Temporal Knowledge Graph Reasoning with Low-rank and Model-agnostic Representations

Ioannis Dikeoulias[1], Saadullah Amin[2], Günter Neumann[1,2]

[1]Department of Computer Science   [2]Department of Language Science and Technology
Saarland Informatics Campus, D3.2, Saarland University, Saarbrücken, Germany
German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany
`{ioannis.dikeoulias,saadullah.amin,guenter.neumann}@dfki.de`

Abstract

Temporal knowledge graph completion (TKGC) has become a popular approach for reasoning over the event and temporal knowledge graphs, targeting the completion of knowledge with accurate but missing information. In this context, tensor decomposition has successfully modeled interactions between entities and relations. Their effectiveness in static knowledge graph completion motivates us to introduce Time-LowFER, a family of parameter-efficient and time-aware extensions of the low-rank tensor factorization model LowFER. Noting several limitations in current approaches to represent time, we propose a cycle-aware time-encoding scheme for time features, which is model-agnostic and offers a more generalized representation of time. We implement our methods in a unified temporal knowledge graph embedding framework, focusing on time-sensitive data processing. The experiments show that our proposed methods perform on par or better than the state-of-the-art semantic matching models on two benchmarks.

1 Introduction

Knowledge graphs offer promising technologies to structure and organize common-sense and domain-specific knowledge and form the information basis for many anticipated technological foundations. Their importance is signified by downstream applications, including speech recognition, sentiment analysis, and knowledge base question answering (Dai et al., 2020). In this context, event and temporal knowledge graphs prove worthy successors of static knowledge graphs, targeting the augmentation of static relational data with temporal meta information. Fig. 1 presents a temporal sub-graph from Wikidata (Vrandečić, 2012), where we are interested in answering the question:
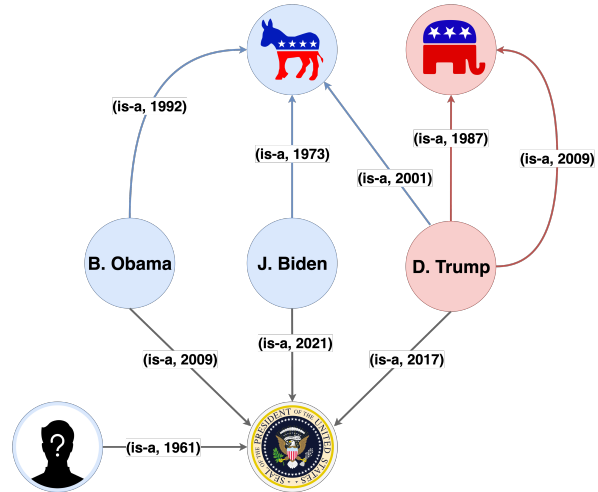
Who was the president of the U.S. in 1961?

Figure 1: Time-sensitive relationships of U.S. presidents with the democratic and republican parties along with their timestamps. Temporal knowledge graph completion (TKGC) aim to perform time-aware reasoning over the KG to answer questions of type *(?, is-a, US_president, 1961)* with an answer *J. F. Kennedy*.

Temporal information opens new opportunities for many time-sensitive domains, including time-series forecasting, biomedical event extraction, and time-sensitive crime reconstruction. However, temporal knowledge graphs show many inconsistencies and lack data quality across different dimensions, including the accuracy, completeness, and timeliness of facts. The quality of evolving knowledge constitutes a challenging task due to the volatile nature of knowledge. To approach the problem of both completeness and correctness in temporal knowledge graphs, this work addresses the task of temporal link prediction and introduces time-aware extensions of the parameter efficient and expressive static embedding model LowFER (Amin et al., 2020). More precisely, we formulate the main contributions of this paper as follows:

- We identify characteristic time-extension themes for extending static knowledge graph embedding models.

- We propose Time-LowFER, a family of time-aware and parameter efficient extensions of LowFER to temporal knowledge graphs.

- We identify limitations in temporal representation learning and propose a time-sensitive encoding scheme based on multi-recurrent cycle-aware time decomposition.

- We introduce a unified time-aware knowledge graph embedding framework focusing on time-sensitive data processing.

2 Related Work

TKGC is a prevalent task in temporal knowledge graph reasoning and targets the incompleteness and timeliness of entailed facts. Formally, the task of TKGC is formulated as: given a temporal knowledge graph $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$ of quadruples (s, p, o, t) where $s, o \in \mathcal{E}$ represent entities, $p \in \mathcal{P}$ represent predicates and $t \in \mathcal{T}$ represent timestamps, the task is to answer either the query $(s, p, ?, t)$ or $(?, p, o, t)$. For the sake of completeness, we refer to s and o also as subject (head) and object (tail) entities, and to p as relation (predicate), also commonly denoted by r.

