Temporal Knowledge Base Completion: New Algorithms and Evaluation Protocols

Prachi Jain[*1], **Sushant Rathi**[*1], **Mausam**[1] and **Soumen Chakrabarti**[2]
[1] Indian Institute of Technology Delhi
[2] Indian Institute of Technology Bombay
{p6.jain, rathisushant5}@gmail.com, mausam@cse.iitd.ac.in, soumen.chakrabarti@gmail.com

Abstract

Temporal knowledge bases associate relational (s, r, o) triples with a set of times (or a single time instant) when the relation is valid. While time-agnostic KB completion (KBC) has witnessed significant research, temporal KB completion (TKBC) is in its early days.

In this paper, we consider predicting missing entities (link prediction) and missing time intervals (time prediction) as joint TKBC tasks where entities, relations and, time are all embedded in a uniform, compatible space. We present TIMEPLEX, a novel time-aware KBC method, that also automatically exploits the recurrent nature of some relations and temporal interactions between pairs of relations. TIMEPLEX achieves state-of-the-art performance on both prediction tasks.

We also find that existing TKBC models heavily overestimate link prediction performance due to imperfect evaluation mechanisms. In response, we propose improved TKBC evaluation protocols for both link and time prediction tasks, dealing with subtle issues that arise from the partial overlap of time intervals in gold instances and system predictions.

1 Introduction

A knowledge base (KB) is a collection of triples (s, r, o), with a subject s, a relation type r and an object o. KBs are usually incomplete, necessitating completion (KBC) of triples not provided in the collection. A KBC model is often evaluated by its performance on link prediction: supplying missing arguments to queries of the form $(s, r, ?)$ and $(?, r, o)$.

Many relations are transient or impermanent. Temporal KBs annotate each fact (event) with the time period in which it holds or occurs. A person is born in a city in an instant, a politician can be a president of a country for several years, and a marriage may last between years and decades. Temporal KBs represent these by (s, r, o, T) tuples, where T is a span of time. Temporal KBC (TKBC) performs completion of temporal KBs. It is also primarily evaluated by link prediction queries $(s, r, ?, T)$ and $(?, r, o, T)$. Recently, time prediction $(s, r, o, ?)$ has also been considered, though not in a general model-independent form.

While KBC has been intensely researched, TKBC is only beginning to be explored. TKBC presents novel challenges in task definition and modeling. For instance, little is known about how best to predict intervals for $(s, r, o, ?)$ queries, or how to evaluate a system response interval. Moreover, we show that even for link prediction queries, evaluation faces subtle complications owing to the inclusion of T in $(s, r, ?, T)$ queries and requires careful rethinking of evaluation protocols. In this paper, we propose improved evaluation protocols for both link and time prediction tasks in a TKBC.

TKBC also brings unique modeling opportunities. We first make the observation that a state-of-the-art (time-agnostic) KBC model, CX (Trouillon et al., 2016) performs at least as well as existing time-aware TKBC models in link prediction! This is surprising, since we believe that TKBC systems should make better predictions by exploiting aggregated knowledge about typical intervals of relation validity, or statistical constraints between events and relations. E.g., a person must be born before becoming president, which must precede death. A nation rarely has two presidents at the same time. A TKBC system can learn such (possibly soft) constraints from training data, and use such knowledge to make more informed link and time predictions.

In response, we present a novel model, TIMEPLEX, which outperforms both time-agnostic and time-aware TKBC models on benchmark datasets

[*] Equal contribution

for both link and time prediction. At a high level, our approach performs tensor factorization of a temporal KB, using complex-valued embeddings for relations, entities and time points. It enables these embeddings to capture implicit temporal relationships across both facts and relations, by providing temporal differences as explicit features.

In summary, our paper makes the following contributions:

- We propose evaluation protocols for link and time prediction queries for TKBC. For link prediction, we highlight that existing evaluation seriously over-estimates system performance, and offer a time-aware filtering method for more reliable evaluation. For time prediction, we propose an evaluation metric that rewards a model for predicting an interval with partial overlap with gold interval, as well as for nearness to gold in case there is no overlap.
- We present TIMEPLEX, a TKBC algorithm that factorizes a temporal KB using entity, relation and time embeddings. It can learn and exploit soft ordering and span constraints between potentially all relation pairs (including that of a relation with itself). TIMEPLEX beats recent and competitive algorithms on several recent standard TKBC data sets.

We will release an open-source implementation[1] of all models and experiments discussed in this paper for future research.

2 Preliminaries and Prior Work

2.1 Time-Agnostic KBC

Time-agnostic KBC has been intensely researched (Bordes et al., 2013; Yang et al., 2015; Nickel et al., 2016; Jain et al., 2018). The most common approach is to learn entity and relation embeddings by scoring an (s, r, o) triple as a function over constituent embeddings. All models train for embeddings using loss functions that impose that scores for known triples should be higher than (randomly sampled) negative triples.

Our work is based on ComplEx (Trouillon et al., 2016), abbreviated as CX here. It embeds s, r, o to vectors of complex elements $\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o} \in \mathbb{C}^D$. CX defines the score ϕ of a fact (s, r, o) as $\Re(\langle \boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}^\star \rangle)$ where

$$\langle \boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o} \rangle = \sum_{d=1}^{D} s[d]\ r[d]\ o^\star[d] \qquad (1)$$

is a 3-way inner product, $\boldsymbol{o}^\star$ is the complex conjugate of $\boldsymbol{o}$, and $\Re(c)$ is real part of $c \in \mathbb{C}$. If real embeddings are used instead, the above formula reduces to DistMult (Yang et al., 2015).

We choose CX as our base model, because it has been broadly found to outperform additive counterparts such as TransE (Bordes et al., 2013) and other projective variants (Wang et al., 2014; Ji et al., 2015). Inspired by the results from Lacroix et al. (2018), we make the following change in training CX: instead of negative sampling, we train it using a full softmax with cross entropy loss. We find that Complex trained in this manner performs competitively with the state-of-the-art models on KBC. We use this CX variant as a strong time-agnostic baseline for TKBC link-prediction tasks.

