# Diachronic Embedding for Temporal Knowledge Graph Completion

**Rishab Goel***, **Seyed Mehran Kazemi***, **Marcus Brubaker, Pascal Poupart**
Borealis AI
{rishab.goel,mehran.kazemi,marcus.brubaker,pascal.poupart}@borealisai.com

## Abstract

Knowledge graph (KG) embedding approaches have proved effective for KG completion, however, they have been developed mostly for static KGs. Developing temporal KG embedding models is an increasingly important problem. In this paper, we build novel models for temporal KG completion through equipping static models with a diachronic entity embedding function which provides the characteristics of entities at *any* point in time. This is in contrast to the existing temporal KG embedding approaches where only static entity features are provided. The proposed embedding function is model-agnostic and can be potentially combined with any static model. We prove that combining it with SimplE results in a fully expressive model for temporal KG completion. Our experiments indicate the superiority of our proposal compared to existing baselines.

## 1 Introduction

Knowledge graphs (KGs) are directed graphs where nodes represent entities and (labeled) edges represent relationships among entities. Each edge in a KG corresponds to a fact and can be represented as a tuple such as (Mary, Liked, God Father) where Mary and God Father are called the head and tail entities respectively and Liked is a relation. KG completion, inferring new facts based on the existing ones, has been extensively studied for static KGs (see [15, 18, 14] for a summary). KG embedding approaches have offered remarkable results for KG completion on several benchmarks.

KG edges are typically associated with a timestamp or time interval; however, KG embedding approaches have been mostly designed for static KGs ignoring the temporal aspect. Recent work has shown a substantial boost in performance by extending these approaches to utilize time [6, 3, 13, 4]. The proposed extensions are mainly through computing a hidden representation for each timestamp and extending the score functions to utilize timestamp representations as well.

In this paper, we develop models for temporal KG completion (TKGC) based on an intuitive assumption: to provide a score for, e.g., (Mary, Liked, God Father, 1995), one needs to know Mary's and God Father's features on 1995; providing a score based on their current features may be misleading. That is because Mary's personality and the sentiment towards God Father may have been quite different on 1995 as compared to now. Consequently, learning a static representation for each entity – as is done by existing approaches – may be sub-optimal as such a representation only captures the entity features at the current time, or an aggregation of entity features during time.

To provide entity features at any given time, we define entity embedding as a function which takes an entity and a timestamp as input and provides a hidden representation for the entity at that time. Inspired by diachronic word embeddings, we call our proposed embedding *diachronic embedding (DE)*. We show the merit of our model on subsets of ICEWS [2] and GDELT [11] datasets.

*Equal contribution.

## 2 Background and Notation

**Notation:** Lower-case letters denote scalars, bold lower-case letters denote vectors, and bold upper-case letters denote matrices. $\boldsymbol{z}[n]$ represents the $n^{\text{th}}$ element of a vector $\boldsymbol{z}$. For $k$ vectors $\boldsymbol{z}_1, \dots, \boldsymbol{z}_k \in \mathbb{R}^d$, $\langle \boldsymbol{z}_1, \dots, \boldsymbol{z}_k \rangle = \sum_{n=1}^{d} (\boldsymbol{z}_1[n] * \cdots * \boldsymbol{z}_k[n])$ represents the sum of the element-wise product of the elements of the $k$ vectors.

**Temporal Knowledge Graph (Completion):** Let $\mathcal{V}$ be a finite set of entities, $\mathcal{R}$ be a finite set of relation types, and $\mathcal{T}$ be a finite set of timestamps. Let $\mathcal{W} \subset \mathcal{V} \times \mathcal{R} \times \mathcal{V} \times \mathcal{T}$ represent the set of all temporal tuples $(\mathsf{v}, \mathsf{r}, \mathsf{u}, \mathsf{t})$ that are facts (i.e. true in a world), where $\mathsf{v}, \mathsf{u} \in \mathcal{V}$, $\mathsf{r} \in \mathcal{R}$, and $\mathsf{t} \in \mathcal{T}$. Let $\mathcal{W}^c$ be the complement of $\mathcal{W}$. A temporal knowledge graph (KG) $\mathcal{G}$ is a subset of $\mathcal{W}$ (i.e. $\mathcal{G} \subset \mathcal{W}$). Temporal KG completion (TKGC) is the problem of inferring $\mathcal{W}$ from $\mathcal{G}$.

