TILP: DIFFERENTIABLE LEARNING OF TEMPORAL LOGICAL RULES ON KNOWLEDGE GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Compared with static knowledge graphs, temporal knowledge graphs (tKG), which can capture the evolution and change of information and knowledge, are more realistic and general. However, due to the complexity from introducing the notion of time, accurate link prediction based on explainable and comprehensible patterns is still a difficult problem. In this paper, we propose TILP, a differentiable framework for temporal logical rules learning. By designing constrained random walk mechanism and corresponding operators, we ensure the efficiency of our model. Furthermore, we discuss temporal features modelling in tKG, e.g., recurrence, temporal order, interval between pair of relations, and duration, and incorporate it into our learning process. TILP is compared with state-of-the-art baselines on two benchmark datasets and shows comparable performance. More importantly, we introduce some hard settings to test the robustness of different models, e.g., few training samples, biased data, and time shifting. In these cases, TILP works better than most state-of-the-art methods.

1 INTRODUCTION

Knowledge graphs (KGs) contain facts (e_s, r, e_o) representing relation r happening between subject entity e_s and object entity e_o, e.g., (David Beckham, plays for, Real Madrid). In real world, many relations are time-dependent, e.g., a player joining a team for a season, a politician holding a position for several years, and two persons' marriage lasting for decades. To represent the evolution and change of information, temporal knowledge graphs (tKGs) have been introduced. Temporal KGs extend the triples into quadruples (e_s, r, e_o, I) with an interval I indicating the valid period of the fact (e_s, r, e_o), e.g., (David Beckham, plays for, Real Madrid, $[2003, 2007]$).

To automatically construct a more complete KG, inferring missing facts with existing facts, i.e., link predication, is a common task for real-world applications. However, the introduction of temporal information makes this task more difficult. The important dynamic interactions between entities can not be captured by learning methods developed for static KGs. Recently, several embedding-based methods have been introduced to address the above limitation, e.g., HyTE (Dasgupta et al. (2018)), TNTComplEx (Lacroix et al. (2020)), and DE-SimplE (Goel et al. (2019)). The common principle adopted by these models is to create time-dependent embeddings for entities and relations.

As a useful representation for KGs, logical rules are interpretable and robust, providing explanations and insights for reference results, which are actually ignored by embedding-based methods. Since these rules contain temporal patterns in tKGs, we call them temporal logical rules. For large datasets, manual searching and confirmation are infeasible which creates the need for learning algorithms. Although the most recent works, e.g., StreamLearner (Omran et al. (2019)), and TLogic (Liu et al. (2021)), have shown some statistic learning methods, it is still interesting to develop a differentiable framework due to the benefits of using neural networks (Wang et al. (2017)).

In this paper, we propose TILP, a differentiable learning framework for temporal logical rules, with the novel mechanism of constrained random walk and the extended module for temporal features modelling. We achieve comparable performances to state-of-the-art methods, while providing logical explanations for the inference results. The main contributions of this work are summarized as follows:

- We introduce TILP, a novel differentiable symbolic framework based on constrained random walks on temporal knowledge graphs and temporal features modelling. It is the first differentiable approach that can learn temporal logical rules from tKGs without restrictions.
- We conduct experiments on two benchmark datasets, i.e., WIDIDATA12k and YAGO11k, and show comparable performance to state-of-the-art baselines. For test queries, we show not only the ranked list of candidates but also explanations for the prediction.
- We demonstrate the robustness of our method in the hard settings, i.e., few training samples, biased data, and time shifting, which simulate some realistic learning problems in temporal knowledge graphs. In these settings, TILP works better than most state-of-the-art methods.

2 RELATED WORKS

Embedding-based methods. Recently, embedding-based methods for tKGs start emerging for more accurate link prediction. The common principle of these methods is to create time-dependent embeddings for entities and relations, e.g., HyTE (Dasgupta et al. (2018)), TA-ComplEx (García-Durán et al. (2018)), TNTComplEx (Lacroix et al. (2020)), and DE-SimplE (Goel et al. (2019)). These embeddings are plugged into standard scoring functions in most cases. In addition to that, some works, e.g, TAE-ILP (Jiang et al. (2016)), and TimePlex (Jain et al. (2020)), start to investigate the influences of explicit temporal feature modelling, and merge it into the embedding algorithms. The main weakness of embedding-based method is lack of interpretability, as well as failure with previously unseen entities, relations, or timestamps.