2.2 Temporal KBC Problem Setup

For TKBC, the domain of "all time" is denoted $\mathfrak{T}$. Any triple (s, r, o) is valid for some $T \subseteq \mathfrak{T}$. For simplicity, we assume time has been scaled and discretized to a suitable granularity, and is thereafter represented by integers. Thus, T and $\mathfrak{T}$ can be regarded as sets of integers. In practice, addressing the case where T is a single contiguous interval $[t_b, t_e]$ is sufficiently interesting and useful. Some event-style facts (e.g., born in) may have $t_b = t_e$. Overall, temporal KB facts looks like (s, r, o, T), and are partitioned into train, dev and eval (test) folds, abbreviated as $\mathrm{tr}, \mathrm{de}, \mathrm{ev}$. System predictions are abbreviated as pr.

TKBC is primarily evaluated via link prediction queries such as $(?, r, o, T)$ and $(s, r, ?, T)$. It also admits time prediction queries $(s, r, o, ?)$. In this form of TKBC, KB incompleteness exists at all times — the eval fold may include instances from any instant or interval in time, arbitrarily overlapping train and dev fold instances.

We also note that there is a related task in the literature, which is concerned with extrapolating to *future* knowledge graphs given past history (Trivedi et al., 2017; Jin et al., 2019). Having observed the history of the KB graph $G_{<t} = G_1, \ldots, G_{t-1}$ leading up to a given epoch $t-1$, the task here is to predict the state of the KB graph G_t at time t as a distribution $\Pr(G_t | G_{\leq t})$, and extend up to some horizon $t + \delta t$. It is not clear whether there is an efficient way to answer ad-hoc link or time prediction queries using models for this task.

2.3 Recent TKBC Systems

State of the art TKBC models include HyTE (Dasgupta et al., 2018) and TA-family of algorithms

[1] github.com/dair-iitd/tkbi.git

(García-Durán et al., 2018). HyTE coarsens time into variable-sized bins and converts a tuple (s, r, o, T) into (s, r, o, t) tuples for all bins t where $t \cap T \neq \emptyset$. It embeds each t to $\boldsymbol{t} \in \mathbb{R}^D$ with $\|\boldsymbol{t}\|_2 = 1$. It then projects $\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}$ on $\boldsymbol{t}$ by

$$\boldsymbol{x} \downarrow \boldsymbol{t} = \boldsymbol{x} - (\boldsymbol{x} \cdot \boldsymbol{t})\boldsymbol{t}, \text{ where } \boldsymbol{x} \in \{\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}\}. \quad (2)$$

After projection, a tuple (s, r, o, t) is scored via TransE, i.e. $\phi(s, r, o, t) =$

$$\boldsymbol{s} \downarrow \boldsymbol{t} + \boldsymbol{r} \downarrow \boldsymbol{t} - \boldsymbol{o} \downarrow \boldsymbol{t}. \quad (3)$$

HyTE was shown to be superior to several time-agnostic baselines, TransE, TransH (Wang et al., 2014), HolE (Nickel et al., 2016) and time-aware t-TransE (Jiang et al., 2016b) on link prediction. However, we find that our strong baseline of CX with full softmax performs at par with, or better than HyTE.

To our knowledge, HyTE is also the first work to introduce time prediction for evaluating TKBC. However, their evaluation is internal to their model, as they assess the prediction of a correct time *bin*, not the exact time-interval.

The "time-aware" (TA) family of algorithms (García-Durán et al., 2018) propose a rather different method of encoding (r, t) as a specially coded token sequence. E.g., $r =$ born in and $t = 1961$ would be written as tokens "[born, 1y, 9y, 6y, 1y]" and encoded by an LSTM into a combined representation $\overrightarrow{(\boldsymbol{r}, \boldsymbol{t})}$, which would then be used with $\boldsymbol{s}$ and $\boldsymbol{o}$ as in TransE (TA-TransE) or DistMult (TA-DM).

There is also some work in explicitly providing ordering constraints between relations (e.g., born, graduated, married, died) (Jiang et al., 2016a; Garcia-Duran and Niepert, 2018). In contrast, our work assumes no such additional engineered inputs. Relation ordering constraints, if any, are learned as model weights in a soft manner, jointly with neural embeddings of entities, relations, and time instants.

2.4 Standard evaluation schemes

Link Prediction: Link prediction queries in KBC are of the form $(s, r, ?)$ with a gold response o^{ev}. Similarly for TKBC they are of the form $(s, r, ?, T)$. The cases of $(?, r, o)$ and $(?, r, o, T)$ are symmetric and receive analogous treatment. Link prediction performance is evaluated by finding the rank of o^{ev} in a list of entities ordered by decreasing score ϕ assigned by the model, and computing MRR. Other measures include the fraction of queries where o^{ev} is recalled within the top 1 or top 10 ranked predictions (HITS@1 and HITS@10).

However, a query may have multiple correct answers. A model must not be penalized for ranking a different *correct* entity over o^{ev}. In KBC this is achieved by filtering out all correct entities above o^{ev} in ranked list and then computing the metrics. In TKBC, filtering requires additional care. Some existing work does no filtering (Dasgupta et al., 2018); others aggressively filter every entity o seen with $(s, r, o, \cdot)$, independent of time (García-Durán et al., 2018; Jin et al., 2019). These lead to systematic underreporting and overreporting a system's performance, respectively. We develop time-aware filtering strategies in Section 3.2.

Time Prediction: Time prediction queries of the form $(s, r, o, ?)$ will require comparing a gold time interval $T^{\text{ev}} = [t_b^{\text{ev}}, t_e^{\text{ev}}]$ with a predicted interval $T^{\text{pr}} = [t_b^{\text{pr}}, t_e^{\text{pr}}]$. Since this is a novel task, existing time prediction metrics do not exist for TKBC. An approach would be to use an adaptation of TAC metric popular in Temporal Slot Filling (Ji et al., 2011; Surdeanu, 2013). TAC[2] adaptation for TKBC will compute a score as $\frac{1}{2}\left[\frac{1}{1+|t_b^{\text{ev}} - t_b^{\text{pr}}|} + \frac{1}{1+|t_e^{\text{ev}} - t_e^{\text{pr}}|}\right]$.