TKGC approaches are divided into (a) *geometric embedding models* using distance-based scoring functions, (b) *semantic matching models* using similarity-based scoring functions, and (c) *deep learning models*. In this work, we focus on the group of semantic matching models, commonly referred to as factorization-based models. The most prominent static models in this area are DistMult (Yang et al., 2015), SimplE (Kazemi and Poole, 2018), ComplEx (Trouillon et al., 2016) and TuckER (Balazevic et al., 2019). Further, noting characteristic patterns throughout several time-aware extensions of static embedding models, we identified four distinct temporal extension themes Fig. 2(b): (1) inclusion-based, (2) feature-based, (3) regularization-based, and (4) aggregation-based extensions.

2.1 Inclusion-based extensions

Inclusion-based approaches represent extensions, where time features are exposed directly to the underlying embedding model. Time features are considered individual input signals, favoring a more expressive inclusion of time information within the model.

(a) Inclusion   (b) Modulation
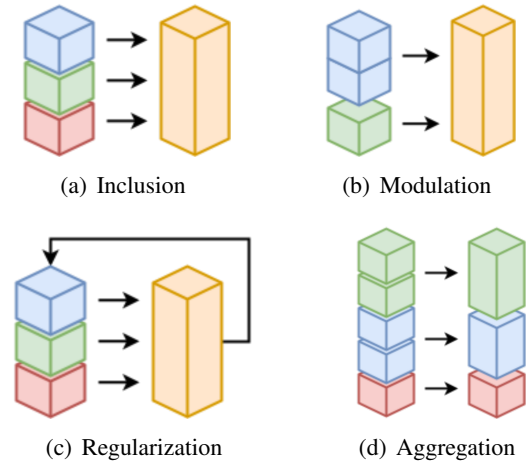
(c) Regularization   (d) Aggregation

Figure 2: Schematic illustration of time extension types: (a) inclusion-based; (b) modulation-based; (c) regularisation-based; and (d) aggregation-based.

A prominent example for inclusion-based time extension is TTransE (Leblay and Chekol, 2018), representing time as temporal translation of entity-relation features. Similarly, TeRo (Xu et al., 2020a) considers individual time features via temporal rotation of entity features. Both models incorporate time as a separate feature and can learn more expressive interactions between input features.

2.2 Modulation-based extensions

Modulation-based extensions are the most commonly used approaches for the time-aware extension of static embedding models. In this context, modulation describes regulating a base signal using a separate modulation signal, i.e., time-based relation modulation allows for a time-sensitive parametrization of relation embeddings.

Most common approaches are TNTComplEx (Lacroix et al., 2020) and TuckERTNT (Shao et al., 2021), which extend their static base models ComplEx (Trouillon et al., 2016) and TuckER (Balazevic et al., 2019) via a temporal modulation of either entity or relation features. Similarly, diachronic embeddings (Goel et al., 2020) represent a model-agnostic and time-aware extension of static embedding models via modulation of entity-specific parameters. In contrast to inclusion-based extensions, modulation-based extensions do not expose time directly to the underlying embedding model. Therefore, modulation is model-agnostic and similarly allows for a parameter-efficient extension of static embedding models.

2.3 Regularization-based extensions

Regularization-based extensions include techniques that impose consistency constraints on learnable feature representations. A commonly used scheme for time-aware regularization is temporal smoothness regularizers, which leverage the semantic nearness of the nearby timestamps.

Both TuckERTNT (Shao et al., 2021), and TNT-ComplEx (Lacroix et al., 2020) implement this type of smoothness regularization scheme. The authors minimize the nuclear p-norm (N3, 3-norm) of the discrete derivative of two nearby time embeddings, effectively penalizing sharp time gradients. Likewise, for TransE-TAE (Jiang et al., 2016) the authors introduce time-wise regularization schemes that enforce constraints for the temporal ordering or disjointness of facts. This way, embedding models are less prone to overfitting and allow for improved generalization to underlying data.

2.4 Aggregation-based extensions

Aggregation-based extensions leverage the compositionality of features such as clustering, grouping, averaging, and sampling.

An application of this idea has been implemented by TeMP (Wu et al., 2020), which introduces time-based subgraph clusters that group information together that occurs within a specific time range. Similarly, ATiSE (Xu et al., 2020b) and TeRo (Xu et al., 2020a) introduce a temporal granularity parameter that varies the temporal sampling rate at which facts are discretized over time.

3 Low-rank Representation

Tensor factorization models decompose the order-3 and order-4 binary tensor into a compressed tensor and a set of factor matrices, respectively, for static and temporal KGs. TuckER (Balazevic et al., 2019) proposed a Tucker decomposition model for static KGs and showed that existing semantic matching models could be subsumed in their formulation. Noting the cubic growth of core tensor in TuckER, LowFER (Amin et al., 2020) proposed an efficient parameter initialization of the core tensor using low-rank factorized bilinear pooling. Due to its ability to handle arbitrary relations (*fully expressive*), parameter efficiency, generalization abilities, and state-of-the-art performance in embedding-based models for static KGs (Zhu et al., 2021), we extend it to the temporal KGs.