Logical-rule-based methods. Logical-rule-based methods for link prediction on tKGs is mainly based on random walks. Although some works show the ability of learning temporal rules, they perform random walks in a very restrict version, which impairs the quality of learned rules. For example, Dynnode2vec (Mahdavi et al. (2018)) and Change2vec (Bian et al. (2019)) conduct random walks on tKG snapshots. DynNetEmbedd (Nguyen et al. (2018)) requires the edges in walks to forward in time. StreamLearner (Omran et al. (2019)) first extracts rules from static random walk, and then extend them into time domain. The consequence is that all body atoms in the extended rules have the same timestamp. TLogic (Liu et al. (2021)) is the most recent work which extracts temporal logical rules from the defined temporal random walks. The temporal constraints for temporal random walks are built on timestamps instead of intervals, and are fixed during the learning process. This inflexibility impairs its ability in temporal constraints learning. Furthermore, the two models, StreamLearner and TLogic, which can truly learn temporal logical rules, are both statistic learning methods which estimates rule confidence by counting the number of rule groundings and body groundings.

Differentiable rule learning. Several works utilize neural network architectures for rule learning, e.g., Neural-LP (Yang et al. (2017)), NTP (Rocktäschel & Riedel (2017)), Deepproblog (Manhaeve et al. (2018)), ∂ILP (Evans & Grefenstette (2018)), RuLES (Ho et al. (2018)), IterE (Zhang et al. (2019)), and NLProlog (Weber et al. (2019)). These works mainly focus on static KGs, lacking the ability of capturing temporal patterns. Converting from static KGs to temporal KGs is not a trivial extension. For example, Neural-LP reduces the rule learning problem to matrix multiplication. By defining the operators and neural control system, this framework realizes logical rule learning in an end-to-end way. However, the extra temporal constraints in logical rules break the Markovian property of random walks which serves as the foundation of most frameworks.

3 PRELIMINARIES

Temporal knowledge graph. A temporal knowledge graph (tKG) $\mathcal{G}$ is a collection of facts represented by a quadruple (e_s, r, e_o, I). This quadruple (e_s, r, e_o, I), also called edge or link, implies a relation r happening between the subject entity e_s and the object entity e_o during interval I. For an interval I, we define it with its start time t_s and end time t_e. To allow bidirectional random walk, we add inverse edges $\left(e_o, r^{-1}, e_s, I\right)$ where r^{-1} is called the inverse relation of r.

Link prediction. This task is to predict missing links with observed facts from the same tKG. To be specific, given a query $(e_s, r, ?, I)$ and the observed facts from the same tKG $\mathcal{G}$, a ranked list

of candidates for the missing object is required. For subject prediction, the query is formulated as $\left(e_o, r^{-1}, ?, I\right)$.

Temporal relation. Temporal relation between two timestamps t and t' can be $before$, $equal$, or $after$, which are denoted by $t < t'$, $t = t'$, and $t > t'$, respectively. For temporal relation between two intervals I and I', there are 13 types in total given by Allen's interval algebra (Allen (1983)). For example, $I := [t_s, t_e]$ is *before* $I' := [t'_s, t'_e]$ iff $t_e < t'_s$. However, our temporal logical rules would become too specific by directly using these 13 types of temporal relations. Thus, we group them into 3 classes: $TR \in \{before, touching, after\}$, where TR denotes the possible temporal relation between two intervals, $before$ is defined as previously mentioned, $after$ is the converse of $before$, and $touching$ is the group of all the other 11 types. It should be noted that timestamp can be considered as a special kind of interval with equal start time and end time. Thus, TR can be also used to describe temporal relations between timestamps.

Temporal logical rule. Let E_i and I_i for $i \in \{1, 2, \ldots, l+1\}$ be variables of entities and intervals, respectively. Let P_i for $i \in \{1, 2, \ldots, l+1\}$ denote predicates, which are equivalent to relations but grounded in logical rules.

A temporal logical rule R of length $l \in \mathbb{N}$ is defined as

$$P_{l+1}\left(E_1, E_{l+1}, I_{l+1}\right) \leftarrow \wedge_{i=1}^{l} P_i\left(E_i, E_{i+1}, I_i\right) \wedge_{j=1}^{l} \left(\wedge_{k=j+1}^{l+1} TR_{j,k}\left(I_j, I_k\right)\right) \tag{1}$$

The left side of R is called the rule head, with P_{l+1} being the head predicate, also called target predicate. For the link prediction task, target predicates are given, and would not change with rule length. Thus, we use P_h to denote it in the following sections. The right side is called the rule body, which is the conjunction of body predicates P_i, as well as body temporal relations $TR_{j,k}$. The rule is also called "chain-like" because the rule body corresponds to a walk from E_1 to E_{l+1}. The left arrow in R is called "entails", i.e., the rule body entails the rule head.

A rule is grounded with the replacement of the variables E_i and I_i by constant terms. For example, a grounding of the above temporal logical rule

$$ReceiveAward\left(E_1, E_2, I_2\right) \leftarrow NominatedFor\left(E_1, E_2, I_1\right) \wedge touching(I_1, I_2)$$

is given by the edges (Alice Bradley Sheldon, receive award, Nebula Award for Best Novelette, [1977, 1977]) and (Alice Bradley Sheldon, nominated for, Nebula Award for Best Novelette, [1977, 1977]) in WIKIDATA12k dataset. Since logical rules can be violated, rule confidence, the probability that a rule is correct, needs to be estimated.