A TAC score is not entirely satisfying for our task. For instance, TAC will assign the same merit score when gold interval [10,20] is compared with predicted interval [5,15], versus when gold [100,200] is compared with prediction [95,195]. However, a human would judge the latter more favorably, because a 5-minute delay in a 10-minute trip would usually be considered more serious than in a 100-minute journey. In response, we investigate alternative evaluation metrics inspired by bounding box evaluation protocols from Computer Vision, in Section 3.1.

3 Evaluation Metrics and Filtering

The preceding discussion motivates why we need clearly-thought-out filtering and evaluation schemes, not only for time prediction queries, but also because time affects link prediction evaluation in subtle but fundamental ways. This section addresses both issues.

[2]TAC's original score compares gold bounds on begin and end of an interval with predicted bounds. The formula provided here is an adaptation, where begin and end are a specific time point each.

Eval query: (s = French National Assembly, r = has member, o =?, $T^{\text{ev}} = [2000, 2003]$)							
Candidates o, system ordered	Known duration of o (any fold)	**Method 1** Unfiltered	**Method 2** Time-insensitive	**Method 3** Time-sensitive			
				2000	2001	2002	2003
Pierre	$[2002, 2003]$	1	0	1	1	0	0
Paul	$[2003, 2008]$	1	0	1	1	1	0
Alain	$[2008, 2009]$	1	0	1	1	1	1
Claude	$[2000, 2003]$	1	0	0	0	0	0
Jean	$[2000, 2007]$	-	-	-	-	-	-
Time-sensitive rank of *Jean*		1+4=5	1+0=1	1+3=4	1+3=4	1+2=3	1+1=2

Table 1: *Jean* is the gold answer. Rows are ranked system predictions. Candidates were also seen in other folds with various intervals that may overlap with T^{ev} in the query. Columns 3–4 show the effective ranks 'lost' by *Jean* to earlier candidates, using earlier filtering methods. Columns 5–8 (Method 3) show the ranks lost for each time point. The bottom row shows ranks of *Jean* as computed by different methods. Time-insensitive filtering over-estimates system performance, while unfiltered evaluation under-estimates it. The final rank assigned by Method 3 to *Jean* is 3.25, which is the average of $\{4, 4, 3, 2\}$, which are the filtered ranks for each time instant in $[2000, 2003]$.

3.1 Time Prediction

One possible way to evaluate time prediction is to adapt measures to compare bounding boxes in computer vision. A possible metric is Intersection Over Union (IOU): $\text{IOU}(T^{\text{ev}}, T^{\text{pr}}) = \text{vol}(T^{\text{ev}} \cap T^{\text{pr}})/\text{vol}(T^{\text{ev}} \cup T^{\text{pr}}) \in [0, 1]$, where vol for our case simply refers to the size of the interval. Unfortunately, IOU loses discrimination once $T^{\text{ev}} \cap T^{\text{pr}} = \varnothing$; e.g., $\text{IOU}([1, 2], [3, 4]) = \text{IOU}([1, 2], [30, 40]) = 0$. This has been noticed recently in computer vision as well, and a metric called gIOU been introduced (Rezatofighi et al., 2019):

$$\text{gIOU}(T^{\text{ev}}, T^{\text{pr}}) = \text{IOU}(T^{\text{ev}}, T^{\text{pr}}) - \frac{\text{vol}((T^{\text{ev}} \Cup T^{\text{pr}}) \setminus (T^{\text{ev}} \cup T^{\text{pr}}))}{\text{vol}(T^{\text{ev}} \Cup T^{\text{pr}})} \in [-1, 1]. \quad (4)$$

$T^{\text{ev}} \Cup T^{\text{pr}}$ is the smallest single contiguous interval (**hull**) containing all of T^{ev} and T^{pr}. E.g., $[1, 2] \Cup [30, 40] = [1, 40]$.

gIOU can be negative, which is not ideal for a performance metric that is aggregated over instances. A simple fix (gIOU′) is to scale it to [0,1] via $(\text{gIOU} + 1)/2$, but we notice that the tiniest overlap between T^{ev} and T^{pr} yields gIOU′ to be at least half, regardless of $\text{vol}(T^{\text{ev}})$ or $\text{vol}(T^{\text{pr}})$. In response, we propose a novel *affinity enhanced IOU*:

$$\text{aeIOU}(T^{\text{ev}}, T^{\text{pr}}) = \frac{\max\{1, \text{vol}(T^{\text{ev}} \cap T^{\text{pr}})\}}{\text{vol}(T^{\text{ev}} \Cup T^{\text{pr}})} \quad (5)$$

When $T^{\text{ev}} \cap T^{\text{pr}} = \varnothing$, the denominator includes "wasted time", reducing aeIOU. As explained in Section 2.2, the '1' in the numerator on the rhs above represents the smallest granularity of time in the data set.

Comparison of Evaluation Metrics: We now compare some properties of the three metrics. First, we list a desirable property of any time-prediction metric (M). It states that if two predicted intervals have intersections of the same size (possibly zero) with the gold interval, then the prediction that has a smaller hull with the gold interval should be scored higher by M. Formally, let T^{pr_1} and T^{pr_2} be two predictions made for T^{ev}.

Property P: Let $\text{vol}(T^{\text{ev}} \cap T^{\text{pr}_1}) = \text{vol}(T^{\text{ev}} \cap T^{\text{pr}_2})$. Then, $M(T^{\text{ev}}, T^{\text{pr}_1}) > M(T^{\text{ev}}, T^{\text{pr}_2})$ if and only if $\text{vol}(T^{\text{ev}} \Cup T^{\text{pr}_1}) < \text{vol}(T^{\text{ev}} \Cup T^{\text{pr}_2})$.

Theorem: IOU and gIOU′ do not satisfy property P, whereas aeIOU satisfies it.