**KG Embedding:** Formally, we define an entity embedding as follows.

**Definition 1.** An *entity embedding*, $\texttt{EEMB} : \mathcal{V} \rightarrow \psi$, is a function which maps every entity $\mathsf{v} \in \mathcal{V}$ to a hidden representation in $\psi$ where $\psi$ is the class of non-empty tuples of vectors and/or matrices.

A *relation embedding* ($\texttt{REMB} : \mathcal{R} \rightarrow \psi$) is defined similarly. We refer to the hidden representation of an entity (relation) as the embedding of the entity (relation). A KG embedding model defines two things: 1- the EEMB and REMB functions, 2- a score function which takes EEMB and REMB as input and provides a score for a given tuple. The parameters of hidden representations are learned from data.

## 3 Diachronic Embedding

We propose an alternative entity embedding function definition which, besides entity, takes time as input as well. Inspired by diachronic word embeddings, we call such an embedding function a *diachronic entity embedding*. Below is a formal definition:

**Definition 2.** A *diachronic entity embedding*, $\texttt{DEEMB} : (\mathcal{V}, \mathcal{T}) \rightarrow \psi$, is a function which maps every pair $(\mathsf{v}, \mathsf{t})$, where $\mathsf{v} \in \mathcal{V}$ and $\mathsf{t} \in \mathcal{T}$, to a hidden representation in $\psi$.

One may take their favorite static KG embedding score function and make it temporal by replacing their entity embeddings with diachronic entity embeddings. The choice of the DEEMB function can be different for various temporal KGs depending on their properties. Here, we propose a DEEMB function which performs well on our benchmarks. We give the definition for models where the output of the DEEMB function is a tuple of vectors but it can be generalized to other cases as well. Let $\boldsymbol{z}_{\mathsf{v}}^{\mathsf{t}} \in \mathbb{R}^d$ be a vector in $\texttt{DEEMB}(\mathsf{v}, \mathsf{t})$ (i.e. $\texttt{DEEMB}(\mathsf{v}, \mathsf{t}) = (\dots, \boldsymbol{z}_{\mathsf{v}}^{\mathsf{t}}, \dots)$). We define $\boldsymbol{z}_{\mathsf{v}}^{\mathsf{t}}$ as follows:

$$\boldsymbol{z}_{\mathsf{v}}^{\mathsf{t}}[n] = \begin{cases} \boldsymbol{a}_{\mathsf{v}}[n] \sigma(\boldsymbol{w}_{\mathsf{v}}[n]\mathsf{t} + \boldsymbol{b}_{\mathsf{v}}[n]), & \text{if } 1 \leq n \leq \gamma d. \\ \boldsymbol{a}_{\mathsf{v}}[n], & \text{if } \gamma d < n \leq d. \end{cases} \tag{1}$$

where $\boldsymbol{a}_{\mathsf{v}} \in \mathbb{R}^d$ and $\boldsymbol{w}_{\mathsf{v}}, \boldsymbol{b}_{\mathsf{v}} \in \mathbb{R}^{\gamma d}$ are (entity-specific) vectors with learnable parameters and $\sigma$ is an activation function. Intuitively, entities may have some features that change over time and some features that remain fixed. The first $\gamma d$ vector elements vector in Equation (1) capture temporal features and the other $(1 - \gamma)d$ elements capture static features. $0 \leq \gamma \leq 1$ is a hyper-parameter controlling the percentage of temporal features. While static features can be obtained from the temporal ones if the optimizer sets some elements of $\boldsymbol{w}_{\mathsf{v}}$ to zero, explicitly modeling static features helps reduce the number of parameters and avoid overfitting to temporal signals (see Section 4.1).

Intuitively, by learning $\boldsymbol{w}_{\mathsf{v}}$s and $\boldsymbol{b}_{\mathsf{v}}$s, the model learns how to turn entity features on and off at different points in time so accurate temporal predictions can be made about them at any time. $\boldsymbol{a}_{\mathsf{v}}$s control the importance of the features. We mainly use *sine* activations because one sine can model several on and off states. Our experiments explore other activation functions as well and provide more intuition.

**Model-Agnosticism:** One may construct temporal versions of TransE, DistMult, SimplE, or other models by replacing their EEMB function with DEEMB in Equation 1. We refer to the resulting models as *DE-TransE*, *DE-DistMult*, *DE-SimplE* and so forth, where *DE* is short for **D**iachronic **E**mbedding.