3.1 LowFER

LowFER (Amin et al., 2020) introduces a low-rank decomposition of the core tensor in TuckER, reducing the parameter growth from $\mathcal{O}(d^3)$ to $\mathcal{O}(kd^2)$, with d and k being the embedding dimension and factorization rank, respectively. Given subject entity and relation embeddings $\mathbf{e}_s$ and $\mathbf{e}_p$, LowFER approximates the interaction tensor using two low-rank projection matrices $\mathbf{U} \in \mathbb{R}^{d_e \times kd_e}$ and $\mathbf{V} \in \mathbb{R}^{d_r \times kd_e}$. More specifically, both entity and relation features are projected to high-dimensional spaces $\mathbf{U}^T\mathbf{e}_s$ and $\mathbf{V}^T\mathbf{e}_p$, followed by Hadamard product (denoted by $\circ$) and k-sized non-overlapping summation pooling:

$$f(e_s, e_p, e_o) = \langle \mathbf{g}(e_s, e_p), \mathbf{e}_o \rangle$$

where $\langle ., . \rangle$ defines dot product and $\mathbf{g}$ represents a vector-valued function, performing factorized bilinear pooling:

$$g(e_s, e_p) := \text{SumPool}(\mathbf{U}^T\mathbf{e}_s \circ \mathbf{V}^T\mathbf{e}_p, k),$$

and $\mathbf{e}_o$ represents the target entity. LowFER can generalize the TuckER model. Moreover, for $k \leq \min(d_e, d_r)$ with d_e, d_r as entity and relation dimensions, respectively, LowFER is able to accurately represent TuckER's core tensor $\mathcal{W}_{\text{tucker}}$. In addition, given the subsumption of TuckER, LowFER is equally fully expressive and thus able to represent arbitrary relations, e.g., symmetric, reflexive, and transitive, among others.

3.2 Time-LowFER

This section introduces Time-LowFER, a family of time-aware extensions of the bilinear embedding model LowFER. These include (i) LowFER-TNT: a modulation-based extension following time-relation modulation (Lacroix et al., 2020), (ii) LowFER-CFB: an inclusion-based extension using chained bilinear pooling, and (iii) LowFER-FTP: a reduced variant of the latter, with factorized trilinear pooling.

3.3 Factorized Bilinear Pooling

Following existing works of Lacroix et al. (2020); Shao et al. (2021), we propose two variants of time modulation also referred to as time modulation (T) and time-no-time modulation (TNT), which extends the factorized bilinear pooling of LowFER. The first variant (T) performs a simple temporal

modulation of relation features e_r using time features e_t. We refer to this extension as LowFER-T and formulate its scoring function as follows:

$$f(e_s, e_p, e_o, e_t) = \langle \mathbf{g}(e_s, e_p, e_t), \mathbf{e}_o \rangle$$

where e_t is time embedding and $\mathbf{g}$ is defined as:

$$g(e_s, e_p, e_t) = \mathbf{U}^T \mathbf{e}_s \circ \mathbf{V}^T (\mathbf{e}_p \odot \mathbf{e}_t)$$

Time modulation (T), denoted by $\odot$, enables LowFER-T to learn joint time-aware representations of entity and relation features such that learned feature-to-feature interactions now incorporate the dynamics of the overlying knowledge graph with more precise predictions.

However, not all predicates are similarly affected by time or show reduced sensitivity to temporal changes in related facts. For instance, the predicate *born_in* is not changing over time, however the predicate *works_at* (most probably) will. To capture both dynamic and static characteristics of temporal relations, we follow Lacroix et al. (2020) and propose a time-no-time (TNT) variant of LowFER, which calculates a combined representation of static and dynamic relations. We refer to this extension as LowFER-TNT and formulate the function $\mathbf{g}$ as:

$$g(e_s, e_p, e_t) = \mathbf{U}^T \mathbf{e}_s \circ \mathbf{V}^T (\mathbf{e}_p^t \odot \mathbf{e}_t + \mathbf{e}_p)$$

e_p^t denotes the time-aware relation embedding and e_p the static relation feature. Both T and TNT variants of LowFER do not modify the assumptions of the underlying methods, i.e., the approximation of TuckER's core interaction tensor. Therefore, both modulation-based extensions are fully expressive and can be seen as (time-aware) generalizations of LowFER, while equally, TNT subsumes T.

3.4 Chained Factorized Bilinear Pooling

Time features encode latent dynamics of evolving knowledge graphs and allow for a time-aware classification of relational links, i.e., time features set crucial constraints on feature interactions, similarly disqualifying the existence of specific graph structures for a given time range. For instance, after *(U.S.) presidential elections*, the link *is_president* will remain static for at least four years. Similarly, the *signing* of international climate agreements would imply a change of *demands* for government and industry. However, bilinear models (Section 3.3) are solely defined over two variables,

making them less suitable for multivariate analysis, e.g., entity, relation, and time features. We propose a multilinear method based on bilinear chaining for use in multivariate learning to overcome these limitations.