4 Constrained Random Walk

Path constraint. Temporal logical rules can be considered as constraints for random walk on tKG. Generally speaking, these constraints can be divided into two classes: Markovian and non-Markovian. With Markovian constraints, we can perform a random walk without the consideration of previous visited edges. Otherwise, we need to record previous visited edges to ensure that non-Markovian constraints are satisfied.

For temporal logical rules shown in Eq.(1), Markovian constraints include body predicates $\bigwedge_{i=1}^{l}\{P_i\}$ and temporal relations between target interval and every body interval $\bigwedge_{i=1}^{l}\{TR_{i,l+1}\}$, and non-Markovian constraints include temporal relations between every pair of body intervals $\bigwedge_{i=1}^{l-1}\bigwedge_{j=i+1}^{l}\{TR_{i,j}\}$. Filtering operators f for these constraints are defined as

$$f_P\left(\left(e_s, r, e_o, I\right)\right) = \begin{cases} 1 & \text{if } r = P, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

$$f_{TR}\left(\left(e_s, r, e_o, I\right), \left(e'_s, r', e'_o, I'\right)\right) = \begin{cases} 1 & \text{if } g(I, I') = TR, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

$$g\left(I, I'\right) = \begin{cases} before & \text{if } t_e < t'_s, \\ after & \text{if } t_s > t'_e, \\ touching & \text{otherwise.} \end{cases} \tag{4}$$

where f_P and f_{TR} denote filtering operators for predicate P and temporal relation TR respectively, and g is the temporal relation evaluation function.

Constrained random walk. Since a successful random walk should satisfy both classes of constraints, we first perform random walk under Markovian ones, and then filter out the results according to non-Markovian ones. To ensure the efficiency of our framework, we use matrix multiplications, and build scalable filtering operators.

For step i, the operator M_{i,CM_i} under corresponding Markovian constraints $CM_i = \{P_i, TR_{i,l+1}\}$ is defined as:

$$\begin{aligned}(M_{i,CM_i})_{x,y} &= f_{CM_i}\left((e_x, r_{x,y}, e_y, I_{x,y})\right) \\ &= f_{P_i}\left((e_x, r_{x,y}, e_y, I_{x,y})\right) f_{TR_{i,l+1}}\left((e_x, r_{x,y}, e_y, I_{x,y}), I_{l+1}\right)\end{aligned} \tag{5}$$

If there exist multiple edges between entities x and y, $(M_{i,CM_i})_{x,y}$ would become the union of all results.

With these operators, we can actually find *all* the paths between any pair of entities that satisfy Markovian constraints. Suppose we start from entity x, after l steps of random walk under corresponding Markovian constraints $\bigwedge_{i=1}^{l}\{CM_i\}$. All the entities we can arrive are

$$\mathbf{v}_{l+1} = \prod_{i=1}^{l} M_{i,CM_i}\mathbf{v}_x \tag{6}$$

where $\mathbf{v}$ is the indicator vector. When arriving some entities, the corresponding entry would be greater than 0. Similar to Breadth First Search, by using these operators, we can obtain constrained random walks W_{CM} given by $\left((e_1, r_1, e_2, I_1), \ldots, (e_l, r_l, e_{l+1}, I_l)\right)$.

For non-Markovian constraints $CN_{i,j} = \{TR_{i,j}\}$, we can apply filtering function to these constrained random walks

$$f_{CN}(W_{CM}) = \prod_{i=1}^{l-1}\prod_{j=i+1}^{l} f_{CN_{i,j}}(I_i, I_j) = \prod_{i=1}^{l-1}\prod_{j=i+1}^{l} f_{TR_{i,j}}(I_i, I_j) \tag{7}$$

The filtered walks W_C are the final results of our constrained random walk. In fact, by introducing new filtering functions, this framework can have more possibilities. For example, using the numerical comparison filtering function (Wang et al. (2019)), our method is able to learn logical rules with numerical features.

5 Learning temporal logical rules

The framework mainly consists of two processes: rule learning and rule application. In the learning process, given a positive example (e_s, r_h, e_o, I) for target predicate P_h, we are going to find all the paths from entity e_s to entity e_o in the tKG $\mathcal{G}$. Then we extract temporal logical rules from these paths, and estimate their confidence. The distribution of different temporal features introduced in this sections is also learned in this process.

