Comparing Apples and Oranges? On the Evaluation of Methods for Temporal Knowledge Graph Forecasting

Julia Gastinger[1,2][0000-0003-1914-6723] (✉), Timo Sztyler[1][0000-0001-8132-5920], Lokesh Sharma[1][0009-0009-2522-1209], Anett Schuelke[1], and Heiner Stuckenschmidt[2][0000-0002-0209-3859]

[1] NEC Laboratories Europe, Heidelberg, Germany
`{firstname.lastname}@neclab.eu`
[2] University of Mannheim, Chair of Artificial Intelligence, Mannheim, Germany
`heiner.stuckenschmidt@uni-mannheim.de`

Abstract. Due to its ability to incorporate and leverage time information in relational data, Temporal Knowledge Graph (TKG) learning has become an increasingly studied research field. To predict the future based on TKG, researchers have presented innovative methods for Temporal Knowledge Graph Forecasting. However, the experimental procedures employed in this research area exhibit inconsistencies that significantly impact empirical results, leading to distorted comparisons among models. This paper focuses on the evaluation of TKG Forecasting models: We examine the evaluation settings commonly used in this research area and highlight the issues that arise. To make different approaches to TKG Forecasting more comparable, we propose a unified evaluation protocol and apply it to re-evaluate state-of-the-art models on the most commonly used datasets. Ultimately, we demonstrate the significant difference in results caused by different evaluation settings. We believe this work provides a solid foundation for future evaluations of TKG Forecasting models, thereby contributing to advancing this growing research area.

Keywords: Temporal Knowledge Graphs · Temporal Graphs · Temporal Knowledge Graph Forecasting

1 Introduction

Temporal Knowledge Graphs (TKG) are Knowledge Graphs (KG) where facts occur, recur or evolve over time [29]. TKG can accommodate time-evolving multi-relational data by extending facts with a timestamp to indicate that a triple is valid at this timestamp [7]. The research field of TKG Forecasting, or TKG Extrapolation, aims at predicting facts at future timesteps, based on the KG history [27]. Recently, various methods have been proposed to advance the field ([13], [19], [8], [7], [17], [18], [27], [31]).

Unfortunately, and despite the progress made so far in TKG Forecasting, various reported experimental settings show discrepancies: first, the existing models are evaluated on scores computed with different filter settings; second, models for single-step prediction that predict one step to the future are lumped together with models for multi-step prediction that predict multiple steps to the future; third, multiple versions of the same datasets exist. Last but not least, some models do use the information from the validation set for testing, whereas others do not. These four issues can strongly influence the empirical results and significantly decrease comparability across works. As an example, the best results in single-step setting are in average 6% better than the best results in multi-step setting. Consequently, it is very difficult to understand existing methods' strengths or weaknesses or to identify the currently best-performing method.

In this paper, we address the aforementioned issues in the evaluation of TKG Forecasting models. We first provide an overview of existing models for TKG Forecasting (Section 2). We then describe common evaluation settings and compare those settings utilized in state-of-the-art approaches to highlight the inconsistencies (Section 5). In this context, we explain the problems we discovered for each setting. As it is essential to evaluate models in a consistent way, we propose a unified evaluation protocol using reasonable and sound evaluation settings (Section 4). We re-evaluate state-of-the-art models on this protocol and show results for eight state-of-the-art models on five commonly used datasets (Section 5). In addition, we provide insights into the influence of different setups on the result scores. We hope to set a new standard for rigorous evaluations of new models in this growing research field. Our contributions are:

1. A comprehensive discussion of evaluation settings and accompanying problems for TKG Forecasting.
2. The design of a unified evaluation protocol for TKG Forecasting from reasonable evaluation settings.
3. An extensive re-evaluation of state-of-the-art models on a consistent evaluation protocol, showing results and insights on the influence of different evaluation settings on these results.

Our work does not question the methods for TKG Forecasting developed by individual researchers. Instead, it aims at giving a fresh view on the state of the field as a whole and provides a solid basis for working on remaining problems.