More precisely, we define a k-fold chaining of bilinear models through the nesting of bilinear transformations:

$$\begin{aligned} B_k^C(x_1, \ldots, x_{k+1}) \\ = (B_1 \circ \cdots \circ B_k)(x_1, \ldots, x_{k+1}) \\ = B_k(B_{k-1}(\ldots, x_k), x_{k+1}), \end{aligned}$$

where $B_1, \ldots, B_k$ denote bilinear transformations, $(x_1, \ldots, x_{k+1})$ denote the input features and the $\circ$ here represents the function composition operator. Following LowFER, we introduce a time-aware extension of LowFER based on two-fold chaining of factorized bilinear methods.

$$g(e_s, e_p, e_t) := \text{SumPool}((B_1 \circ B_2)(e_s, e_p, e_t), k)$$

where B_1 and B_2 denote two nested bilinear transformations. Similarly, the above equation can be rewritten in terms of three low-rank projection matrices, $\mathbf{U}, \mathbf{V}$ as similar to LowFER and $\mathbf{Q} \in \mathbb{R}^{dr \times kd_e}$ as:

$$\begin{aligned} g(e_s, e_p, e_t) \\ = \text{SumPool}(\mathbf{U}^T \mathbf{e}_s \circ \mathbf{R}^T (\mathbf{V}^T \mathbf{e}_p \circ \mathbf{Q}^T \mathbf{e}_t), k) \end{aligned}$$

We refer to this method as chained factorized bilinear (CFB) pooling. CFB learns two joint representations between relation and time features and once between entities and the joint representation of time-relation features. However, we note that the intermediate projection matrix $\mathbf{R} \in \mathbb{R}^{kd_e \times kd_e}$ is likely to share redundant parameters with both relation and time projection matrices $\mathbf{V}$ and $\mathbf{Q}$.

3.5 Factorized Trilinear Pooling

CFB enables the computation of fine-grained interactions between different feature spaces. However, CFB introduces redundant parameters via the intermediate projection of joint time-relation features. To retain the efficiency of the original low-rank bilinear method, we introduce a (reduced) specialization of the CFB, which omits the intermediate feature projection. Replacing $\mathbf{R}$ by an identity matrix $\mathbf{I}$, we formulate the respective method via a three-way Hadamard product or entity, relation,

and time features, followed by summation pooling:

$$g(e_s, e_p, e_t)$$
$$= \text{SumPool}(\mathbf{U}^T \mathbf{e}_s \circ \mathbf{V}^T \mathbf{e}_p \circ \mathbf{Q}^T \mathbf{e}_t, 1),$$

Where $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{Q}$ are the low-rank projection matrices, note the factorization rank, k is set to 1.

4 Cycle-aware Time Embedding

In this section, we propose a novel extension technique for embedding time features ($\mathbf{e}_t$), which relies on multi-recurrent cycle-aware (MRCA) time decomposition and is model-agnostic. We first explain the concept of *multi-recurrence* and then show its application for *cycle-aware encoding* of time features.

Temporal recurrence denotes the concept of expressing time as recurrent component within a certain time frame e.g., a *week* occurs approximately four times per *month* or equally a *year* contains four *seasons*. In this context, we speak of *multi-recurrence*, if a time frame is expressed in terms of multiple cycles, e.g., a *year* entails four *seasons*, 12 *months*, *52* weeks and *365* days. Multi-recurrence is expressible for all time concepts with one or more underlying recurrent cycles. Even more, (long-term) cycles themselves are also expressible in terms of more fine-grained (short-term) cycles.

Our multi-recurrent cycle-aware (MRCA) encoding uses a mapping ψ^{cyc} of timestamps T to a set of recurrence encodings $C^{m \times l}$:

$$\psi^{cyc} : T \to C^{m \times l}$$
$$t_i \mapsto (\phi_{W_1}^{rec}(t_i), \dots, \phi_{W_m}^{rec}(t_i)), \ 1 \leq i \leq n,$$

where each recurrence encoding $\phi_{W_k}^{rec}$ defined upon cycle window W_k, uses a mapping of timestamps T to a set of cycle indices C^l:

$$\phi_{W_k}^{rec} : T \to C^l$$
$$t_i \mapsto (c_1^k, \dots, c_l^k), \ 1 \leq k \leq m,$$

and each time component c_p^k with $1 \leq p \leq l$ is defined as:

$$c_p^k = \begin{cases} v & \text{if } \textit{p-th} \text{ subcycle exists in } W_k \\ 0 & \text{else} \end{cases}.$$

