Enhancing neural link predictors for temporal knowledge graphs with temporal regularisers

Manuel Dileo[1] Pasquale Minervini[2,3] Matteo Zignani[1]
Sabrina Gaito[1]

[1] Department of Computer Science, University of Milan, Italy
[2] School of Informatics, University of Edinburgh, UK [3] Miniml.AI, UK
manuel.dileo@unimi.it

Abstract. The problem of link prediction in temporal knowledge graphs (TKGs) consists of finding missing links in the knowledge base under temporal constraints. Recently, [4] and [8] proposed a solution to the problem inspired by the canonical decomposition of 4-order tensors, where they regularise the representations of time steps by learning similar transformation for adjacent timestamps. However, the impact of the choice of temporal regularisation terms is still poorly understood. In this work, we systematically analyse several choices of temporal regularisers using linear functions and recurrent architectures. In our experiments, we show that by carefully selecting the temporal regulariser and regularisation weight, a simple method like TNTComplEx [4] can produce comparable results with state-of-the-art methods and enhance its original performance. Specifically, we observe that linear regularisers for temporal smoothing based on specific nuclear norms can significantly improve the predictive accuracy of the base temporal link prediction methods.

1 Introduction

Temporal Knowledge Graphs (TKGs) [10] are graph-structured knowledge bases where knowledge about the world is encoded in the form of relationships of various kinds between entities, provided with their timestamps, indicating when relations happened. When reasoning with TKGs, one of the most crucial tasks is finding or completing missing links in the temporal knowledge graph at a precise time point, often referred to as the *TKG completion task*. Recently, [4] and [8] proposed two state-of-the-art approaches by extending factorisation-based neural link prediction models with dense representations for each timestamp and regularising such representations by enforcing them to change slowly over time. However, the impact of the choice of temporal regularisation terms is still not well understood.

Hence, in this work, we systematically analyse a comprehensive array of temporal regularisers for neural link predictors based on tensor factorisation. Starting from the temporal smoothing regulariser proposed by [4], we also consider a broad class of norms in the L_p and N_p families, which allow us to control the strength of the smoothing. We further extend the set of temporal regularisers by considering the regulariser proposed by [8], defining it as an explicit modeling of the temporal dynamic between adjacent timestamps. Lastly, we also adopt a recurrent architecture as an implicit temporal regulariser that can generate

timestamp embeddings sequentially and learn the temporal dynamic during the training phase.

By conducting an experimental evaluation over three well-known benchmark datasets (ICEWS14, ICEWS05-15, and YAGO15K) we show that using our proposed temporal regularisers, neural link predictors based on tensor factorisation models can significantly improve their predictive accuracy in temporal link prediction tasks. By carefully selecting a temporal regulariser and regularisation weight, our version of TNTComplEx [4] produces comparable results with other competitors, such as methods based on temporal message passing [7] or convolutional neural networks [3]. Overall, linear regularizers for temporal smoothing that introduce smaller loss penalties for closer timestamp representations achieve the best performance. In contrast, recurrent architecture struggles to generate a long sequence of timestamps. In general, our work shows that by carefully tuning simple tensor factorisation models, they can reach comparable performance with other competitors, enabling several applications useful in network science and graph mining, such as adding logical constraints, producing more interpretable results, or scaling to very large graphs [6], without substantial sacrifices in performance.

Background. A Temporal Knowledge Graph (TKG) is referred to as a set of quadruples $\mathcal{K} \subseteq \{(s, p, o, \tau) \mid s, o \in \mathcal{E}, p \in \mathcal{R}, \tau \in \mathcal{T}\}$. Each quadruple represents a true temporal fact. $\mathcal{E}$ is the set of all entities and $\mathcal{R}$ is the set of all relations in the ontology. The fourth element in each quadruple represents time, often discretised. $\mathcal{T}$ represents the set of all possible timestamps. Temporal Knowledge Graph Completion refers to the problem of completing a TKG by inferring facts from a given subset of its facts. The subject of temporal link prediction has been studied using a wide range of approaches. A recent survey [10] gives a general overview of temporal link prediction models for TKGs. In this work, we focus on methods that learn the temporal behaviour by using a representation of time. For instance, [4] performs tensor decomposition based on the time representation, while [8] performs 4th-order tensor factorization using a linear temporal regularizer and multivector embeddings. Hence, we systematically analyse a wide array of temporal regularisers to understand their impact on both performance and learning temporal behaviour.