**Learning:** The facts in a KG $\mathcal{G}$ are split into $train$, $validation$, and $test$ sets. Model parameters are learned using stochastic gradient descent with mini-batches. Let $B \subset train$ be a mini-batch. For each fact $f = (\mathsf{v}, \mathsf{r}, \mathsf{u}, \mathsf{t}) \in B$, we generate two queries: 1- $(\mathsf{v}, \mathsf{r}, ?, \mathsf{t})$ and 2- $(?, \mathsf{r}, \mathsf{u}, \mathsf{t})$. For the first query, we generate a candidate answer set $\mathsf{C}_{f,\mathsf{v}}$ which contains $\mathsf{v}$ and $n$ (hereafter referred to as

Table 1: Results on ICEWS14, ICEWS05-15, and GDELT. Best results are in bold.

| | ICEWS14 | | | | ICEWS05-15 | | | | GDELT | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Model | MRR | Hit@1 | Hit@3 | Hit@10 | MRR | Hit@1 | Hit@3 | Hit@10 | MRR | Hit@1 | Hit@3 | Hit@10 |
| TransE [1] | 0.280 | 9.4 | - | 63.7 | 0.294 | 9.0 | - | 66.3 | 0.113 | 0.0 | 15.8 | 31.2 |
| DistMult [19] | 0.439 | 32.3 | - | 67.2 | 0.456 | 33.7 | - | 69.1 | 0.196 | 11.7 | 20.8 | 34.8 |
| SimplE [8] | 0.458 | 34.1 | 51.6 | 68.7 | 0.478 | 35.9 | 53.9 | 70.8 | 0.206 | 12.4 | 22.0 | 36.6 |
| ConT [12] | 0.185 | 11.7 | 20.5 | 31.5 | 0.163 | 10.5 | 18.9 | 27.2 | 0.144 | 8.0 | 15.6 | 26.5 |
| TTransE [6] | 0.255 | 7.4 | - | 60.1 | 0.271 | 8.4 | - | 61.6 | 0.115 | 0.0 | 16.0 | 31.8 |
| HyTE [3] | 0.297 | 10.8 | 41.6 | 65.5 | 0.316 | 11.6 | 44.5 | 68.1 | 0.118 | 0.0 | 16.5 | 32.6 |
| TA-DistMult [4] | 0.477 | 36.3 | - | 68.6 | 0.474 | 34.6 | - | 72.8 | 0.206 | 12.4 | 21.9 | 36.5 |
| DE-TransE | 0.326 | 12.4 | 46.7 | 68.6 | 0.314 | 10.8 | 45.3 | 68.5 | 0.126 | 0.0 | 18.1 | 35.0 |
| DE-DistMult | 0.501 | 39.2 | 56.9 | 70.8 | 0.484 | 36.6 | 54.6 | 71.8 | 0.213 | 13.0 | 22.8 | 37.6 |
| DE-SimplE | **0.526** | **41.8** | **59.2** | **72.5** | **0.513** | **39.2** | **57.8** | **74.8** | **0.230** | **14.1** | **24.8** | **40.3** |

negative ratio) other entities selected randomly from $\mathcal{V}$. For the second query, we generate a similar candidate answer set $\mathsf{C}_{f,\mathsf{u}}$. Then we minimize the cross entropy loss which has been used and shown good results for both static and temporal KG completion (see, e.g., [7, 4]):

$$\mathcal{L} = -\Big( \sum_{f=(\mathsf{v},\mathsf{r},\mathsf{u},\mathsf{t})\in B} \frac{\exp(\phi(\mathsf{v},\mathsf{r},\mathsf{u},\mathsf{t}))}{\sum_{\mathsf{u}'\in\mathsf{C}_{f,\mathsf{u}}} \exp(\phi(\mathsf{v},\mathsf{r},\mathsf{u}',\mathsf{t}))} + \frac{\exp(\phi(\mathsf{v},\mathsf{r},\mathsf{u},\mathsf{t}))}{\sum_{\mathsf{v}'\in\mathsf{C}_{f,\mathsf{v}}} \exp(\phi(\mathsf{v}',\mathsf{r},\mathsf{u},\mathsf{t}))} \Big)$$

**Expressivity:** We prove the full expressivity of DE-SimplE.

**Definition 3.** A model with parameters $\theta$ is *fully expressive* if given any world with true tuples $\mathcal{W}$ and false tuples $\mathcal{W}^c$, there exists an assignment for $\theta$ that correctly classifies the tuples in $\mathcal{W}$ and $\mathcal{W}^c$.