Here, n denotes the number of timestamps, m is the number of time windows, and l is the number of recurrent subcycles. Now, to generate the cycle-aware multi-recurrent time embedding, we consider
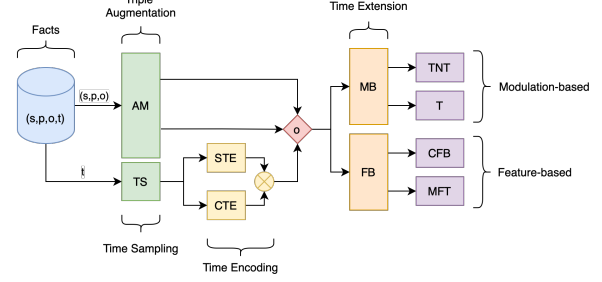

Figure 3: Overview of data processing and modeling pipeline in the CHRONOKGE framework. A quadruple fact is processed using triple augmentation and time sampling (*green*), and afterward, timestamps are decomposed into subcomponents (*yellow*). Then, the processed quadruple is passed to the respective temporal extension (*orange*) and model variation (*violet*).

the following five components and their respective cycle decompositions[1]:

$$\phi_W^{rec}(t_i) \qquad \to (c_d^W)$$
$$\phi_M^{rec}(t_i) \qquad \to (c_d^M, c_w^M)$$
$$\phi_S^{rec}(t_i) \qquad \to (c_d^S, c_w^S, c_m^S)$$
$$\phi_Y^{rec}(t_i) \qquad \to (c_d^Y, c_w^Y, c_m^Y, c_s^Y)$$
$$\phi_G^{rec}(t_i) \qquad \to (c_1^G, c_{10}^G, c_{100}^G, c_{1000}^G)$$

Where W, M, S, Y, G denote the weekly, monthly, seasonal, yearly, and global components, respectively, and d, w, m, s, y denote the daily, weekly, monthly, seasonal, and yearly subcycles. The components $c_1, c_{10}, c_{100}, c_{1000}$ denote the positional year component for single years, decades, centuries and milleniums, respectively. The final cycle-aware time encoding ψ^{MRCA} is then generated by summing over all cycle decompositions of each separate recurrence mapping $\phi_{W_k}^{rec}$:

$$\mathbf{e}_t = \psi^{MRCA}(t) = \sum_{i=1}^{l_1} (\phi_G^{rec}(t))_i + \sum_{i=1}^{l_2} (\phi_Y^{rec}(t))_i$$
$$+ \sum_{i=1}^{l_3} (\phi_S^{rec}(t))_i + \sum_{i=1}^{l_4} (\phi_M^{rec}(t))_i$$
$$+ \sum_{i=1}^{l_5} (\phi_W^{rec}(t))_i$$

---

[1]The generation of recurrent time cycles rely on prior knowledge of empirically defined periods, e.g., a week has seven days, a month has 30 days, a year consists of approximately 365 days, e.t.c.

	#E	#R	#T	time span	gran.	MD.	TQ.	MC.
ICEWS14	7,129	230	365	2014	daily	✗	✗	✗
ICEWS05-15	10,488	251	4,017	2005-2015	daily	✗	✗	✗

Table 1: Data statistics and properties: No. entities (E), no. relations (R), no. timestamps (T), time span, time granularity, multiple domains (MD), temporal qualifier (TQ), and manual curation (MC).

5 Experiments and Results

5.1 Data

ICEWS (Integrated Crisis Early Warning System) was founded in 2008 as a DARPA program and is currently maintained by Lockheed Martin. The conflict warning system collects news about political events from different digital and social media platforms and stores the extracted information in the associated ICEWS database.

ICEWS14 is a subset of the ICEWS database, including facts from the year 2014. It consists of 7,129 distinct entities, 230 relations, and 365 timestamps with 24 hours (daily) temporal granularity.

ICEWS05-15 is another subset of the ICEWS database, including facts between the start of 2005 and the end of 2015. It consists of 10,488 distinct entities, 251 relations, and 4,017 timestamps with a temporal granularity of 24 hours (daily). In Table 1 we provide an overview of the datasets.

5.2 Implementation

For implementation, we developed an extensible temporal knowledge representation learning framework CHRONOKGE[2]. Fig. 3 shows the data and modeling pipeline in the framework. We present more details in Appendix A.1.

For training, we use the Adam (Kingma and Ba, 2015) optimizer with a learning rate of 0.01 and a decay rate of 0.99. We perform 1-N scoring with binary cross-entropy loss and choose a batch size of 1000. We further apply a label smoothing of 0.01 to the target labels. The embedding dimension of entities, relations, and time is set to 300. We use $k = 32$ in LowFER-TNT and LowFER-CFB, and use dropout following Amin et al. (2020).