2 Terminology and Related Work

2.1 Terminology

A TKG is formalized as a sequence of timestamped Knowledge Graphs, $G = (G_1, G_2, ..., G_t, ...)$. A timestamped KG $G_t = \{\mathcal{V}, \mathcal{R}, \mathcal{E}_t\}$, or KG snapshot, describes the TKG at timestep t, with the set of entities $\mathcal{V}$, the set of relations $\mathcal{R}$, and the set of facts $\mathcal{E}_t$ at discrete timestamp t. Facts $\mathcal{E}_t$ are quadruples (s, r, o, t), with $s, o, \in \mathcal{V}$, and $r \in \mathcal{R}$, for example (Kamala Harris, visit, France, 2021-11-10). Entity prediction for TKG Forecasting is the task of predicting the missing object entity $(s, r, ?, t+k)$ and subject entity $(?, r, o, t+k)$ for a query, with $k \in \mathbb{N}^+$. [19]

2.2 Related Work on Temporal Knowledge Graph Forecasting

In recent years (2017-2022), researchers have proposed various methods for TKG Forecasting:

Graph Neural Networks (GNNs): A large group of models leverages a GNN [25, 23] in combination with a sequential approach to integrate the structural and sequential information. RE-Net [13] applies an autoregressive architecture. It learns the temporal dependency from a sequence of graphs and the local structural dependency from the neighborhood. The occurrence of a fact is modeled as a probability distribution conditioned on the temporal sequence of past snapshots. RE-Net can predict full graphs. RE-GCN [19] also models the sequence of the Knowledge Graph snapshots recurrently. For this, it combines a convolutional graph Neural Network with a sequential Neural Network model. Further, RE-GCN introduces a static graph constraint to take into account additional information like entity types. TANGO [8] bases on neural ordinary differential equations to model the temporal sequences combined with a GNN to capture the structural information. In addition, the authors introduce a stochastic jump method to incorporate stochastic events, i.e., triples appearing or disappearing over time. xERTE [7] bases on so-called temporal relational attention mechanisms. To answer a query, it extracts query-relevant subgraphs. Further, it computes and propagates attention scores to identify the relevant evidence in the subgraphs, using a modified time-aware version of a message passing. CEN [17] integrates a Convolutional Neural Network which can handle evolutional patterns of different lengths via a learning strategy that learns these evolutional patterns from short to long. The model can learn in an online setting, meaning that it is updated with historical facts during testing.

Reinforcement Learning: CluSTeR [18] introduces a two-step process: First, a Reinforcement Learning agent, working with randomized beam strategy, searches and induces clue paths related to a given query. Second, an adapted GNN and sequence method models temporal information among the clues to find answers to a query. TimeTraveler [27] leverages a Reinforcement Learning model based on temporal paths. Starting from the query's subject node, the agent traverses outgoing edges across graph snapshots. For this, TimeTraveler samples actions according to transition probabilities, which are based on dynamic embeddings of the query, the path history, and the candidate actions. TimeTraveler uses a time-shaped reward based on Dirichlet distribution [14]. The model is able to predict in the inductive setting.

Rule-based Approaches: TLogic [21], a symbolic framework, learns so-called temporal logic rules via temporal random walks, traversing edges through the graph backward in time. TLogic applies the rules to events that happened prior to the query. For scoring the answer candidates, it takes into account the rules' confidence as well as time differences.

Other: CyGNet [31] predicts future facts purely based on the appearance of historical facts. For this, to answer a query, it first computes each entity's embedding vector. Further, using these embeddings, it computes entity probabilities by combining predictions from a so-called "copy mode" that computes probabilities for historical events based on the repetition of facts in history and a "generation mode" that computes probabilities for every entity.

In our work, we analyze the evaluation discrepancies of the introduced models and evaluate the models on a joint evaluation protocol.

In addition to the described methods, there are also approaches focusing on a slightly different problem setting. We exclude these from our evaluation, but list them below for completeness: Know-Evolve [29] and the Graph Hawkes Neural Network (GHNN) [9] utilize temporal point processes to estimate conditional probabilities of future facts in a continuous time setting. Unlike the other methods discussed in this section, Know-Evolve and GHNN allow scenarios where no facts occur at the same timestamp [19]. Due to their distinct problem setting, where continuous time is considered, these works are not included in our evaluation.