2 Temporal KG Representation Learning

This section presents a framework for temporal knowledge graph representation learning [4]. Given a TKG, we want to learn representations for entities, relations, and timestamps (e.g., $\mathbf{s}, \mathbf{p}, \mathbf{o}, \mathbf{t}_\tau \in \mathbb{R}^{\mathbf{d}}$) and a scoring function $\phi_\theta(s, p, o, \tau) \in \mathbb{R}$, such that true quadruples receive high scores. Thus, given ϕ_θ, the embeddings can be learned by optimising an appropriate cost function. As in [5], we minimise, for each of the train tuples (s, p, o, τ), the instantaneous multi-class loss:

$$\ell(\phi_\theta; (s, p, o, \tau)) = -\phi_\theta(s, p, o, \tau) + \log \left(\sum_{o' \in \mathcal{E}} \exp \left(\phi_\theta(s, p, o', \tau) \right) \right), \qquad (1)$$

This loss is only suited to queries of the type (subject, predicate, ?, time),i.e. the queries considered in related work. For a training set S (augmented with reciprocal relations [5]), and parametric tensor estimate ϕ_θ, we minimize the following objective, with an embedding regulariser Ω:

$$\mathcal{L}(\phi_\theta) = \frac{1}{|S|} \sum_{(s,p,o,\tau)\in S} \left[\ell(\phi_\theta; (s,p,o,\tau)) + \lambda\Omega(\theta; (s,p,o,\tau))\right]. \tag{2}$$

In our experiments, embedding regularisation is performed using a nuclear tensor 3-norm [5].

TNTComplEx. TNTComplEx [4] extends the ComplEx [5] decomposition to the TKG completion task by adding a factor T resulting in the scoring function ϕ_θ:

$$\phi_\theta^{\text{TComplEx}}(s,p,o,\tau) = \text{Re}\left(\langle \mathbf{s}, \mathbf{p}, \overline{\mathbf{o}}, \mathbf{t}_\tau \rangle\right) \tag{3}$$

where $\mathbf{s}$ and $\mathbf{o}$ are the embeddings of entities $s, o \in \mathcal{E}$, $\mathbf{p}$ is the embedding for the relation $p \in \mathcal{R}$, and $\mathbf{t}_\tau$ is the embedding for the timestamp $\tau \in \mathcal{T}$. Intuitively, timestamp embedding modulates the multi-linear dot product.

In TKG, where only a few relation types are temporal, they introduce an embedding representation – $\mathbf{p^t}$ – whether the relation p is temporal and an embedding $\mathbf{p}$ otherwise. Thus, the scoring function $\phi_\theta^{\text{TComplEx}}$ is extended into the TNTComplEx scoring function $\phi_\theta^{\text{TNTComplEx}}$:

$$\phi_\theta^{\text{TNTComplEx}}(s,p,o,\tau) = Re\left(\langle \mathbf{s}, \mathbf{p^t} \odot \mathbf{t}_\tau + \mathbf{p}, \overline{\mathbf{o}} \rangle\right) \tag{4}$$

2.1 Temporal regularisers

Temporal regularisers encourage tensor factorisation models to learn transformation for timestamp embeddings that capture specific temporal properties of real datasets. For instance, one would like the model to take advantage of the fact that most entities behave smoothly over time, i.e. learn similar transformations for closer timestamps. Alternatively, one would like to push away the representation of distant timestamps or allow the timestamp embeddings to be generated sequentially. Formally, a temporal regulariser is a penalty term $\Lambda(T)$ in the loss function, leading to the minimisation of the following objective:

$$\mathcal{L}(\phi_\theta) = \frac{1}{|S|} \sum_{(s,p,o,\tau)\in S} \left[\ell(\phi_\theta; (s,p,o,\tau)) + \lambda_1\Omega(\theta; (s,p,o,\tau))\right] + \lambda_2\Lambda(T). \tag{5}$$

Temporal smoothing. Most of the work in the literature [4, 9] adds a temporal smoothness objective to the loss function to encourage neighbouring timestamps to have close representations. The temporal smoothing regulariser is defined as:

$$\Lambda_p(T) = \frac{1}{|T|-1} \sum_{\tau=1}^{|T|-1} \|\mathbf{t}_{\tau+\mathbf{1}} - \mathbf{t}_\tau\|_{\mathbf{p}}^{\mathbf{p}}. \tag{6}$$

and it has been shown to increase performance on several benchmark datasets [4, 9]. However, it has been tested using only the N_3 norm. Hence, we decide to further investigate its impact by considering a wide class of norms in the L_p and N_p families. In our intuition, the choice of the norm function controls the strength of the smoothing, since norms differently penalize changes in the behavior of the entities on neighbouring timestamps. For instance, using an L_p norm always penalises no equal timestamp representations, and the value of p determines the magnitude of the penalty; while an N_p norm allows not penalising close representation for neighbouring timestamps (with $p = 5$ it starts to grow only for $|\mathbf{t_{\tau+1}} - \mathbf{t_\tau}| \geq \mathbf{0.4}$). In this case, the value of p controls the magnitude of the penalty and the range of distances to not penalise.