5.3 Baselines

We choose the static LowFER model for our experiments to apply to our proposed temporal extensions. Since LowFER is a semantic matching linear model, we only compare our results with the extensions of such linear models. In particular, we use

---

2 https://github.com/iodike/ChronoKGE

as baselines the time-aware extension of both ComplEx (TComplEx, TNTComplEx) (Lacroix et al., 2020), TuckER (TuckERT, TuckERTNT) (Shao et al., 2021) as well as SimplE (DE-SimplE) and DistMult (DE-DistMult) (Goel et al., 2020).

5.4 Time and No-Time Modulation

Following the findings of Shao et al. (2021) and Lacroix et al. (2020), we only modulate relations with time information (instead of entities). For both ICEWS datasets, as depicted in Table 2, we see persistent improvements in the TNT-extension over the T-extension. It shall be noted that both TuckERTNT and TNTComplEx use time and embedding regularization schemes, where Time-LowFER extensions are reported without any regularization that can further improve the results.

While the T-extension learns a time-aware relation that primarily relies on temporal information, the TNT extends it by learning an additional (static) relation embedding as shown in Fig. 4. This effect is beneficial for highly frequent relations (light areas), which should not rely too strictly on time. In contrast, temporal information is much more valuable for relations that occur less frequently (dark areas) and should be incorporated with a higher weighting.

### 5.5 Chained and Factorized Bilinear Pooling

Now we evaluate Chained Factorized Bilinear (CFB) pooling and Factorized Trilinear (FTP) pooling. As shown in Table 2, the LowFER-CFB outperforms all baselines, including the T and TNT extensions of LowFER. This can be attributed to the additional expressive modeling capacity of LowFER-CFB with multi-layered bilinear interactions. CFB learns more accurate feature fusion between all three input modalities. However, the inclusion of intermediate feature projections favors the redundancy of captured feature interactions. Therefore, while the CFB offers improved results (w.r.t. MRR) over the FTP, it is more vulnerable to overfitting.

	ICEWS14				ICEWS05-15			
	MRR	H@10	H@3	H@1	MRR	H@10	H@3	H@1
DE-DistMult◇	0.501	0.708	0.569	0.392	0.484	0.718	0.546	0.366
DE-SimplE◇	0.526	0.725	0.592	0.418	0.513	0.748	0.578	0.392
TComplEx†	0.560	0.730	0.610	0.470	0.580	0.760	0.640	0.490
TNTComplEx†	0.560	0.740	0.610	0.460	0.600	0.780	0.650	0.500
TuckERT‡	0.594	0.731	0.640	0.518	<u>0.627</u>	0.769	0.674	0.550
TuckERTNT‡	0.604	<u>0.753</u>	0.655	0.521	**0.638**	0.783	<u>0.686</u>	**0.559**
LowFER-T	0.584	0.734	0.630	0.505	0.559	0.714	0.605	0.476
LowFER-TNT	0.586	0.735	0.632	0.507	0.562	0.717	0.608	0.480
LowFER-CFB	**0.623**	0.757	**0.671**	**0.549**	**0.638**	<u>0.791</u>	**0.690**	<u>0.555</u>
LowFER-FTP	<u>0.617</u>	**0.765**	<u>0.665</u>	<u>0.537</u>	0.625	**0.792**	0.681	0.534

Table 2: Time-LowFER performance on ICEWS datasets. ◇ Results taken from (Goel et al., 2020). † Results taken from (Lacroix et al., 2020). ‡ Results taken from (Shao et al., 2021).
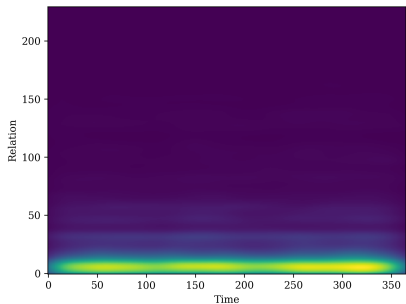
(a) ICEWS14
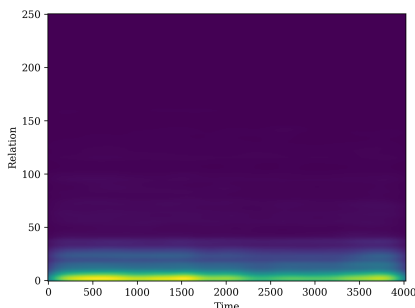
(b) ICEWS05-15

Figure 4: Time-relation heatmap: (a) for ICEWS14; and (b) for ICEWS05-15.

5.6 Simple and Cycle-aware Time Encoding

In this experiment, we evaluate two approaches for timestamp encoding, Simple Time Encoding (STE), which performs a bijective projection of timestamps to natural numbers, and Cyclical Time Encoding (CTE), which relies on multi-recurrent cycle-aware time decomposition (MRCA). CTE targets specific limitations in representation learning of absolute timestamps, such as its inability to learn shared representations across different timestamps. By introducing cyclical time components, time features to benefit from an improved sharing of parameters within individual embedding subspaces and allow for an increased generalization of short-term events. Technically, CTE reduces the multi-collinearity of low-latent time features and allows for an improved semantic separability across individual representations.