**Theorem 1** (**Expressivity**). *DE-SimplE is fully expressive for temporal knowledge graph completion.*

## 4 Experiments & Results

**Datasets:** Our datasets are subsets of ICEWS [2] and GDELT [11]. For ICEWS, we use the two subsets generated by García-Durán *et al.* [4]: 1- *ICEWS14* corresponding to the facts in 2014 and 2- *ICEWS05-15* corresponding to the facts between 2005 to 2015. For GDELT, we use the subset extracted by Trivedi *et al.* [17] corresponding to 3,419,607 facts from April 1, 2015 to March 31, 2016. We changed the train/validation/test sets following a similar procedure as in [1] to make the problem into a TKGC rather than an extrapolation problem.

**Metrics:** We report the filtered versions of mean reciprocal rank (MRR) and Hit@k for $k = 1, 3, 10$.

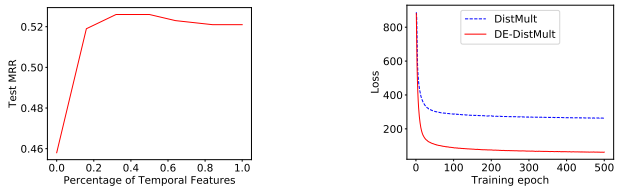
**Comparative Study:** We compare the existing baselines with three variants of our model: 1- DE-TransE, 2- DE-DistMult, and 3- DE-SimplE. The obtained results are in Table 1. DE-TransE outperforms the other TransE-based baselines (TTransE and HyTE) on ICEWS14 and GDELT and gives on-par results with HyTE on ICEWS05-15. This result shows the superiority of our diachronic embeddings compared to TTransE and HyTE. DE-DistMult outperforms TA-DistMult, the only DistMult-based baseline, showing the superiority of our diachronic embedding compared to TA-DistMult. Moreover, DE-DistMult outperforms all TransE-based baselines. Finally, just as SimplE beats TransE and DistMult due to its higher expressivity, our results show that DE-SimplE beats DE-TransE, DE-DistMult, and the other baselines due to its higher expressivity.

### 4.1 Model Variants & Ablation Study

**Activation Function:** So far, we used *sine* as the activation function in Equation 1. The performance for other activation functions are presented in Table 2. The results show that other activation functions also perform well. Specifically, squared exponential performs almost on-par with sine. We believe one reason why sine and squared exponential give better performance is because a combination of sine or square exponential features can generate more sophisticated features than a combination of Tanh, sigmoid, or ReLU features. While a temporal feature with Tanh or sigmoid as the activation corresponds to a smooth off-on (or on-off) temporal switch, a temporal feature with sine or squared exponential activation corresponds to two (or more) switches (e.g., off-on-off) which can potentially model relations that start at some time and end after a while (e.g., PresidentOf). These results also provide evidence for the effectiveness of diachronic embedding across several DEEMB functions.

Table 2: Results for different variations of our model on ICEWS14.

| Model | Variation | MRR | Hit@1 | Hit@3 | Hit@10 |
|---|---|---|---|---|---|
| DE-DistMult | No variation (Activation function: *Sine*) | 0.501 | 39.2 | 56.9 | 70.8 |
| DE-DistMult | Activation function: *Tanh* | 0.486 | 37.5 | 54.7 | 70.1 |
| DE-DistMult | Activation function: *Sigmoid* | 0.484 | 37.0 | 54.6 | 70.6 |
| DE-DistMult | Activation function: *Leaky ReLU* | 0.478 | 36.3 | 54.2 | 70.1 |
| DE-DistMult | Activation function: *Squared Exponential* | 0.501 | 39.0 | 56.8 | 70.9 |
| DE-DistMult | Diachronic embedding for both entities and relations | 0.502 | 39.4 | 56.6 | 70.4 |
| DE-DistMult | $\boldsymbol{a}_{\mathsf{v}}[n] = 1$ for $1 \leq n \leq \gamma d$ for all $\mathsf{v} \in \mathcal{V}$ | 0.458 | 34.4 | 51.8 | 68.3 |
| DE-DistMult | $\boldsymbol{w}_{\mathsf{v}}[n] = 1$ for $1 \leq n \leq \gamma d$ for all $\mathsf{v} \in \mathcal{V}$ | 0.470 | 36.4 | 53.1 | 67.1 |
| DE-DistMult | $\boldsymbol{b}_{\mathsf{v}}[n] = 0$ for $1 \leq n \leq \gamma d$ for all $\mathsf{v} \in \mathcal{V}$ | 0.498 | 38.9 | 56.2 | 70.4 |




(a) (b)

Figure 1: (a) Test MRR of DE-SimplE on ICEWS14 as a function of $\gamma$. (b) The training curve for DistMult and DE-DistMult.