Linear3 Regulariser. In [8], they propose a new temporal regulariser, namely Linear3, that can be defined as follows:

$$\Lambda_p(T) = \frac{1}{|T|-1} \sum_{\tau=1}^{|T|-1} \|\mathbf{t_{\tau+1}} - \mathbf{t_\tau} - \mathbf{W_b}\|_{\mathbf{p}}^{\mathbf{p}}, \tag{7}$$

where $\mathbf{W_b} \in \mathbb{R}^{\mathbf{d}}$ denotes the embedding of a bias component between the neighbouring temporal embeddings, d the embedding size. The bias embedding is randomly initialised and then learned from the training process.

This linear regulariser promotes that the difference between embeddings of two adjacent timestamps is smaller than the difference between embeddings of two distant timestamps. In the context of this study, it is interesting to notice that Linear3 can be interpreted as the average score of triples $(\tau + 1,\ \textit{follows},\ \tau)$, where the embedding of the relation *follows* is given by the bias component. From this point of view, Linear3 encourages similar embeddings for neighbouring timestamps by explicitly modeling their temporal dynamic through the predicate *follows*.

Modelling Temporal Dynamics via Recurrent Architectures. Timestamp embeddings can be generated sequentially using a recurrent neural architecture that, starting from a random initialised hidden state, can learn implicitly the temporal dynamic through the learning process. Given a specific Recurrent Neural Network (RNN) architecture, the equation that describes its forward procedure acts as a temporal regulariser that maps the embedding of timestamp τ as:

$$\mathbf{t_\tau} = \mathit{MLP}(\mathit{RNN}(\mathbf{h_{\tau-1}}, \underline{\mathbf{0}})); \tau \in \{\mathbf{1}, ..., |\mathcal{T}|\} \tag{8}$$

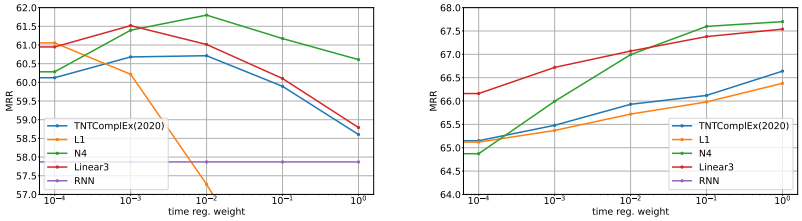
where $h_0 \in \mathbb{R}^m$ is the learnable initial hidden state, $\underline{\mathbf{0}}$ is the zero vector, *RNN* is the function that describes the recurrent architecture, *MLP* is a function that describes one multi-layer perceptron layer that has (m, d) channels, d is the embedding dimension and $m < d$.

3 Results and Discussion

We evaluate the impact of temporal regularisers on TNTComplEx for link prediction on temporal knowledge graph datasets, specifically ICEWS14, ICEWS05-15,

Model	ICEWS14				ICEWS05-15				YAGO15K			
	MRR	Hit@1	Hits@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
DistMult (2014)	43.9	32.3	-	67.2	45.6	33.7	-	69.1	27.5	21.5	-	43.8
ComplEx (2016)	47.0	35.0	54.0	71.0	49.0	37.0	55.0	73.0	36.0	29.0	36.0	54.0
TA-TransE (2018)	27.5	9.5	-	62.5	29.9	9.6	-	66.8	32.1	23.1	-	51.2
HyTE (2018)	29.7	10.8	41.6	65.5	31.6	11.6	44.5	68.1	-	-	-	-
TA-DistMult (2018)	47.7	-	36.3	68.6	47.4	34.6	-	72.8	29.1	21.6	-	47.6
TIMEPLEX (2020)	60.40	51.50	-	77.11	63.99	54.51	-	81.81	-	-	-	-
TeRo (2020)	56.2	46.8	62.1	73.2	58.6	46.9	66.8	79.5	-	-	-	-
BoxTE (2022)	61.5	53.2	**66.7**	76.7	66.7	58.2	71.9	82.0	-	-	-	-
TeLM (2021)[2]	61.41	53.39	66.0	76.12	66.70	58.99	71.28	80.85	-	-	-	-
ChronoR (2021)[2]	56.97	46.50	63.66	76.06	60.64	49.39	67.97	81.13	32.89	25.88	33.51	50.71
RoAN-DES (2023)	58.80	47.60	66.10	**78.80**	59.90	47.90	67.90	82.30	-	-	-	-
CE-CGCN (2024)	52.90	42.30	59.60	72.30	49.20	37.30	55.30	72.60	-	-	-	-
TNTComplEx (2020)[2]	60.72	51.91	65.92	77.17	66.64	58.34	71.82	81.67	35.94	28.49	36.84	53.75
TNTComplEx + Temp.Reg. (ours)	**61.80**	**53.60**	66.55	76.97	**67.70**	**59.90**	**72.35**	**82.30**	**37.05**	**29.00**	**39.62**	**54.02**

Table 1: Evaluation on the YAGO15K, ICEWS14, and ICEWS05-15 datasets. Results reported for previous related works are the best numbers reported in their respective paper.