In the application process, given a query $(e_s, r_h, ?, I)$ and a set of temporal logical rules R related to P_h, we are going to find *all* the random walks that satisfy these rules. By calculating arriving rate of each rule $\alpha(R)$ and aggregating all the rules, we can obtain the temporal logical rule score ϕ_{TLR} for all the candidates. In addition, we use all the evidences related to these candidates to evaluate their temporal feature scores ϕ_{tfm} which, together with temporal logical rule scores, form the final scores ϕ_{TILP}.

Rule confidence. For the extracted temporal logical rules, we need to estimate their confidence for rule application. Instead of estimating confidence for every single rule, we create attention vectors for every type of constraints, and reuse them in different rules. Sharing confidence of constraints creates a joint and robust learning process, and largely reduces model parameters.

In our temporal logical rules, there are two types of constraints: predicates and temporal relations. For each type, we create attention vectors w_P and w_{TR} to denote the confidence of using this

constraint. Obviously, there are many factors that can affect these attention vectors, e.g, target predicate, target interval, length of rule, entity properties, and so on. Furthermore, to deal with varying length of rules, another attention vector w_{Len} is also required.

To make this problem solvable, some simplifications are made here. We suppose that the attention vectors of predicates and temporal relations are dependent on target predicate and length of rule, and the attention vector of rule length is determined by target predicate. Inspired by Neural-LP (Yang et al. (2017)), we design a set of mapping functions based on RNN:

$$\mathbf{h}_{i,l} = \text{update}\left(\mathbf{h}_{i-1,l}, f_{embdd}(\mathbf{x}_l, r_h)\right) \tag{8}$$

$$w_{P_i,l} = \text{softmax}\left(W_P \mathbf{h}_{i,l} + b_P\right) \tag{9}$$

$$w_{TR_{i,h},l} = \text{softmax}\left(W_{TR} \mathbf{h}_{i,l} + b_{TR}\right) \tag{10}$$

$$w_{TR_{i,j},l} = \text{softmax}\left(W'_{TR}[\mathbf{h}_{i,l}; \mathbf{h}_{j,l}] + b'_{TR}\right) \tag{11}$$

$$w_{Len} = \text{softmax}\left(W_{Len} f_{embdd}(\mathbf{x}_0, r_h) + b_{Len}\right) \tag{12}$$

where $\mathbf{h}_{i,l}$ denotes the ith-step hidden layer vector in the l-length rule, $\mathbf{x}_l$ denotes the corresponding embedding of input, f_{embdd} is an embedding lookup function, and all the W and b are learnable parameters.

Given these attention vectors, the confidence of a rule R in the form of Eq.(1) is calculated as:

$$\begin{aligned} score(R) =& f_{embdd}(w_{Len}, l) \prod_{i=1}^{l} f_{embdd}(w_{P_i,l}, r_i) \prod_{i=1}^{l} f_{embdd}(w_{TR_{i,l+1},l}, g(I_i, I_{l+1})) \\ & \prod_{i=1}^{l-1} \prod_{j=i+1}^{l} f_{embdd}(w_{TR_{i,j},l}, g(I_i, I_j)) \end{aligned} \tag{13}$$

Temporal feature modelling. Temporal relations between intervals mainly describe the order of different predicates. In fact, more temporal features can be utilized to better improve prediction accuracy. Inspired by TimePlex (Jain et al. (2020)), we design a new module for temporal feature modelling. The main features considered here include $Recurrence$, $TemporalOrder$, $RelationPairInterval$ and $Duration$.

- $Recurrence$ describes the distribution of recurrence of the same event. Different from TimePlex's description $(e_s, r, e_o, *)$, we define the same event in a more general form $(e_s, r, *, *)$. It should be noted that the form $(*, r, e_o, *)$ is also involved as $(e_o, r^{-1}, *, *)$. For each relation r, we use a parameter $p_{rec}(r)$ to denote the probability that this relation will happen again for the same subject.
- $TemporalOrder$ describes the distribution of temporal order between pair of relations happening for the same subject $(e_s, r, *, I), (e_s, r', *, I')$. It is implied by temporal relations in temporal logical rules to some extent. However, for two $touching$ intervals, we still can not tell which one happens earlier. Thus, a parameter $p_{order}(r, r')$ is adopted for every pair of relations to denote the probability that r happens earlier than r'. In practice, if there are multiple events of the same relation $\{(e_s, r'_k, *, I'_k)\}_k$, we will choose the closest one, i.e., $argmin_k |t'_{s,k} - t_s|$.
- $RelationPairInterval$ describes the time interval distribution between pair of relations happening for the same subject $(e_s, r, *, I), (e_s, r', *, I')$. Different from TimePlex, we do not restrict this distribution to be Gaussian. Instead, we use two types of distributions: Gaussian and exponential, with parameters $\mu_{pair}(r, r'), \sigma_{pair}(r, r')$ and $\lambda_{pair}(r, r')$, respectively. Gaussian distribution is preferred when there is a roughly fixed interval such as the birth date and death date of the same person, while exponential distribution can be better with two strongly correlated relations. Similarly, we choose the closest one for multiple events of the same relation in practice.
- $Duration$ describes the distribution of length of interval. We suppose the duration of each relation r follows Gaussian distribution with parameters $\mu_d(r), \sigma_d(r)$. It is common in large tKGs that the exact date of some events are missing. With duration modelling, we can fill in these missing dates and improve our model performance.