In CTE, we focus on high recall and therefore concentrate primarily on the Hits@10 metric in our evaluation. Detailed results are provided in Table 3. The MRCA-algorithm uses predefined time cycles, which are divided into 10 short-term (in-year) cycles and four long-term (multi-year) cycles, particularly favouring the dense distribution of time information for ICEWS datasets (see Figure 4(a), 4(b)). As a consequence, both ICEWS datasets show increased results (w.r.t Hits@10) with an increase of 3.4% (ICEWS14) and 5.7% (ICEWS05-15) for modulation-based extensions as well as 1.4% (ICEWS14) for feature-based extensions.
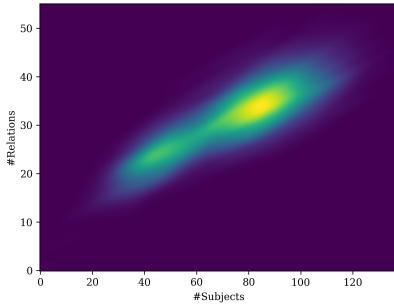
5.7 Time Sampling Rate

In this section, we investigate the effect of time sampling in TKGC. Following the approach of Xu et al. (2020a) and Xu et al. (2020b), we include a time granularity parameter, allowing to sample timestamps from a given dataset at different sampling rates. In particular, we investigate sampling rates in the power of two $\{1.0, 2.0, \ldots, 1024.0\}$,

		ICEWS14				ICEWS05-15			
		MRR	H@10	H@3	H@1	MRR	H@10	H@3	H@1
STE	LowFER-T	0.584	0.734	0.630	0.505	0.559	0.714	0.605	0.476
	LowFER-TNT	0.586	0.735	0.632	0.507	**0.562**	0.717	0.608	**0.480**
CTE	LowFER-T	**0.600**	0.764	**0.654**	**0.511**	0.556	**0.771**	**0.621**	0.442
	LowFER-TNT	0.583	**0.769**	0.640	0.485	0.549	0.767	0.614	0.434

Table 3: STE and CTE results for T/TNT-extensions on ICEWS datasets.
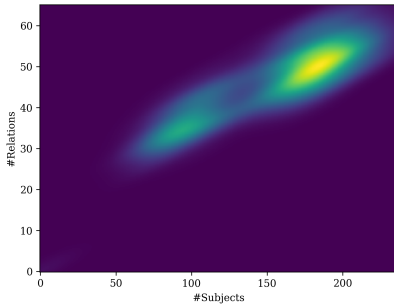
(a) ICEWS14

(b) ICEWS05-15

Figure 5: Time concentration heatmap: (a) for ICEWS14; and (b) for ICEWS05-15.

where 1.0 represents the initial sampling rate, in which timestamps are discretized w.r.t to time granularity.

Sampling timestamps at lower rates cause the underlying temporal KG to aggregate facts into smaller time clusters, up to the extreme case where all facts are linked only to one timestamp. In other words, with increasing time sampling rates, a temporal knowledge graph is synthetically transformed into a static KG. While time sampling does not represent a viable extension for evaluating time-sensitive embedding models, it allows examining the significance of temporal facts for individual benchmark datasets.
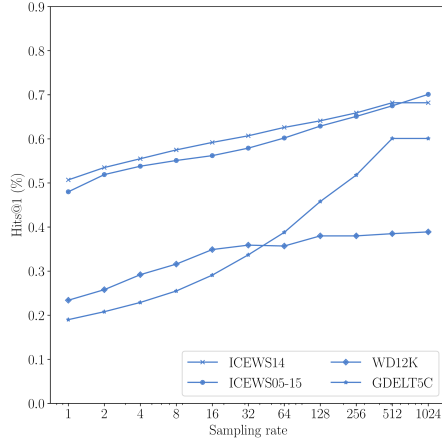

Figure 6: Influence of increasing time sampling rates.

6 Conclusion

In this work, we introduced Time-LowFER, a family of time-aware extensions of the bilinear factorization model LowFER. Following existing work in temporal link prediction, we extended LowFER using time-modulated relations (TNT). Further, noting several limitations of modulation-based extensions, we proposed two feature-based extensions of LowFER, which are based on bilinear chaining (CFB) and trilinear fusion (FTP). In particular, we showed that the FTP represents a parameter-efficient specialization of the CFB, while CFB offers state-of-the-art results among semantic matching models for temporal link prediction.

In addition, we investigated four different approaches for time-aware extension of static embedding models and outlined, despite the increased popularity of time modulation techniques, the superiority of feature-based KGE extensions. Furthermore, we investigated the process of time encoding in representation learning and proposed a model-agnostic method (CTE) for encoding timestamps based on multi-recurrent cycle-aware (MRCA) time decomposition.

References

Saadullah Amin, Stalin Varanasi, Katherine Ann Dunfield, and Günter Neumann. 2020. Lowfer: Low-rank bilinear pooling for link prediction. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 257–268. PMLR.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5185–5194, Hong Kong, China. Association for Computational Linguistics.

