other sophisticated models include Holographic Embeddings (HoLE) (Nickel et al., 2016b) that employs circular correlation on entity embeddings and Neural Association Models (NAM) (Liu et al., 2016), a deep network used for probabilistic reasoning.

Translation Based Models. (Bordes et al., 2011) uses two relation-specific matrices to project subject and object entities and computes L_1 distance to score a fact between two entity vectors. (Bordes et al., 2013) proposed TransE model that computes score as a distance between relation-specific translations of entity embeddings. (Wang et al., 2014) improved TransE by allowing entities to have distributed representations on relation specific hyperplane where distance between them is computed. TransR (Lin et al., 2015) extends this model to use separate semantic spaces for entities and relations and does translation in the relationship space.

(Nickel et al., 2016a) and (Yang et al., 2015; Toutanova & Chen, 2015) contains comprehensive reviews and empirical comparison of relational learning techniques respectively. All these methods consider knowledge graphs as static models and lack ability to capture temporally evolving dynamics.

6.2. Temporal Modeling

Temporal point processes have been shown as very effective tool to model various intricate temporal behaviors in networks (Yang & Zha, 2013; Farajtabar et al., 2014; 2015; Du et al., 2015; 2016; Wang et al., 2016a;b;c; 2017a;b). Recently, (Wang et al., 2016a; Dai et al., 2016b) proposed novel co-evolutionary feature embedding process that captures self-evolution and co-evolution dynamics of users and items interacting in a recommendation system. In relational setting, (Loglisci et al., 2015) proposed relational mining approach to discover changes in structure of dynamic network over time. (Loglisci & Malerba, 2017) proposes method to capture temporal autocorrelation in data to improve predictive performance. (Sharan & Neville, 2008) proposes summarization techniques to model evolving relational-temporal domains. Recently, (Esteban et al., 2016) proposed multi-way neural network architecture for modeling event based relational graph. The authors draw a synergistic relation between a static knowledge graph and an event set wherein the knowledge graph provide information about entities participating in events and new events in turn contribute to enhancement of knowledge graph. They do not capture the evolving dynamics of entities and model time as discrete points which limits its capacity to model complex temporal dynamics. (Jiang et al., 2016) models dependence of relationship on time to facilitate time-aware link prediction but do not capture evolving entity dynamics.

7. Conclusion

We propose a novel deep evolutionary knowledge network that efficiently learns non-linearly evolving entity representations over time in multi-relational setting. Evolutionary dynamics of both subject and object entities are captured by deep recurrent architecture that models historical evolution of entity embeddings in a specific relationship space. The occurrence of a fact is then modeled by multivariate point process that captures temporal dependencies across facts. The superior performance and high scalability of our method on large real-world temporal knowledge graphs demonstrate the importance of supporting temporal reasoning in dynamically evolving relational systems. Our work establishes previously unexplored connection between relational processes and temporal point processes with a potential to open a new direction of research on reasoning over time.

Acknowledgement

This project was supported in part by NSF IIS-1218749, NIH BIGDATA 1R01GM108341, NSF CAREER IIS-1350983, NSF IIS-1639792 EAGER, ONR N00014-15-1-2340, NVIDIA, Intel and Amazon AWS.

References

Aalen, Odd, Borgan, Ornulf, and Gjessing, Hakon. *Survival and event history analysis: a process point of view*. Springer, 2008.

Bordes, Antoine, Weston, Jason, Collobert, Ronan, and Bengio, Yoshua. Learning structured embeddings of knowledge bases. In *Conference on Artificial Intelligence*, number EPFL-CONF-192344, 2011.

Bordes, Antoine, Usunier, Nicolas, Garcia-Duran, Alberto, Weston, Jason, and Yakhnenko, Oksana. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pp. 2787–2795, 2013.

Boschee, Elizabeth, Lautenschlager, Jennifer, O'Brien, Sean, Shellman, Steve, Starz, James, and Ward, Michael. Icews coded event data. 2017.

Cox, D.R. and Lewis, P.A.W. Multivariate point processes. *Selected Statistical Papers of Sir David Cox: Volume 1, Design of Investigations, Statistical Methods and Applications*, 1:159, 2006.

Dai, Hanjun, Dai, Bo, and Song, Le. Discriminative embeddings of latent variable models for structured data. In *ICML*, 2016a.

Dai, Hanjun, Wang, Yichen, Trivedi, Rakshit, and Song, Le. Deep coevolutionary network: Embedding user and item features for recommendation. *arXiv preprint arXiv:1609.03675*, 2016b.

Daley, D.J. and Vere-Jones, D. *An introduction to the theory of point processes: volume II: general theory and structure*, volume 2. Springer, 2007.

Dong, Xin, Gabrilovich, Evgeniy, Heitz, Geremy, Horn, Wilko, Lao, Ni, Murphy, Kevin, Strohmann, Thomas, Sun, Shaohua, and Zhang, Wei. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 601–610, 2014.