This suggests that aeIOU is a more defensible metric for our task, compared to other alternatives.

3.2 Link Prediction

As we mentioned before, filtering in TKBC link prediction queries offers unique challenges compared to plain KBC. We illustrate these with an example in Table 1. The query asks for the name of person who was a member of the French National Assembly in interval [2000,2003]. Let the gold answer (object) o^{ev} be Jean, which is ranked at the fifth position by the model. All four candidate entities above Jean are seen with the same subject and relation in the training fold, but for different time intervals. E.g., Pierre is also a member of the assembly, but during $[2002, 2003]$. The key question is: how should the four entities above Jean be

filtered to compute its final rank?

Using Table 1, we will argue that existing approaches are unsatisfactory. Dasgupta et al. (2018) underrate model performance by not performing any filtering (Method 1). In this example, the model is penalized for Claude, even though the time-interval for Claude exactly matches the query. On the other hand, García-Durán et al. (2018) and Jin et al. (2019) ignore time information altogether and filter out *all* entities seen with gold (s, r). This can greatly overestimate system quality (Method 2). For instance, the model is not penalized for predicting Alain, even though its membership interval has no overlap with the query interval.

Ideally, filtering must account for the overlap between the query time interval and the time intervals associated with system-proposed entities. We propose such a filtering strategy (Method 3 in Table 1). Method 3 breaks the query into time instants, and computes a filtered rank for each time instant. It associates a rank penalty of 0 with entities known to be exactly correct (Claude) from any fold during the time instant. Entities which not known to have occurred at that time instant induce a full rank penalty of 1. After computing filtered ranks for each time instant, we output the final rank as an average of all such ranks. In this example, this approach will find the average of $\{4, 4, 3, 2\}$, which is 3.25.

Thus, we generalize the notion of rank from a discrete count to a continuous-valued quantity, by aggregating over filtered ranks for each time point in the query interval. This generalized rank is used when computing standard metrics like MRR and HITS@10.

4 The Proposed TIMEPLEX Framework

TIMEPLEX extends the time-agnostic CX model by adding time instants t as first-class objects with their own embeddings, though unlike HyTE it does not bin time instants. The first step in developing a TKBC system is the design of a suitable scoring function $\phi(s, r, o, t)$ and thereby, $\phi(s, r, o, T)$. We first discuss our base proposal for ϕ in Section 4.1. We then present a fully automatic mechanism to introduce additional features to capture recurrent nature of a relation, as well as temporal interactions between pairs of relations. Section 4.2 elaborates on these, along with concomitant enhancements to ϕ. Finally, we specify details of training (Section 4.3) and testing (Section 4.4) protocols.

4.1 TIMEPLEX Base Model

CX assigns score $\Re(\langle \boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}^\star \rangle)$ to triple (s, r, o). The question is how to extend $\phi(s, r, o)$ to $\phi(s, r, o, t)$. Just as a joint distribution is often approximated using lower-order marginals in graphical models (Koller and Friedman, 2009), we construct a base score (ϕ^{TX}) by augmenting CX score with three time dependent terms:

$$\begin{aligned}\phi^{TX}(s, r, o, t) = \\ \Re(\langle \boldsymbol{s}, \boldsymbol{r}^{\text{SO}}, \boldsymbol{o}^\star \rangle) + \alpha\, \Re(\langle \boldsymbol{s}, \boldsymbol{r}^{\text{ST}}, \boldsymbol{t}^\star \rangle) \\ + \beta\, \Re(\langle \boldsymbol{o}, \boldsymbol{r}^{\text{OT}}, \boldsymbol{t}^\star \rangle) + \gamma\, \Re(\langle \boldsymbol{s}, \boldsymbol{o}, \boldsymbol{t}^\star \rangle). \quad (6)\end{aligned}$$

Here, t is a specific time point and not a duration; each s, o and t are embedded as D-dimensional complex vectors, whereas each r is embedded as a concatenation of three sub-vectors $(\boldsymbol{r}^{\text{SO}}, \boldsymbol{r}^{\text{ST}}, \boldsymbol{r}^{\text{OT}})$, and hence requires three times the parameters. $\boldsymbol{r}^{\text{ST}}$ is a relation which is true for entity s at time t (similarly for $\boldsymbol{r}^{\text{SO}}$ and $\boldsymbol{r}^{\text{OT}}$). α, β and γ are hyperparameters.

Jiang et al. (2016a) observed that several relations attach to a subject or object only at specific time points. E.g., subject Barack Obama was president in 2009, regardless of the object United States. In such cases, the approximation above is fully expressive.

To extend from single time instants t to interval T, we propose

$$\phi^{TX}(s, r, o, T) = \textstyle\sum_{t \in T} \phi^{TX}(s, r, o, t). \quad (7)$$

4.2 Relation Recurrence and Pair Scores

We extend TIMEPLEX's base model via additional temporal (soft) constraints that can help in assessing the validity of a tuple better. We aim to capture three types of temporal constraints:

Relation Recurrence: Many relations do not recur (e.g., a person is born only once) for a given entity. Some relations recur with fixed periodicity (e.g., Olympic games recur every four years). Recurrences of other relations may be distributed around a mean time period.

Ordering: A relation precedes another, e.g., *personBornYear* should precede *personDiedYear* for a given subject entity (of type person).

Difference Distribution: The difference in time instants of two relations (wrt to an entity) is distributed around a mean, e.g., *personDiedYear* minus *personBornYear* has a mean of about 70 with some observed variance.

Notice that the first constraint is about a single relation, whereas the latter two concern pairs of relations. Jiang et al. (2016a) attempted to capture relation ordering constraints as a regularizer in the base model. But their approach does not take into account time differences, such as, a person dies roughly 70 years after birth. Nor does it model relation recurrence.