**Adding Diachronic Embedding for Relations:** Compared to entities, we hypothesize that relations may evolve at a very lower rate or, for some relations, evolve only negligibly. Therefore, modeling relations with a static representation may suffice. To test this hypothesis, we ran DE-DistMult on ICEWS14 where relation embeddings are also a function of time. From the obtained results in Table 2, one can see that the model with diachronic embeddings for both entities and relations performs on-par with the model with diachronic embedding only for entities. We conducted the same experiment on ICEWS05-15 (which has a longer time horizons) and GDELT and observed similar results. These results show that at least on our benchmarks, modeling the evolution of relations may not be helpful. Future work can test this hypothesis on datasets with other types of relations and longer horizons.

**Importance of Model Parameters Used in Equation 1:** In Equation 1, the temporal part of the embedding contains three components: $\boldsymbol{a}_{\mathsf{v}}$, $\boldsymbol{w}_{\mathsf{v}}$, and $\boldsymbol{b}_{\mathsf{v}}$. To measure the importance of each component, we ran DE-DistMult on ICEWS14 under three settings: 1- when $\boldsymbol{a}_{\mathsf{v}}$s are removed (i.e. set to 1), 2- when $\boldsymbol{w}_{\mathsf{v}}$s are removed (i.e. set to 1), and 3- when $\boldsymbol{b}_{\mathsf{v}}$s are removed (i.e. set to 0). According to the results in Table 2, all three components are important, especially $\boldsymbol{a}_{\mathsf{v}}$s and $\boldsymbol{w}_{\mathsf{v}}$s. Removing $\boldsymbol{b}_{\mathsf{v}}$s does not affect the results as much as $\boldsymbol{a}_{\mathsf{v}}$s and $\boldsymbol{w}_{\mathsf{v}}$s. Therefore, if one needs to reduce the number of parameters, removing $\boldsymbol{b}_{\mathsf{v}}$ may be a good option as long as they can tolerate a slight reduction in accuracy.

**Static Features:** Figure 1(a) shows the test MRR of DE-SimplE on ICEWS14 as a function of $\gamma$, the percentage of temporal features. As soon as some features become temporal, a substantial boost in performance can be observed. As $\gamma$ becomes larger, MRR reaches a peak and then slightly drops. This slight drop in performance can be due to overfitting to temporal cues. This result demonstrates that modeling static features explicitly can help reduce the number of learnable parameters and avoid overfitting. Such a design choice may be even more important when the embedding dimensions are larger. However, it comes at the cost of adding one hyper-parameter to the model.

**Training Curve:** While it has been argued that using sine activation functions may complicate training in some neural network architectures (see, e.g., [10, 5]), it can be in viewed Figure 1(b) that when using sine activations, the training curve for our model is quite stable.

## 5 Conclusion

We developed a diachronic embedding for temporal KG completion which provides a hidden representation for the entities of a temporal KG at any point in time. Future work includes designing functions other than the one proposed in Equation 1, a comprehensive study of which functions are favored by different types of KGs, and using our proposed embedding for diachronic word embedding.

## References


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

[2] Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. Icews coded event data. *Harvard Dataverse*, 12, 2015.

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

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

[5] Tuomas Virtanen Giambattista Parascandolo, Heikki Huttunen. Taming the waves: sine as activation function in deep neural networks. 2017.

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

[7] Ondrej Bajgar Kadlec, Rudolf and Jan Kleindienst. Knowledge base completion: Baselines strike back. *arXiv preprint arXiv:1705.10744*, 2017.

[8] Seyed Mehran Kazemi and David Poole. SimplE embedding for link prediction in knowledge graphs. In *NeurIPS*, 2018.

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] Alan Lapedes and Robert Farber. Nonlinear signal processing using neural networks: Prediction and system modelling. Technical report, 1987.

[11] Kalev Leetaru and Philip A Schrodt. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, volume 2, pages 1–49. Citeseer, 2013.

[12] Yao Ma, Ziyi Guo, Zhaochun Ren, Eric Zhao, Jiliang Tang, and Dawei Yin. Streaming graph neural networks. *arXiv preprint arXiv:1810.10627*, 2018.