2.3 Related Work on the Evaluation of Graph-based Machine Learning Models

When conducting empirical evaluations of Machine Learning algorithms, various issues can arise [20]. Such problems have been reported and partially addressed in various subfields, but in the following, we limit the discussion to works in the field of Graph Machine Learning. [26] describe the shortcomings of evaluation strategies for Graph Neural Network models for node classification. [5] focus on graph classification, providing standard practices that should be avoided for a fair comparison. Further, [24] and [28] describe shortcomings in the evaluation of KG link prediction. [11] focus on the evaluation of models for TKG completion (not Forecasting). Our work is the first to study evaluation problems for TKG Forecasting.

3 Description of Evaluation Settings and Evaluation Problems

In this chapter, we subsequently focus on evaluation settings for TKG Forecasting. In each subsection, we first describe a setting, and second, describe problems that we have encountered in that setting. In addition, Table 1, provides an overview, showing the settings each model uses by default. We refer to the respective parts of the table in each subsection. Further, the table contains links to the published code for each model, if available.

3.1 Filter Settings for link prediction metrics

Researchers in TKG Forecasting evaluate the models on metrics known from static link prediction, namely Mean Reciprocal Rank (MRR) and Hits@k, with $k = 1, 3, 10$. There are three settings which have been introduced subsequently, *raw*, *static filter*, and *time-aware filter*:

Raw: As introduced by [2], for each test triple $(s_{test}, r_{test}, o_{test})$, remove the object $(s_{test}, r_{test}, ?)$, and compute the score that the model assigns for each entity $v \in \mathcal{V}$ to be the object in that triple, where the set of all possible triples (s_{test}, r_{test}, v) is termed corrupted triples. Sort the scores in descending order, and note the rank of the correct entity o_{test}. Repeat this by removing the subject $(?, r_{test}, o_{test})$. The MRR is the mean of the reciprocal of these ranks across all queries from the test set, and Hits@k is the proportion of correct entities ranked in the top k.

Static filter: To avoid counting higher ranks from other valid predictions as errors and thus having flaws in the metrics, [1] propose to remove all triples (except the triple of interest) that appear in the train, valid, and test set from the list of corrupted triples.

Time-aware filter: [10] note that the static filter setting is inappropriate for temporal link prediction because it filters out all triples that have ever appeared from the list of corrupted triples, ignoring the time validity of facts. As a consequence, it does not consider predictions of such triples as erroneous. For example, if there is a test query (Barack Obama, visit, India, 2015-01-25) and if the train set contains (Barack Obama, visit, Germany, 2013-01-18), the triple (Barack Obama, visit, Germany) is filtered out for the test query according to the static filter setting, even though it is not true for 2015-01-25 [7]. For this reason, numerous works [7, 21, 8, 27, 17, 18] apply the *time-aware filter* setting which only filters out quadruples with the same timestamp as the test query. In the above example, (Barack Obama, visit, Germany, t) would only be filtered out for the given test query, if it had the timestamp t = 2015-01-25, and otherwise stay in the list of corrupted triples.

Problem 1: Different Filter Settings. The works introduced in Section 2 do present result scores with MRR and Hits@k using the above-described filter settings. However, not all works report results on all filter settings, which is a problem, as it decreases comparability across works. Further, as mentioned above, the raw, and especially the static filter setting are not appropriate for TKG Forecasting. The first part of Table 1 illustrates the filter settings that each model reports.

3.2 Single-step and Multi-step prediction

Methods for Forecasting operate within two distinct prediction settings, single-step and multi-step prediction. Single-step (or one-step) prediction means that the model always forecasts the next timestep [4]. The ground truth facts are then fed before predicting the subsequent timestep. Multi-step prediction means that the model forecasts more than one future time step [4]. More specifically, the model predicts all timesteps from the test set, without seeing any ground truth information in between. As described by [4], multi-step prediction is more challenging, as the model can only leverage information from its own forecasts, and uncertainty accumulates with an increasing number of forecasted timesteps.