The TIMEPLEX base model may not be able to learn these from data, since each time instant is modeled as a separate embedding with *independent* parameters — it has no explicit understanding of the difference between two time instants. Our preliminary experiments trying to regularize $\|\boldsymbol{t} - \boldsymbol{t}'\|$ for nearby instants t, t' yielded no significant benefits. In response, we augment the model to capture additional features that capture how soon a relation recurs, or how soon after the occurrence of one relation, another relation is likely to follow. We define two scoring functions ϕ^{Rec} and ϕ^{Pair} for these two cases, to be aggregated with ϕ^{TX} (Equation 6).

For simplicity, we model these time gaps as drawn from Gaussian distributions. In what follows, we use $\mathcal{N}(x|\mu, \sigma)$ to denote the probability density of a Gaussian distribution with mean μ and standard deviation σ at the time (difference) value x. We denote as KB^{tr} all tuples in the train fold.

Recurrence Score: We say that (s, r, o) *recurs* if there are at least two distinct intervals T such that $(s, r, o, T) \in \text{KB}^{\text{tr}}$. If there are at least K^{Rec} distinct pairs (s, o) such that (s, r, o) recurs, then relation r is considered *recurrent*. If a relation is not recurrent, we omit them from ϕ^{Rec}. Here K^{Rec} is a hyperparameter.

For each recurrent relation r, our model learns four new parameters: μ_r, σ_r, b_r, and b_r^0. Intuitively, $\mathcal{N}(\cdot|\mu_r, \sigma_r)$ represents a distribution of typical durations between two recurring instances of a relation (with a specific subject and object entity) and b_r and b_r^0 are bias terms.

While computing recurrence features, all training tuples of the form (s, r, o, T) are reduced to (s, r, o, t), i.e., with a singleton time interval, where $t = t_b$, the start time of T. To compute ϕ^{Rec} for (s, r, o, t), we first find the time gap to its closest recurrence, which is

$$\delta = \min_{\{(s,r,o,t') \in \text{KB}^{\text{tr}}:\, t' \neq t\}} |t - t'|. \tag{8}$$

Thereafter we follow these steps:

1. If r is not recurrent, set $\phi^{\text{Rec}} = 0$.
2. If r is recurrent but (s,r,o) does not recur, set a base value $\phi^{\text{Rec}} = b_r^0$.
3. Otherwise, set

$$\phi^{\text{Rec}}(s, r, o, T = [t_b, t_e]) =$$
$$\phi^{\text{Rec}}(s, r, o, t_b) = w_r \mathcal{N}\big(\delta|\mu_r, \sigma_r\big) + b_r. \tag{9}$$

The intuition is that ϕ^{Rec} should penalize the proposed (s, r, o, T) if δ is not close to the μ_r. For example, (Presidential election, held in, USA, 2017) should be penalized as (Presidential election, held in, USA, 2016) is known to be true and the event reoccurs every 4 years ($\mu_r = 4, \sigma_r \approx 0$).

Relation Pairs Score: TIMEPLEX also implements learnable coupling between pairs of relations that might influence each other via soft time constraints. We describe this mechanism for the subject; the object is handled analogously. For each relation pair (r, r'), we maintain four parameters, $\mu_{rr'}$, $\sigma_{rr'}$, $b_{rr'}$ and $w_{rr'}$, whose purpose we will describe presently.

As in the case of recurrence scores, all training tuples of the form (s, r, o, T) are reduced to (s, r, o, t), i.e., with a singleton time interval, where $t = t_b$, the start time of T.

Now, given the candidate tuple (s, r, o, t) to score, we collect fact tuples

$$\{f_i = (s, r_i, o_i, t_i) \in \text{KB}^{\text{tr}}, r_i \neq r\}, \tag{10}$$

having the same subject but a different relation, into the set called $\text{KB}^{\text{Pair}}(s)$. The i^{th} tuple in $\text{KB}^{\text{Pair}}(s)$ is scored as $sc(f_i) = \mathcal{N}(t - t_i|\mu_{rr_i}, \sigma_{rr_i}) + b_{rr_i}$. This represents the contribution of f_i in the validity of candidate tuple, based on their (signed) time difference, and typical time differences observed between these two relations. $\phi^{\text{Pair}}_{\text{sub}}$ needs to aggregate these over f_i. The (trained) parameter $w_{rr'}$ measures how much information the times associated with r' influence our belief in (s, r, o, t). Using these, we define the weighted average

$$\phi^{\text{Pair}}_{\text{sub}}(s, r, o, t) = \sum_{f_i \in \text{KB}^{\text{Pair}}(s)} sc(f_i) \frac{\exp(w_{rr_i})}{\sum_{f_j} \exp(w_{rr_j})}. \tag{11}$$

As with recurrence features, we define $\phi^{\text{Pair}}_{\text{sub}}(s, r, o, T = [t_b, t_e]) = \phi^{\text{Pair}}_{\text{sub}}(s, r, o, t_b)$. A similar $\phi^{\text{Pair}}_{\text{obj}}$ score is computed for the object entity, and the overall $\phi^{\text{Pair}} = \phi^{\text{Pair}}_{\text{sub}} + \phi^{\text{Pair}}_{\text{obj}}$.

Final Score: The final scoring function of TIMEPLEX is

$$\phi(s,r,o,T) = \phi^{TX}(s,r,o,T) + \delta\phi^{\text{Pair}}(s,r,o,T) + \lambda\phi^{\text{Rec}}(s,r,o,T), \quad (12)$$

where δ and λ are model hyperparameters.

4.3 Training

We train TIMEPLEX in a curriculum of two phases. In the first phase, we try to find the best embeddings for all entities, relations and time-instants by minimizing the log-likelihood loss using only the base model TX. We compute the probability of predicting a response o for a query $(s, r, ?, T)$ as:

$$\Pr(o|s,r,T) = \frac{\exp(\phi^{TX}(s,r,o,T))}{\sum_{o'}\exp(\phi^{TX}(s,r,o',T))} \quad (13)$$

Similar terms are computed for $\Pr(s|r,o,T)$ and $\Pr(T|s,r,o)$. The **log-likelihood loss** minimizes:

$$-\sum_{\langle s,r,o,T_+\rangle \in KB^{\text{tr}}} \Big(\log Pr(o|s,r,T_+;\theta) + \log\Pr(s|o,r,T_+;\theta) + \log\Pr(T_+|s,r,o;\theta)\Big) \quad (14)$$

The summation is over all facts known in the training KB, called KB^{tr}. If these are of the form $(s, r, o, [t_b, t_e])$, we convert them into time-instant format by enumerating all $T_+ \in [t_b, t_e]$. To speed up training, we randomly sample a t from this set.