[13] Yunpu Ma, Volker Tresp, and Erik A Daxberger. Embedding models for episodic knowledge graphs. *Journal of Web Semantics*, 2018.

[14] Dat Quoc Nguyen. An overview of embedding models of entities and relationships for knowledge base completion. *arXiv preprint arXiv:1703.08098*, 2017.

[15] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

[16] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

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

[18] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE TKDE*, 29(12):2724–2743, 2017.

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

# A Experiments & Results

**Datasets:** A summary of dataset statistics can be found in Table 3.

**Metrics:** For each fact $f = (\mathsf{v}, \mathsf{r}, \mathsf{u}, \mathsf{t}) \in test$, we create two queries: 1- $(\mathsf{v}, \mathsf{r}, ?, \mathsf{t})$ and 2- $(?, \mathsf{r}, \mathsf{u}, \mathsf{t})$. For the first query, the model ranks all entities in $\mathsf{u} \cup \overline{\mathsf{C}}_{f,\mathsf{u}}$ where $\overline{\mathsf{C}}_{f,\mathsf{u}} = \{\mathsf{u}' : \mathsf{u}' \in \mathcal{V}, (\mathsf{v}, \mathsf{r}, \mathsf{u}', \mathsf{t}) \notin \mathcal{G}\}$. This corresponds to the filtered setting commonly used in the literature [1]. We follow a similar approach for the second query. Let $k_{f,\mathsf{u}}$ and $k_{f,\mathsf{v}}$ represent the ranking for u and v for the two queries respectively. We report *mean reciprocal rank (MRR)* defined as $\frac{1}{2*|test|} \sum_{f=(\mathsf{v},\mathsf{r},\mathsf{u},\mathsf{t}) \in test} (\frac{1}{k_{f,\mathsf{u}}} + \frac{1}{k_{f,\mathsf{v}}})$. Compared to its counterpart *mean rank* which is largely influenced by a single bad prediction, MRR is more stable [15]. We also report Hit@1, Hit@3 and Hit@10 measures where Hit@k is defined as $\frac{1}{2*|test|} \sum_{f=(\mathsf{v},\mathsf{r},\mathsf{u},\mathsf{t}) \in test} (\mathbb{1}_{k_{f,\mathsf{u}} \leq k} + \mathbb{1}_{k_{f,\mathsf{v}} \leq k})$, where $\mathbb{1}_{cond}$ is 1 if $cond$ holds and 0 otherwise.

**Implementation:** We implemented our model and the baselines in PyTorch [16]. We ran our experiments on a node with four GPUs. For the two ICEWS datasets, we report the results for some of the baselines from [4]. For the other experiments on these datasets, for the fairness of results, we follow a similar experimental setup as in [4] by using the ADAM optimizer [9] and setting learning rate $= 0.001$, batch size $= 512$, negative ratio $= 500$, embedding size $= 100$, and validating every 20 epochs selecting the model giving the best validation MRR. Following the best results obtained in [13] (and considering the memory restrictions), for ConT we set embedding size $= 40$, batch size $= 32$ on ICEWS14 and GDELT and 16 on ICEWS05-15. We validated dropout values from $\{0.0, 0.2, 0.4\}$. We tuned $\gamma$ for our model from the values $\{16, 32, 64\}$. For GDELT, we used a similar setting but with a negative ratio $= 5$ due to the large size of the dataset. Unless stated otherwise, we use $sine$ as the activation function for Equation (1). Since the timestamps in our datasets are dates rather than single numbers, we apply the temporal part of Equation (1) to year, month, and day separately (with different parameters) thus obtaining three temporal vectors. Then we take an element-wise sum of the resulting vectors obtaining a single temporal vector. Intuitively, this can be viewed as converting a date into a timestamp in the embedded space.