With these temporal feature distributions, we can further evaluate the candidates of a query. Similar to TimePlex, a linear function of probability is used as scoring function ϕ_{rec}, ϕ_{order}, ϕ_{pair}. However, more evidences are used by our model in the evaluation process. The evidences of TimePlex only include facts happening between the known entity and every candidate, while our model extends with the constrained random walks. It should be noted that these temporal feature distributions are re-learned on constrained random walks.

Given a query $(e_s, r_h, ?, I_h)$ and a candidate e_c, all the evidences using in the temporal feature modelling module include facts happening between e_c and e_s, i.e, $\mathcal{F}_{c,s} := \{(e_c, *, e_s, *)\}$, facts happening on e_c but not on e_s, i.e., $\mathcal{F}_{c,\bar{s}} := \{(e_c, *, *, *)\} - \mathcal{F}_{c,s}$ and constrained random walks W_C between e_c and e_s. The temporal feature modelling score ϕ_{tfm} of the candidate is given as

$$\phi_{tfm}(W_C) = \gamma_{order} \sum_{i,j} \phi_{order}(r_i, r_j, I_i, I_j) + \gamma_{pair} \sum_{i,j} \phi_{pair}(r_i, r_j, I_i, I_j) \tag{14}$$

$$\phi_{tfm}(e_c) = \gamma_1 \phi_{tfm}(\mathcal{F}_{c,s}) + \gamma_2 \phi_{tfm}(\mathcal{F}_{c,\bar{s}}) + \gamma_3 \phi_{tfm}(W_C) \tag{15}$$

Candidate ranking. Suppose by applying a temporal logical rule R, we find in total N successful random walks, and N_c of them arriving at entity e_c. With the assumption that each walk has equal confidence, we can calculate the arriving rate $\alpha_c := N_c/N$. By aggregating all rules learned for the target predicate, we obtain the logical rule score ϕ_{TLR}. Combined with the temporal feature modelling score, the final score ϕ_{TILP} becomes:

$$\phi_{TLR}(e_c) = \sum_R \alpha_c(R) score(R) \tag{16}$$

$$\phi_{TILP}(e_c) = \gamma_{TLR} \phi_{TLR}(e_c) + \gamma_{tfm} \phi_{tfm}(e_c) \tag{17}$$

The training process is divided into two phases. In the first phase, the attention vectors for predicates, temporal relations and rule length are learned by maximizing the confidence of correct candidates. In the second phase, all the distribution parameters of temporal features are fitted with training samples. Then we train the parameters of weights γ for the temporal feature modelling module with frozen attention vectors, i.e, ϕ_{TLR} is used for prediction in the first stage, and ϕ_{TILP} is adopted in the second stage.

6 Experiments

6.1 Experimental Setup

We report on experiments with two standard tKG datasets: WIKIDATA12k and YAGO11k (Dasgupta et al. (2018)). They contain temporal facts like (John Okell, worksAt, SOAS University of London, $[1959, 1999]$). In this experiment, only the year of the facts is used since the month and day of most facts are missing. For datasets with higher granularity, our model performance is supposed to be improved due to more precise temporal relations can be found. For the link predication task (e_s, r, e_o, I), we generate a list of ranked candidates for both object prediction $(e_s, r, ?, I)$ and subject prediction $\left(e_o, r^{-1}, ?, I\right)$. The maximum rule length is set to 5 for both datasets. The standard metrics mean reciprocal rank (MRR), hits@1, hits@10 are adopted. Briefly speaking, MRR is the average of reciprocal ranks of the correct answer across all queries, and hits@k indicates the proportion of queries for which the correct answer ranks within the top k. Similar to Jain et al. (2020), we perform time-aware filtering which gives a more valid performance evaluation.

We compare TILP with the state-of-the-art baselines in two dimensions: static v.s. temporal, logical-rule-based v.s. embedding-base. The static logical-rule-based methods include Neural-LP (Yang et al. (2017)), and AnyBURL (Meilicke et al. (2020)). The temporal logical-rule-based method is TLogic (Liu et al. (2021)). The static embedding-base model is ComplEx (Trouillon et al. (2016)). The temporal embedding-base model include TA-ComplEx (García-Durán et al. (2018)), HyTE (Dasgupta et al. (2018)), DE-SimplE (Goel et al. (2019)), TNT-Complex (Lacroix et al. (2020)), and TimePlex (Jain et al. (2020)). The results for all embedding-based models are from Jain et al. (2020). Ablation studies on the temporal feature modelling module (TILP w/o tfm) are also conducted.