In the second phase, we freeze all embeddings and train the parameters of the recurrence and pairs models. Here, too, we use the log-likelihood loss, except that ϕ^{TX} is replaced by the overall ϕ function. One other minor difference is that $\mu_{rr'}$ and $\sigma_{rr'}$ parameters from the Pairs model are not trained via backpropagation. Instead, they are fitted separately, using the difference distributions for the pair of relations in the training KB. This improves the overall stability of training.

4.4 Inference

At test time, for a link prediction query, TIMEPLEX ranks all entities in decreasing order of $\Pr(o|s,r,T)$ or $\Pr(s|r,o,t)$ scores. For time prediction, its goal is to output a predicted time duration T^{pr}. We first compute a probability distribution over time instants $\Pr(t|s,r,o) = \frac{\exp(\phi(s,r,o,t))}{\sum_{t'\in\mathfrak{T}}\exp(\phi(s,r,o,t'))}$. We then greedily coalesce time instants to output the best duration. For greedy coalescing, we tune a threshold parameter θ_r for each relation r using the dev fold. We then initialize the predicted interval T^{pr} as $\text{argmax}_t \Pr(t|s,r,o)$. Then, as long as total probability of the interval, i.e., $\sum_{t\in T^{\text{pr}}}\Pr(t|s,r,o)$ is less than θ_r, we extend T^{pr} with the instant to its left or right, depending on which has the larger probability.

	YAGO11k	WIKIDATA12k	ICEWS14	ICEWS05-15
Entities	10622	12554	7128	10488
Relations	10	24	230	251
#Instants	308	541	1	11
#Intervals	1814	2232	-	-
Train	16408	32497	72826	368962
Valid	2051	4062	8941	46275
Test	2050	4062	8943	46092

Table 2: Details of datasets used.

5 Experiments

Our experiments assess the performance of TIMEPLEX model using several recent standard TKBC datasets. Our research questions are:

- Does our approach TIMEPLEX convincingly outperform the best time-agnostic and time-aware TKBC systems? Do recurrence and pair features help?
- Are our time embeddings meaningful? Do they capture the passage of time in an interpretable manner?
- Do TIMEPLEX predictions honour intuitive temporal constraints between relations?

5.1 Data sets

We report on experiments with four standard TKBC datasets. WIKIDATA12k and YAGO11k (Dasgupta et al., 2018) are two knowledge graphs with a time interval associated with each fact triple. ICEWS14 and ICEWS05-15 (García-Durán et al., 2018) are two event-based temporal knowledge graphs. The first two datasets contains facts like (David Beckham, plays for, Manchester United; [1992, 2003]). The second group includes several versions of the Integrated Crisis Early Warning System repository. Records here are about political events with timestamps (no nontrivial intervals). We consider the time granularity for interval datasets as 1 year, and for ICEWS datasets as 1 month. See Table 2 for some salient statistics of the datasets. By experimenting across the spectrum, from 'point' events to facts with duration, we wish to ensure the robustness of our observations.

Dataset→	WIKIDATA12k			YAGO11k			ICEWS05-15			ICEWS14		
↓Methods	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10
CX	24.82	14.3	48.9	18.14	11.46	31.11	48.68	37.00	72.63	45.50	33.87	69.73
TA-DM	23.04	12.99	46.07	15.54	9.8	26.74	42.72	30.67	68.01	40.72	29.24	63.87
TA-CX	22.78	12.69	46	15.24	9.36	26.26	49.23	37.6	72.69	40.97	29.58	63.87
HyTE	25.28	14.7	48.26	13.55	3.32	29.81	23.73	3.11	62.76	-	-	-
TIMEPLEX(base)	32.38	22.03	52.79	18.35	10.99	31.86	**63.2**	**54.2**	**80.22**	**58.9**	**49.87**	**76.12**
TIMEPLEX	**33.35**	**22.78**	**53.2**	**23.64**	**16.92**	**36.71**	-	-	-	-	-	-

Table 3: Link prediction performance of time-agnostic and time aware models. The newly-proposed time-sensitive filtering scheme is used. TIMEPLEX(base) means TIMEPLEX without relation pair / recurrent relation features.

5.2 Hyperparameters and policies

We implemented TIMEPLEX using PyTorch. AdaGrad was used for fitting model weights, run for up to 500 epochs for all losses, with early stopping on the dev fold to prevent overfitting. CX, TA-CX and TIMEPLEX use 200-dim complex vectors. We trained HyTE and TA-DM with 400-dimensional real vectors for fair comparison. Only relation embeddings increase to $3\times$ in TIMEPLEX, but this makes little overall difference compared to the much larger number of entity and time embeddings.

Time embeddings are re-normalized to unit L2 length after every epoch. L2 regularization penalty is applied to only those entities, relations and times that are a part of the current batch update, as proposed by Trouillon et al. (2016). Some instances in YAGO11k and WIKIDATA12k datasets may have t_b or t_e missing. We replace missing values by $-\infty$ or $+\infty$ as appropriate for training and link prediction. For evaluating models on time prediction, we filter out such instances from the test set. ICEWS dataset family is event based, duration of each fact is 0. So we set the start time same as end time.

HyTE performs best with margin-based ranking loss as proposed by Dasgupta et al. (2018), whereas all others perform best with logistic loss. We obtain time predictions from HyTE and then get the interval by reusing the greedy coalescing inference method from Section 4.4. For the TIMEPLEX scoring function, α=β=γ=5. For computing the recurrence score, we use K^{rec}=1. We performed a grid search over [1,3,5,7,9] for δ and λ, and found δ=3 and λ=5 to give the best MRR on dev set.