Du, Nan, Wang, Yichen, He, Niao, and Song, Le. Time sensitive recommendation from recurrent user activities. In *NIPS*, 2015.

Du, Nan, Dai, Hanjun, Trivedi, Rakshit, Upadhyay, Utkarsh, Gomez-Rodriguez, Manuel, and Song, Le. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, 2016.

Esteban, Cristobal, Tresp, Volker, Yang, Yinchong, Baier, Stephan, and Krompa, Denis. Predicting the co-evolution of event and knowledge graphs. In *2016 19th International Conference on Information Fusion (FUSION)*, pp. 98–105, 2016.

Farajtabar, Mehrdad, Du, Nan, Gomez-Rodriguez, Manuel, Valera, Isabel, Zha, Hongyuan, and Song, Le. Shaping social activity by incentivizing users. In *NIPS*, 2014.

Farajtabar, Mehrdad, Wang, Yichen, Gomez-Rodriguez, Manuel, Li, Shuang, Zha, Hongyuan, and Song, Le. Coevolve: A joint point process model for information diffusion and network co-evolution. In *NIPS*, 2015.

Gutmann, Michael U and Hyvärinen, Aapo. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361, 2012.

Hawkes, Alan G. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

Hidasi, Balazs, Karatzoglou, Alexandros, Baltrunas, Linas, and Tikk, Domonkos. Session-based recommendations with recurrent neural networks. In *ICLR*, 2016.

Isham, V. and Westcott, M. A self-correcting pint process. *Advances in Applied Probability*, 37:629–646, 1979.

Jiang, Tingsong, Liu, Tianyu, Ge, Tao, Lei, Sha, Li, Suijan, Chang, Baobao, and Sui, Zhifang. Encoding temporal information for time-aware link prediction. 2016.

Leetaru, Kalev and Schrodt, Philip A. Gdelt: Global data on events, location, and tone. *ISA Annual Convention*, 2013.

Lin, Yankai, Liu, Zhiyuan, Sun, Maosong, and Zhu, Xuan. Learning entity and relation embeddings for knowledge graph completion. 2015.

Liu, Quan, Jiang, Hui, Evdokimov, Andrew, Ling, Zhen-Hua, Zhu, Xiaodan, Wei, Si, and Hu, Yu. Probabilistic reasoning via deep learning: Neural association models. *arXiv:1603.07704v2*, 2016.

Loglisci, Corrado and Malerba, Donato. Leveraging temporal autocorrelation of historical data for improving accuracy in network regression. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 10(1):40–53, 2017.

Loglisci, Corrado, Ceci, Michelangelo, and Malerba, Donato. Relational mining for discovering changes in evolving networks. *Neurocomputing*, 150, Part A:265–288, 2015.

Nickel, Maximilian, Tresp, Volker, and Kriegel, Hans-Peter. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 809–816, 2011.

Nickel, Maximilian, Murphy, Kevin, Tresp, Volker, and Gabrilovich, Evgeniy. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 2016a.

Nickel, Maximilian, Rosasco, Lorenzo, and Poggio, Tomaso. Holographic embeddings of knowledge graphs. 2016b.

Schein, Aaron, Zhou, Mingyuan, Blei, David, and Wallach, Hanna. Bayesian poisson tucker decomposition for learning the structure of international relations. *arXiv:1606.01855*, 2016.

Sharan, Umang and Neville, Jennifer. Temporal-relational classifiers for prediction in evolving domains. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 540–549, 2008.

Socher, Richard, Chen, Danqi, Manning, Christopher D, and Ng, Andrew. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pp. 926–934, 2013.

Toutanova, Kristina and Chen, Danqi. Observed versus latent features for knowledge base and text inference. 2015.

Wang, Yichen, Du, Nan, Trivedi, Rakshit, and Song, Le. Coevolutionary latent feature processes for continuous-time user-item interactions. In *NIPS*, 2016a.

Wang, Yichen, Theodorou, Evangelos, Verma, Apurv, and Song, Le. A stochastic differential equation framework for guiding online user activities in closed loop. *arXiv preprint arXiv:1603.09021*, 2016b.

Wang, Yichen, Xie, Bo, Du, Nan, and Song, Le. Isotonic hawkes processes. In *ICML*, 2016c.

Wang, Yichen, Williams, Grady, Theodorou, Evangelos, and Song, Le. Variational policy for guiding point processes. In *ICML*, 2017a.

Wang, Yichen, Ye, Xiaojing, Zhou, Haomin, Zha, Hongyuan, and Song, Le. Linking micro event history to macro prediction in point process models. In *AISTAT*, 2017b.

Wang, Zhen, Zhang, Jianwen, Feng, Jianlin, and Chen, Zheng. Knowledge graph embedding by translating on hyperplanes. 2014.

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

Yang, Shuang-Hong and Zha, Hongyuan. Mixture of mutually exciting processes for viral diffusion. In *ICML*, pp. 1–9, 2013.