# B Proof of Theorem

**Theorem 1.** *DE-SimplE is fully expressive for temporal knowledge graph completion.*

*Proof.* For every entity $\mathsf{v}_i \in \mathcal{V}$, let $\texttt{DEEMB}(\mathsf{v}_i, t) = (\vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}, \overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i})$ where, according to Equation (1) with sine activations, $\vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i} \in \mathbb{R}^d$ and $\overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i} \in \mathbb{R}^d$ are defined as follows:

$$\vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}[n] = \begin{cases} \vec{\boldsymbol{a}}_{\mathsf{v}_i}[n] \sin(\vec{\boldsymbol{w}}_{\mathsf{v}_i}[n]t + \vec{\boldsymbol{b}}_{\mathsf{v}_i}[n]), & \text{if } n \leq \gamma. \\ \vec{\boldsymbol{a}}_{\mathsf{v}_i}[n], & \text{if } n > \gamma. \end{cases} \tag{2}$$

and:

$$\overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}[n] = \begin{cases} \overleftarrow{\boldsymbol{a}}_{\mathsf{v}_i}[n] \sin(\overleftarrow{\boldsymbol{w}}_{\mathsf{v}_i}[n]t + \overleftarrow{\boldsymbol{b}}_{\mathsf{v}_i}[n]), & \text{if } n \leq \gamma. \\ \overleftarrow{\boldsymbol{a}}_{\mathsf{v}_i}[n], & \text{if } n > \gamma. \end{cases} \tag{3}$$

We provide the proof for a specific case of DE-SimplE where the elements of $\vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}}$s are all temporal and the elements of $\overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}}$s are all non-temporal. This specific case can be achieved by setting $\gamma = d$, and $\overleftarrow{\boldsymbol{w}}_{\mathsf{v}}[n] = 0$ and $\overleftarrow{\boldsymbol{b}}_{\mathsf{v}}[n] = \frac{\pi}{2}$ for all $\mathsf{v} \in \mathcal{V}$ and for all $1 \leq n \leq d$. If this specific case of DE-SimplE is fully expressive, so is DE-SimplE. In this specific case, $\vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}$ and $\overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}$ for every $\mathsf{v}_i \in \mathcal{V}$ can be re-written as follows:

$$\vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}[n] = \vec{\boldsymbol{a}}_{\mathsf{v}_i}[n] \sin(\vec{\boldsymbol{w}}_{\mathsf{v}_i}[n]t + \vec{\boldsymbol{b}}_{\mathsf{v}_i}[n]) \tag{4}$$

$$\overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}[n] = \overleftarrow{\boldsymbol{a}}_{\mathsf{v}_i}[n] \tag{5}$$

For every relation $\mathsf{r}_j \in \mathcal{R}$, let $\texttt{REMB}(\mathsf{r}) = (\vec{\boldsymbol{z}}_{\mathsf{r}_j}, \overleftarrow{\boldsymbol{z}}_{\mathsf{r}_j})$. To further simplify the proof, following [8], we only show how the embedding values can be set such that $\langle \vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}, \vec{\boldsymbol{z}}_{\mathsf{r}_j}, \overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_k} \rangle$ becomes a positive number if $(\mathsf{v}_i, \mathsf{r}_j, \mathsf{v}_k, \mathsf{t}) \in \mathcal{W}$ and a negative number if $(\mathsf{v}_i, \mathsf{r}_j, \mathsf{v}_k, \mathsf{t}) \in \mathcal{W}^c$. Extending the proof the case where the score contains both components ($\langle \vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i}, \vec{\boldsymbol{z}}_{\mathsf{r}_j}, \overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_k} \rangle$ and $\langle \vec{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_k}, \overleftarrow{\boldsymbol{z}}_{\mathsf{r}_j}, \overleftarrow{\boldsymbol{z}}^{\mathsf{t}}_{\mathsf{v}_i} \rangle$) can be done by doubling the size of the embedding vectors and following a similar procedure as the one explained below for the second half of the vectors.

Table 3: Statistics on ICEWS14, ICEWS05-15, and GDELT.

| Dataset | $\lvert\mathcal{V}\rvert$ | $\lvert\mathcal{R}\rvert$ | $\lvert\mathcal{T}\rvert$ | $\lvert train\rvert$ | $\lvert validation\rvert$ | $\lvert test\rvert$ | $\lvert\mathcal{G}\rvert$ |
|---|---|---|---|---|---|---|---|
| ICEWS14 | 7,128 | 230 | 365 | 72,826 | 8,941 | 8,963 | 90,730 |
| ICEWS05-15 | 10,488 | 251 | 4017 | 386,962 | 46,275 | 46,092 | 479,329 |
| GDELT | 500 | 20 | 366 | 2,735,685 | 341,961 | 341,961 | 3,419,607 |