Table 1: Link prediction performance on the two benchmark datasets

Dataset	WIKIDATA12k			YAGO11k		
Model	MRR	hit@1	hit@10	MRR	hit@1	hit@10
Neural-LP	0.1823	0.0908	0.3848	0.1001	0.0401	0.1845
AnyBURL	0.1908	0.1030	0.3904	0.0908	0.0378	0.1814
TLogic	0.2536	0.1754	0.4424	0.1545	0.1180	0.2309
ComplEx	0.2482	0.1430	0.4890	0.1814	0.1146	0.3111
TA-ComplEx	0.2278	0.1269	0.4600	0.1524	0.0936	0.2626
HyTE (TransE)	0.2528	0.1470	0.4826	0.1355	0.0332	0.2981
DE-SimplE	0.2529	0.1468	0.4905	0.1512	0.0875	0.2674
TNT-Complex	0.3010	0.1973	0.5069	0.1801	0.1102	0.3128
TimePlex (base)	0.3238	0.2203	0.5279	0.1835	0.1099	0.3186
TimePlex	**0.3335**	0.2278	**0.5320**	0.2364	**0.1692**	0.3671
TILP (w/o tfm)	0.3114	0.2152	0.5077	0.1880	0.1336	0.3089
TILP	0.3328	**0.2342**	0.5289	**0.2411**	0.1667	**0.4149**

6.2 Results and Analysis

The results of the experiments are shown in Table 1. TILP has comparable performances with respect to all metrics. For some metrics, TimePlex performs slightly better, but TILP can also provide temporal logical rules and body groundings that support the predictions. Some examples of learned rules and groundings in WIKIDATA12k dataset are given below.

Rule 1:

$$memberOf\,(E_1, E_4, I_4) \leftarrow memberOf\,(E_1, E_2, I_1) \wedge memberOf^{-1}\,(E_2, E_3, I_2)$$
$$\wedge memberOf\,(E_3, E_4, I_3) \wedge_{i=1}^{3} (\wedge_{j=i+1}^{4} touching(I_i, I_j))$$

Grounding: E_1 = Somalia, E_2 = International Development Association, E_3 = Kingdom of the Netherlands, E_4 = International Finance Corporation, I_1 = $[1962, present]$, I_2 = $[1961, present]$, I_3 = $[1956, present]$, I_4 = $[1962, present]$.

Rule 2:

$$receiveAward\,(E_1, E_4, I_4) \leftarrow nominatedFor\,(E_1, E_2, I_1) \wedge nominatedFor^{-1}\,(E_2, E_3, I_2)$$
$$\wedge receiveAward\,(E_3, E_4, I_3) \wedge before(I_1, I_2) \wedge_{i=1}^{2} after(I_i, I_3) \wedge_{j=1}^{3} before(I_j, I_4)$$

Grounding: E_1 = ZDF, E_2 = International Emmy Award for best drama series, E_3 = DR, E_4 = Peabody Awards, I_1 = $[2005, 2005]$, I_2 = $[2009, 2009]$, I_3 = $[1997, 1997]$, I_4 = $[2013, 2013]$.

Granularity of temporal relations: We observe that TLogic doesn't perform well on these two *interval-based* tKGs. Several reasons can be found for this phenomenon. First, the foundation of the temporal walks in TLogic is the relative order of time points. Without the introduction of temporal relation between intervals, TLogic can not be simply extended for interval-based tKGs. Second, to improve model efficiency, TLogic uses sampling strategy with the control of the total number of temporal walks. This strategy actually impairs model performance due to no guarantee for successful long-distance random walks. It is proven by the facts that a lot of temporal logical rules we found in these two datasets are of length 5, which can be challenges for TLogic.

Temporal feature modelling: To begin with, a safe conclusion is that time-aware learning is necessary for tKGs since all static learning methods lose to their counterparts with temporal learning abilities. Then, two different directions are chosen by logical-rule-based methods and embedding-based methods. The former introduces temporal relations into logical rules, while the latter uses time-dependent embeddings. Even though these ideas make some improvements, there are few works on explicit temporal features modelling, i.e., $Recurrence$, $TemporalOrder$, $RelationPairInterval$ and $Duration$. Introduction of these distributions as extra information is beneficial, which has been proven by both TimePlex and our model. Our extensions of the original module are non-trivial since evidences of random walks are added, which can not be realised by any embedding-based methods.

6.3 HARD SETTINGS

Although our model does not outperform the-state-of-art embedding-based method for some metrics, there are several advantages in consideration of realistic problems in tKGs. To simulate these situations, we propose the following hard settings. In this part, we mainly compare our model with two representative models for temporal link prediction: TLogic (logical-rule-based) and TimePlex (embedding-based).