Yuanfei Dai, Shiping Wang, Neal N Xiong, and Wenzhong Guo. 2020. A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics*, 9(5):750.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3988–3995. AAAI Press.

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

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4289–4300.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion Proceedings of the The Web Conference 2018*, pages 1771–1776.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Pengpeng Shao, Dawei Zhang, Guohua Yang, Jianhua Tao, Feihu Che, and Tong Liu. 2021. Tucker decomposition-based temporal knowledge graph completion. *Knowledge-Based Systems*, page 107841.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Denny Vrandečić. 2012. Wikidata: A new platform for collaborative data collection. In *Proceedings of the 21st international conference on world wide web*, pages 1063–1064.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L. Hamilton. 2020. TeMP: Temporal message passing for temporal knowledge graph completion. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5730–5746, Online. Association for Computational Linguistics.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020a. TeRo: A time-aware knowledge graph embedding via temporal rotation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1583–1593, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Chenjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Yazdi, and Jens Lehmann. 2020b. Temporal knowledge graph completion based on time series gaussian embedding. In *International Semantic Web Conference*, pages 654–671. Springer.

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 *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. 2021. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in Neural Information Processing Systems*, 34.

A Experimental Setup and Hyperparameters

The experiments of our work are conducted on a *SLURM* computing cluster. The virtual environments are initiated over a Unix-based system running *Ubuntu* 18.04.5 (Bionic Beaver) with kernel version 5.4.0-80-generic. Each job uses a single RTX3090 GPU (Ampere) with 24GB of shared memory and 8 CPUs. All experiments are built using the machine learning framework *PyTorch* at version 1.9.0 and NVIDIA's graphics programming interface *CUDA* with toolkit version 11.1.

In addition, we performed HPT using Optuna's hyper-parameter optimization framework. We configured the tuner to perform combined (relative/independent) sampling and used a median pruner with a warm-up threshold of 10% and set startup trials to 10. Further, we activated early stopping and set a maximum timeout of 24h. For fine-tuning, we set learning rate $\in \{0.1, 0.01, 0.001, 0.0001\}$, decay rate $\in \{0.1, 0.01, 0.001, 0.0001\}$, batch size $\in \{128, 256, 512, 1024\}$ and label smoothing $\in \{0.1, 0.01, 0.001\}$. We selected the best parameters for the final experiments and set the batch size to 1024.

B ChronoKGE Framework

CHRONOKGE is a unified graph embedding framework for the development of time-aware knowledge graph completion models. It is implemented in Python and builds upon PyTorch's (Paszke et al., 2019). Our framework focuses on time-sensitive representation learning tasks for temporal and event knowledge graphs and offers an easy-to-use and flexible library with various time-focused functionalities, including time-specific sampling and encoding routines.

CHRONOKGE supports multiple temporal knowledge graphs with diverse graph schemas and offers a dynamic interface for adding new knowledge graphs. Similarly, our framework provides
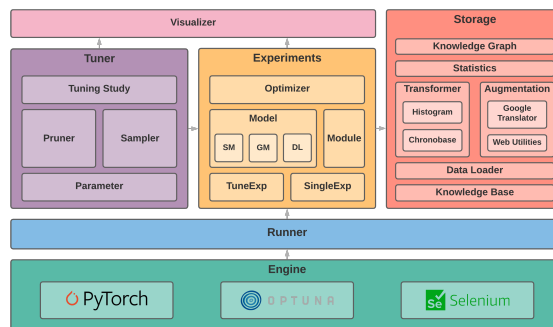

Figure 7: Overview of the ChronoKGE framework.

an easy interface to add new or extend existing learning models. Therefore, several generic embedding models are already available within the `model.kge` package, which extend PyTorch's default `nn.Module` by integrating commonly required methods in knowledge representation learning. In addition, it provides a customizable and flexible package for defining experimental jobs as well as additional modules for training. To support hyper-parameter optimization, we integrated a parameter tuning system which is based on the *Optuna*[3] framework. The tuning system is part of the integrated `chrono_kge.tuner` package and allows for an automatic search of optimal hyper-parameters.

C Limitations

In its current form, our proposed methods can overfit since they lack commonly used regularization schemes, such as time-smoothness and nuclear 3-norm (Lacroix et al., 2020; Shao et al., 2021). However, extending our work with regularization schemes is straightforward. In terms of the design choices, LowFER-CFB offers a more expressive representation. However, it is computationally more expensive (7.85 sec/epoch for LowFER-CFB compared to 4.19 sec/epoch for LowFER-TNT on ICEWS-14) and MRCA, despite offering a generalized time representation, has limited performance gains and further adds a computational footprint (2.19 additional seconds per epoch on ICEWS-14 for LowFER-TNT).

---

[3]https://optuna.org