Assume $d = |\mathcal{R}| \cdot |\mathcal{V}| \cdot |\mathcal{T}| \cdot L$ where $L \in \mathbb{N}$ is a natural number. These vectors can be viewed as $|\mathcal{R}|$ blocks of size $|\mathcal{V}| \cdot |\mathcal{T}| \cdot L$. For the $j^{\text{th}}$ relation $\mathsf{r}_j$, let $\vec{\boldsymbol{z}}_{\mathsf{r}_j}$ be zero everywhere except on the $j^{\text{th}}$ block where it is 1 everywhere. With such a value assignment to $\vec{\boldsymbol{z}}_{\mathsf{r}_j}$s, to find the score for a fact $(\mathsf{v}_i, \mathsf{r}_j, \mathsf{v}_k, \mathsf{t})$, only the $j^{\text{th}}$ block of each embedding vector is important. Let us now focus on the $j^{\text{th}}$ block.

The size of the $j^{\text{th}}$ block (similar to all other blocks) is $|\mathcal{V}| \cdot |\mathcal{T}| \cdot L$ and it can be viewed as $|\mathcal{V}|$ sub-blocks of size $|\mathcal{T}| \cdot L$. For the $i^{\text{th}}$ entity $\mathsf{v}_i$, let the values of $\vec{\boldsymbol{a}}_{\mathsf{v}_i}$ be zero in all sub-blocks except the $i^{\text{th}}$ sub-block. With such a value assignment, to find the score for a fact $(\mathsf{v}_i, \mathsf{r}_j, \mathsf{v}_k, \mathsf{t})$, only the $i^{\text{th}}$ sub-block of the $j^{\text{th}}$ block is important. Note that this sub-block is unique for each tuple $(\mathsf{v}_i, \mathsf{r}_j)$. Let us now focus on the $i^{\text{th}}$ sub-block of the $j^{\text{th}}$ block.

The size of the $i^{\text{th}}$ sub-block of the $j^{\text{th}}$ block is $|\mathcal{T}| \cdot L$ and it can be viewed as $|\mathcal{T}|$ sub-sub-blocks of size $L$. According to the Fourier sine series, with a large enough $L$, we can set the values for $\vec{\boldsymbol{a}}_{\mathsf{v}_i}$, $\vec{\boldsymbol{w}}_{\mathsf{v}_i}$, and $\vec{\boldsymbol{b}}_{\mathsf{v}_i}$ in a way that the sum of the elements of $\vec{\boldsymbol{z}}^{\,\mathsf{t}}_{\mathsf{v}_i}$ for the $p^{\text{th}}$ sub-sub-block becomes 1 when $\mathsf{t} = \mathsf{t}_p$ (where $\mathsf{t}_p$ is the $p^{\text{th}}$ timestamp in $\mathcal{T}$) and 0 when $t$ is a timestamp other than $\mathsf{t}_p$. Note that this sub-sub-block is unique for each tuple $(\mathsf{v}_i, \mathsf{r}_j, \mathsf{t}_p)$.

Having the above value assignments, if $(\mathsf{v}_i, \mathsf{r}_j, \mathsf{v}_k, \mathsf{t}_p) \in \mathcal{W}$, we set all the values in the $p^{\text{th}}$ sub-sub-block of the $i^{\text{th}}$ sub-block of the $j^{\text{th}}$ block of $\overleftarrow{\boldsymbol{a}}_{\mathsf{v}_k}$ to 1. With this assignment, $\langle \vec{\boldsymbol{z}}^{\,\mathsf{t}}_{\mathsf{v}_i}, \vec{\boldsymbol{z}}_{\mathsf{r}_j}, \overleftarrow{\boldsymbol{z}}^{\,\mathsf{t}}_{\mathsf{v}_k} \rangle = 1$ at $\mathsf{t} = \mathsf{t}_p$. If $(\mathsf{v}_i, \mathsf{r}_j, \mathsf{v}_k, \mathsf{t}_p) \in \mathcal{W}^c$, we set all the values for the $p^{\text{th}}$ sub-sub-block of the $i^{\text{th}}$ sub-block of the $j^{\text{th}}$ block of $\overleftarrow{\boldsymbol{a}}_{\mathsf{v}_k}$ to $-1$. With this assignment, $\langle \vec{\boldsymbol{z}}^{\,\mathsf{t}}_{\mathsf{v}_i}, \vec{\boldsymbol{z}}_{\mathsf{r}_j}, \overleftarrow{\boldsymbol{z}}^{\,\mathsf{t}}_{\mathsf{v}_k} \rangle = -1$ at $\mathsf{t} = \mathsf{t}_p$. □