Few training samples. Generally speaking, structured data are always hard to obtain. When establishing a new database, it is important that our model can start with few training samples. One advantage of our learned logical rules compared with embedding-based method is that we do not need a lot of training samples. In experiments, we randomly reduce the number of training samples in training set and evaluate different models for the two datasets. To alleviate the effects of different data distributions, we repeat this experiments for 5 rounds. The results are shown in Fig.1, where we draw the average MRR curve with err bars. When the training set size decreases, TILP outperforms all the baseline methods. Through constrained random walks, TILP is able to capture all the patterns related to a query relation which are independent on entities. Reducing training set size only changes the frequency of different patterns. To contrast, embedding-based methods require enough training samples to learn good embeddings of entities and relations.

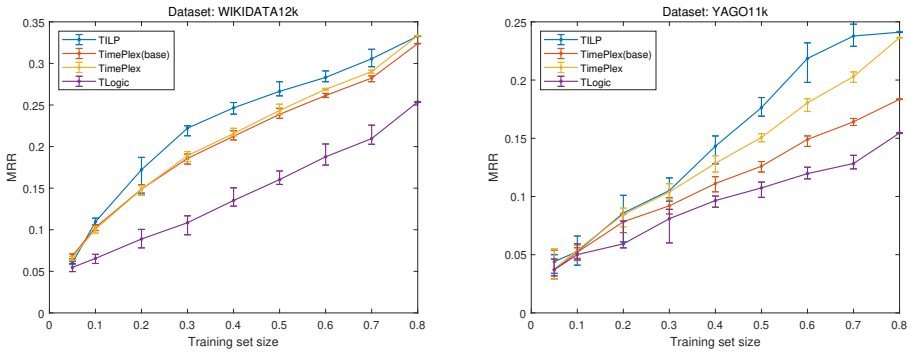

Figure 1: Link prediction performance with few training samples

Biased data. Another common problem in knowledge graph learning is data imbalance. For some rare relations, it is hard to collect samples. For example, in WIKIDATA12k dataset where there are 40621 edges in total, the number of edges for relation $capitalOf$ and $residence$ are only 86 and 80, respectively. Thus, a good model for every relation should be able to deal with biased data. For the two datasets, we first obtain statistic results on the number of edges for each relation. Then for a specific relation, we randomly reduce 50% of the edges in the training set, and evaluate the performance changes caused by this setting. We conduct the experiments for every relation separately, and repeat for 5 rounds to alleviate the influence of different data distributions. The original test set from random generation contains few queries of rare relations. To guarantee enough test queries for each relation, we adjust the distribution of queries in test set, trying to set the number of queries of different relations to be equal. If queries of a certain relation are not enough, we would randomly choose half of them. In Fig.2, the average change of MRR for different models is shown, where err bars are ignored for reading. Since the attention vectors of predicates, temporal relations and rule length are relation-dependent in TILP, it is less affected by data imbalance. To contrast, embeddings of entities are shared among all the relations, making embedding-based method suffer more in this situation.

Time shifting. For tKG, the problem of time shifting is also noteworthy. Intuitively, more overlapping between the period of training set and that of test set brings the higher model performance. However, this condition can be violated in scenarios such as link forecasting. Similar to TLogic, we re-split the whole dataset according to the start time of each relation. In practice, we put the edges missing start time into training set, and correct the edges with wrong start time (greater than the year of 2022). For WIKIDATA12k dataset, the start time range for training set, validation set and test set are $[0, 2008]$, $[2008, 2012]$, $[2012, 2018]$, respectively. For YAGO11k dataset, the start time range for training set, validation set and test set are $[-431, 2006]$, $[2006, 2011]$, $[2011, 2022]$, respectively.

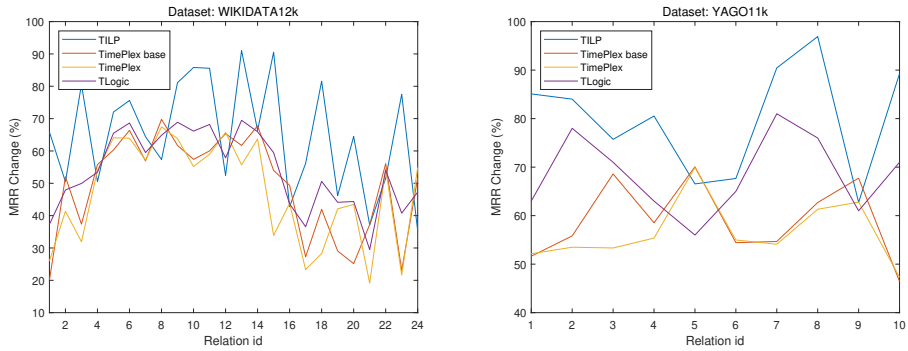

Figure 2: Link prediction performance with biased data

The results are shown in Table 2. One major limitation of most time-aware embedding-based methods is to utilize *absolute* timestamp as anchors, causing them to fail on both time shifting settings and inductive settings (Liu et al. (2021)). Instead, TILP extracts temporal logical rules with *relative* temporal relation, bringing it more possibilities in transfer learning.