(a) (b)

Fig. 1: Comparison of various temporal regularisers with different regularisation weights on TNTComplEx trained on ICEWS14 (a) and ICEWS05-15 (b).

and YAGO15K, following the experimental set-up described in [2]. We use baselines from both static and temporal KG embedding models. The source code to reproduce the full experimental results is made public on GitHub[1].

Table 1 demonstrates link prediction performance comparison on all datasets. Our TNTComplEx model achieves better performance than its original counterpart for all datasets and metrics. Most importantly, TNTComplEx, with one of our proposed temporal regularisers, outperforms all the competitors in terms of link prediction MRR and Hits@1 metric on the three datasets. It is important to note that two very recent works achieve better performance than our method. The first [9] is based on tensor factorization as TNTComplEx and utilizes the same temporal regulariser: hence, the insights provided by our study can allow to achieve even better performance. The second [3], instead, leverages several deep learning modules such as a GRU, and a CNN, which require more learnable parameters than a simple tensor dot product.

In Fig. 1, we plot a detailed comparison of some of our proposed regularisers for TNTComplEx, on ICEWS14 and ICEWS05-15 datasets. We observe that linear regularisers based on Linear3 or nuclear norms typically outperform TNTComplEx (2020), while RNNs struggle to generate very long sequences of embeddings. For

[1] `https://github.com/manuel-dileo/tkbc-reg`

instance, N_4 increases MRR by 1.08 points on ICEWS14, and carefully selecting regularisation weight can increase MRR up to 3.2 points.

Future Works. We plan to extend our analysis to inductive tasks and show how temporal regularisers can be generalised to work with unseen timestamps, entities, and relation types – for example, by leveraging recent work connecting factorisation-based models and GNNs [1] — and apply these solutions to challenging contexts such biological-inspired knowledge graphs and temporal heterogeneous networks gathered from Web3 online social platforms.

Acknowledgement. This work has been partially funded by the Italian Ministry of University and Research (MUR) and the European Union – NextGenerationEU in the framework of the PRIN 2022 project "AWESOME: Analysis framework for WEb3 SOcial MEdia" – CUP: I53D23003680006; and by the National Center for Gene Therapy and Drugs Based on RNA Technology—MUR (Project no. CN 00000041) funded by NextGeneration EU program

References

[1] Y. Chen, P. Mishra, L. Franceschi, P. Minervini, P. Stenetorp, and S. Riedel. Refactor gnns: Revisiting factorisation-based models from a message-passing perspective. In NeurIPS, 2022.

[2] A. García-Durán, S. Dumancic, and M. Niepert. Learning sequence encoders for temporal knowledge graph completion. In EMNLP, pages 4816–4821. Association for Computational Linguistics, 2018.

[3] M. He, L. Zhu, and L. Bai. Convtkg: A query-aware convolutional neural network-based embedding model for temporal knowledge graph completion. Neurocomputing, 588:127680, 2024.

[4] T. Lacroix, G. Obozinski, and N. Usunier. Tensor decompositions for temporal knowledge base completion. In ICLR. OpenReview.net, 2020.

[5] T. Lacroix, N. Usunier, and G. Obozinski. Canonical tensor decomposition for knowledge base completion. In ICML, volume 80 of Proceedings of Machine Learning Research, pages 2869–2878. PMLR, 2018.

[6] L. Loconte, N. Di Mauro, R. Peharz, and A. Vergari. How to turn your knowledge graph embeddings into generative models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, Advances in Neural Information Processing Systems, volume 36, pages 77713–77744. Curran Associates, Inc., 2023.

[7] L. Tran, B. Le, and T. Le. Exploring temporal knowledge graphs with compositional interactions and diachronic mechanisms. In Proceedings of the 32nd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), pages 173–180, 2024.

[8] C. Xu, Y. Chen, M. Nayyeri, and J. Lehmann. Temporal knowledge graph completion using a linear temporal regularizer and multivector embeddings. In NAACL-HLT, pages 2569–2578. Association for Computational Linguistics, 2021.

[9] F. Zhang, H. Chen, Y. Shi, J. Cheng, and J. Lin. Joint framework for tensor decomposition-based temporal knowledge graph completion. Information Sciences, 654:119853, 2024.

[10] Y. Zhang, X. Kong, Z. Shen, J. Li, Q. Yi, G. Shen, and B. Dong. A survey on temporal knowledge graph embedding: Models and applications. Knowl. Based Syst., 304:112454, 2024.