5.3 Results and Observations

Table 3 shows that TIMEPLEX dominates across all data sets. To our surprise, time-agnostic CX baseline performs comparably or better than time-aware baselines (TA-DM and HyTE). The time spans in ICEWS data sets are too short to explore temporal difference features, but TIMEPLEX performs best even without them.

Datasets→	WIKIDATA12k	YAGO11k
↓Methods	aeIOU@1	aeIOU@1
HyTE	5.41	5.14
TIMEPLEX(base)	26.2	14.21
TIMEPLEX	**26.36**	**20.03**

Table 4: Time prediction performance.

Trouillon et al. (2016) established that TransE (Bordes et al., 2013) (used in HyTE) and DistMult (Yang et al., 2015) (used in TA-DM), perform worse than CX. To control for a possibly weaker foundation, we enable the previous baselines with CX as their underlying KB geometry (TA-CX and HyTE-CX). However, CX was not able to lift their accuracy substantially. The performance of HyTE-CX dropped for all datasets, so we exclude it. Time queries are limited to the datasets with nontrivial intervals and time spans and shown in Table 4.

TIMEPLEX has up to 8 point (34%) improvement in link prediction MRR and upto 21 point improvement in aeIoU@1 scores. Note that CX does not have any capability to predict time intervals, and also TA models are not designed to predict time instants or intervals.

5.4 Diagnostics

5.4.1 Time gap vs. embedding distances

Longevity of relations, or gaps between events, are often determined by physical phenomena that are smooth and continuous in nature. Therefore, we expect the embedding of the year 1904 to be closer to that of 1905 compared to the embedding of 1950.

To validate this hypothesis, we compute mean

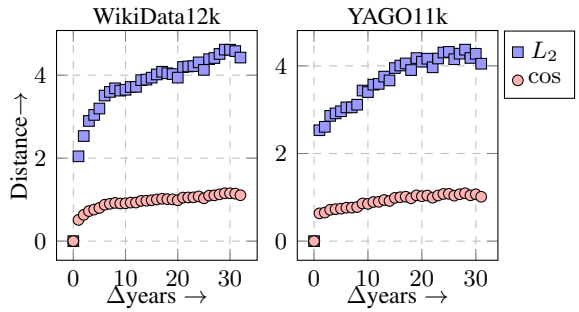

Figure 1: L_2 and cosine distances (y-axis) between time embeddings trained by TIMEPLEX increase with actual distance between time points (x-axis).

L2 distance and mean cosine distance $(1 - \cos)$ of embeddings of time instants which are apart by a given time gap. To account for noise, we drop instant pairs with extreme gaps that have low support (less than 30). For WIKIDATA12k we used embeddings of years $[1984, 2020]$ and for YAGO11k we use embeddings of years $[1958, 2017]$[3].

Figure 1 shows that both L2 and cosine distance between pairs of time embeddings increases with the actual year gap between them. This strongly suggests that the time embeddings learnt by TIMEPLEX naturally represents physical time, even though no temporal smoothing or regularization was attempted.

5.4.2 TimePlex performance across relations

In this experiment, we bucket relations into instantaneous, short-duration, and long-duration based on their mean durations in the train fold. We ignore train facts which have missing times. Relations with duration 1 are termed instantaneous. Relations with duration in $[2, 10]$ are classified as short-duration. Relations with duration > 10 are termed long-duration. On WIKIDATA12k, we report MRR scores of HyTE, CX, and TIMEPLEX on relations in these three buckets, after removing eval fold relations with frequency less than 20.

Long-duration relations include *regionWithinCountry*, and *instanceOf*. Short-duration relations include *significantEvent*, *memberOfSportsTeam*, *educatedAt*, *positionHeld* and *memberOf*. Instantaneous relations include *winner*, *nominatedFor*, *awardReceived*.

TIMEPLEX gives significant improvement across all categories, with a huge jump of 16.6 point (48.3%) MRR for long duration relations.

[3]We do not report results on ICEWS05-15 as it spans only 2005–2015.

	Interval based relation		Instant
	Long Duration	Short Duration	
HyTE	33.7	16.7	38
CX	34.4	15.8	38.4
TIMEPLEX	**51**	**22.6**	**49.8**

Table 5: TIMEPLEX performs better (MRR) on long duration relations as compared to short duration relations on both datasets. On Wikidata12k the model performs well on instant relations.

For instant relations as well an 11.4 point (29.7%) jump is achieved.

5.4.3 Temporal ordering of relation pairs

Both YAGO11k and WIKIDATA12k contain relations with temporal dependencies. E.g., $(\textit{MotherTeresa}, \textit{bornInPlace}, \textit{Macedonia}, 1910)$ and $(\textit{MotherTeresa}, \textit{diedInPlace}, \textit{Kolkata}, 1997)$. For a fixed s, *bornInPlace* obviously always precede *diedInPlace*. Are TKBC models really learning such natural constraints?

We exhaustively checked, for each relation pair (r_1, r_2), whether the existence of both $(s, r_1, \star, t_1)$ and $(s, r_2, \star, t_2)$ was accompanied by $t_1 < t_2$ at least 99% of the time, with a minimum support of 100 entities s. A list of such relation pairs extracted for YAGO11k and WIKIDATA12k is given in the appendix.

For each query in the test set of the form $(?, r, o, t)$, we check whether, for the best s predicted by the model, the query (r, t) violates any temporal ordering constraints when compared with the set of (r, t) pairs seen with s in the training data. For example, for a query *(?, **hasWonPrize**, Nobel Prize, **1925**)*, if a model predicted *Barack Obama* on top, with the fact *(Barack Obama, **wasBornIn**, Hawaii, **1961**)* in the training data, it would violate the ordering constraint *wasBornIn* $\rightarrow$ *hasWonPrize*.

We report the number of such violations in test set, for both time-aware models and our best time-agnostic baseline in Table 6. TIMEPLEX significantly reduces such errors across both datasets, especially in YAGO11k, and this benefit is clearly reflected in its link prediction performance as well.