Table 2: Link prediction performance with time shifting setting

Dataset	WIKIDATA12k			YAGO11k		
Model	MRR	hit@1	hit@10	MRR	hit@1	hit@10
TimePlex (base)	0.0975	0.0647	0.1588	0.0581	0.0299	0.1085
TimePlex	0.0976	0.0651	0.1580	0.0639	0.0341	0.1146
TLogic	0.1508	0.0934	0.3201	0.1107	0.0621	0.1804
TILP	**0.2657**	**0.1623**	**0.4412**	**0.2069**	**0.1194**	**0.3686**

7 Conclusion

We have proposed TILP, the first differentiable framework for temporal logical rules learning, and applies these rules for link predication task on temporal knowledge graphs. The framework considers both temporal logical rules and temporal features when generating answers for test queries. Experiments on two standard datasets indicate that TILP achieves comparable performance and provides logical explanations for the predictions. In addition, we consider some realistic problems in temporal knowledge graph learning, where TILP works better than most state-of-the-art methods. As future work, it would be interesting to predict event intervals with temporal logical rules. In this task, the learned rules must contain both numerical values and temporal relations, which further improves the expressive power of logical rules.

References

James F Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26 (11):832–843, 1983.

Ranran Bian, Yun Sing Koh, Gillian Dobbie, and Anna Divoli. Network embedding and change modeling in dynamic heterogeneous networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 861–864, 2019.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. HyTE: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2001–2011, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1225. URL https://aclanthology.org/D18-1225.

Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research*, 61:1–64, 2018.

Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. Learning sequence encoders for temporal knowledge graph completion. *arXiv preprint arXiv:1809.03202*, 2018.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. Diachronic embedding for temporal knowledge graph completion, 2019. URL https://arxiv.org/abs/1907.03143.

Vinh Thinh Ho, Daria Stepanova, Mohamed H Gad-Elrab, Evgeny Kharlamov, and Gerhard Weikum. Rule learning from knowledge graphs guided by embedding models. In *International Semantic Web Conference*, pp. 72–90. Springer, 2018.

Prachi Jain, Sushant Rathi, Soumen Chakrabarti, et al. Temporal knowledge base completion: New algorithms and evaluation protocols. *arXiv preprint arXiv:2005.05035*, 2020.

Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Baobao Chang, Sujian Li, and Zhifang Sui. Towards time-aware knowledge graph completion. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 1715–1724, 2016.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. Tensor decompositions for temporal knowledge base completion, 2020. URL https://arxiv.org/abs/2004.04926.

Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs, 2021. URL https://arxiv.org/abs/2112.08025.

Sedigheh Mahdavi, Shima Khoshraftar, and Aijun An. dynnode2vec: Scalable dynamic network embedding. In *2018 IEEE international conference on big data (Big Data)*, pp. 3762–3765. IEEE, 2018.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *Advances in Neural Information Processing Systems*, 31, 2018.

Christian Meilicke, Melisachew Wudage Chekol, Manuel Fink, and Heiner Stuckenschmidt. Reinforced anytime bottom up rule learning for knowledge graph completion. *arXiv preprint arXiv:2004.04412*, 2020.

Giang H Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim. Dynamic network embeddings: From random walks to temporal random walks. In *2018 IEEE International Conference on Big Data (Big Data)*, pp. 1085–1092. IEEE, 2018.

Pouya Ghiasnezhad Omran, Kewen Wang, and Zhe Wang. Learning temporal rules from knowledge graph streams. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*, 2019.

Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. *Advances in neural information processing systems*, 30, 2017.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International conference on machine learning*, pp. 2071–2080. PMLR, 2016.

Po-Wei Wang, Daria Stepanova, Csaba Domokos, and J Zico Kolter. Differentiable learning of numerical rules in knowledge graphs. In *International Conference on Learning Representations*, 2019.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12): 2724–2743, 2017.

Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. Nlprolog: Reasoning with weak unification for question answering in natural language. *arXiv preprint arXiv:1906.06187*, 2019.

Fan Yang, Zhilin Yang, and William W. Cohen. Differentiable learning of logical rules for knowledge base reasoning, 2017. URL https://arxiv.org/abs/1702.08367.

Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. Iteratively learning embeddings and rules for knowledge graph reasoning. In *The World Wide Web Conference*, pp. 2366–2377, 2019.