6 Conclusion

Intense attention on KBC has only just begun to transfer to TKBC. We argue that current evaluation schemes for both link and time prediction have limitations, and propose more meaningful schemes. Surprisingly, a well-tuned CX time-

	YAGO11k	WIKIDATA12k
CX	125	14
HyTE	79	2
TIMEPLEX	**3**	**2**

Table 6: Time constraint violations among top predictions of various models.

ignorant KBC system can perform better than many recent TKBC systems. We propose a new TKBC framework, TIMEPLEX, which combines representations of time with representations of entities and relations. It also has learnt temporal consistency (soft) constraints, which allow other temporal facts in the KB to influence the validity of a given fact. TIMEPLEX exceeds the performance of all baseline and existing TKBC systems. Our experiments suggest that time embeddings are temporally meaningful, and the model makes fewer temporal consistency and ordering mistakes compared to previous models.

Acknowledgements

This work is supported by IBM AI Horizons Network grant, an IBM SUR award, grants by Google, Bloomberg and 1MG, and a Visvesvaraya faculty award by Govt. of India. We thank IIT Delhi HPC facility for compute resources. Soumen Chakrabarti is supported by grants from IBM and Amazon.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS Conference*, pages 2787–2795.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. HyTE: Hyperplane-based temporally aware knowledge graph embedding. In *EMNLP*, pages 2001–2011.

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

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

Alberto Garcia-Duran and Mathias Niepert. 2018. Kblrn: End-to-end learning of knowledge base representations with latent, relational, and numerical features. In *UAI*.

Prachi Jain, Pankaj Kumar, Mausam, and Soumen Chakrabarti. 2018. Type-sensitive knowledge base inference without explicit type supervision. In *ACL Conference*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL Conference*, pages 687–696.

Heng Ji, Ralph Grishman, Hoa Trang Dang, and Kira Griffitt. 2011. Overview of the TAC2011 knowledge base population (KBP) track. In *Text Analysis Conference (TAC)*, pages 14–15.

Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Baobao Chang, Sujian Li, and Zhifang Sui. 2016a. Towards time-aware knowledge graph completion. In *COLING*, pages 1715–1724.

Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Sujian Li, Baobao Chang, and Zhifang Sui. 2016b. Encoding temporal information for time-aware link prediction. In *EMNLP*, pages 2350–2354.

Woojeong Jin, Changlin Zhang, Pedro A. Szekely, and Xiang Ren. 2019. Recurrent event network for reasoning over temporal knowledge graphs. *CoRR*, abs/1904.05530.

Daphne Koller and Nir Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Timothee Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. In *ICML*, pages 2863–2872.

Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016. Holographic embeddings of knowledge graphs. In *AAAI Conference*, pages 1955–1961.

Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, pages 658–666.

Mihai Surdeanu. 2013. Overview of the TAC2013 knowledge base population evaluation: English slot filling and temporal slot filling. In *Text Analysis Conference (TAC)*.

Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. 2017. Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In *ICML*, pages 3462–3471.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080.

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

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

Appendix

A Further training details of TIMEPLEXand HyTE

Each dataset spans along a *time range*, with a certain *time granularity*, which can be year, month or day. TIMEPLEX learns a time embedding for every point in this time range, discretized on the basis of the dataset's granularity (years for the interval datasets WIKIDATA12k and YAGO11k, and days for ICEWS datasets). At training time, TIMEPLEX looks at a single time point at a time - for this, we sample a time point uniformly at random from the query interval $[t_b, t_e]$ associated with the fact. In contrast, HyTE maps each time point to bin (heuristically determined), making the data granularity coarser, and learns representation of these bins. HyTE looks at time points in an interval as well, but enumerates each interval fact to produce a separate fact for each time point beforehand.
Our method of sampling is efficient as the data size is unchanged. It also ensures each fact is sampled uniformly, not hurting link prediction performance by oversampling of long duration facts.

HyTE time prediction: HyTE can only predict a bin for the test fact. To convert predicted bins to years (or days), we take a mean of all years seen with the predicted bin and then do greedy coalescing to output time interval in years.

B Discussion on time evaluation metrics

Here we show that IoU and gIoU do not satisfy a desirable time-metric property P (formally defined in section 3.1).
IoU: This metric gives 0 score to a model, if model's predicted interval does not intersect with the gold, irrespective of the hull. Hence IoU do not satisfy property P.
gIoU: Let us look at the following example. Suppose gold interval is [2002,2005], and consider 2 predictions- [1999,2001] and [1900,2001]. For both predictions, $\text{vol}((T^{\text{ev}} \Cup T^{\text{pr}}) \setminus (T^{\text{ev}} \cup T^{\text{pr}}))$ is zero, so the hull for the two predictions will be ignored (refer to equation (4)), resulting in same scores for both predictions. Hence gIoU does not to satisfy property P.

C Temporal Constraints: Relation Ordering

Table 7 and 8 lists automatically extracted high confidence relation orderings seen in Yago11k and Wikidata12k datasets respectively. These orderings are used to guide TIMEPLEX at the time of training.

graduatedFrom → diedIn
graduatedFrom → hasWonPrize
wasBornIn → graduatedFrom
wasBornIn → diedIn
wasBornIn → isAffiliatedTo
wasBornIn → hasWonPrize
wasBornIn → playsFor
wasBornIn → worksAt
wasBornIn → isMarriedTo
isAffiliatedTo → diedIn
worksAt → diedIn
isMarriedTo → diedIn

Table 7: High confidence (99%) relation orderings extracted from YAGO11k.

educated at → position held
educated at → employer
educated at → member of
educated at → award received
educated at → academic degree
educated at → nominated for
instance of → head of government
residence → award received
academic degree → nominated for
spouse → position held
located in the administrative territorial entity → award received

Table 8: High confidence (99%) relation orderings extracted from WIKIDATA12k

This figure "wikidata-duration.png" is available in "png" format from:

http://arxiv.org/ps/2005.05035v1

This figure "yago-duration.png" is available in "png" format from:

http://arxiv.org/ps/2005.05035v1