Problem 2: Comparison of multi-step and single-step setting. The models described in Section 2 run in different settings. Some can do single-step prediction only, some can do multi-step prediction only, and some do both (see Table 1, second part). Still, single-step models are compared to multi-step models without drawing attention to the different setups. For example, TLogic [21] and TANGO [8] (single-step) are compared to RE-Net [13] (multi-step), xERTE [7] is compared to CyGNet [31], and CEN [17] is compared to CyGNet [31] and RE-Net [13]. The second part of Table 1 shows each model's prediction setting.

3.3 Datasets

Researchers in the domain of TKG Forecasting use the following datasets: Three instances of ICEWS [3]: ICEWS05-15 [6], ICEWS14 [6], and ICEWS18 [12], where the numbers mark the respective years; further, YAGO [22] and WIKI [15], preprocessed according to [12], as well as GDELT [16]. Table 2 shows dataset statistics for dataset version (a), as reported by [19].

Problem 3: Multiple versions of the same dataset. The models described in Section 2 report results on different versions of the same dataset. For instance, three versions exist for ICEWS14. This hinders the comparability of results across works, causing confusion and potential errors. The third part of Table 1 shows an overview of different versions of each dataset, describing each version (marked with (a), (b), (c)) by the number of training triples. One version of the ICEWS14 dataset (see Table 1, version (c)) is especially problematic, as it does not contain a validation set. Instead, the test set is used for both validation and testing. Thus, with this setting, the test set is leaked during training.

3.4 Train, Validation, and Test Set

Researchers in TKG Forecasting split each dataset D into a training D_{train}, validation D_{valid}, and test set D_{test}. The model's training is conducted on D_{train}, not using information contained in D_{valid} or D_{test}. D_{valid} can be used for monitoring the training process, and selecting the best model (parameters) across epochs. There are different options to use the validation set during testing:

(a) The model can leverage all information from D_{train}, but not from D_{valid}, to predict D_{test}. This is consistent with the setting in link prediction for static knowledge graphs.

(b) The model can leverage all information from D_{train} and from D_{valid}, to predict D_{test}. This means, if a model has to answer the query $(s, r, ?, n)$ during testing, all quadruples from D_{train} and D_{valid} can be used. This is consistent with the setting used in time-series Forecasting.

Problem 4: Usage of Validation set for Testing. For multi-step setting, during testing, some models (CygNet, TLogic) do not use the information from the validation set (option (a)), whereas others (RE-GCN, RE-Net) do use it (option (b)), see the fourth part of Table 1. Not using the information from the validation set leads to a significantly harder task, as the model needs to forecast more steps in the future: Instead of starting to predict the next unknown timestep $t+1$ for the first test set sample, the model needs to already predict the timestep $t+num_{valid}+1$, with num_{valid} being the number of timesteps in the validation set, as an information gap between training and testing.

3.5 Problem Summary

When putting all four problems together, a dramatic picture emerges: results have been compared using different filter settings, prediction settings, dataset versions, and dataset splits. Table 1 illustrates the scattered landscape of evaluation settings, where no two models have ever been evaluated on identical settings. Without a uniform and standardized evaluation protocol, we will never be able to gauge true progress in the field. Still, in existing work, the methods are compared to each other, leading to confusion and inconsistencies.

4 A unified evaluation protocol

To tackle the problems introduced in Section 5, it is essential to evaluate TKG models in a consistent way. For this reason, we introduce a unified evaluation protocol with clear and reproducible choices.[3]

Filter settings: We report results on the time-aware filter setting. As explained in Section 3.1, this setting avoids counting higher ranks from other valid predictions as errors while taking into account time validity of facts.

Single-step and Multi-Step: While both settings are valid, the comparison of results for different settings is not fair (see Section 3.2). The setting to be used depends on the use case and on the methods' capabilities. If the method can predict in single- and multi-step, we re-evaluate it on both settings.

Datasets Versions: The same dataset versions should be used across works to ensure comparability. We suggest using version (a) for each dataset (see Table 1). We selected the dataset versions used by the authors of RE-GCN [19], mainly because these are (among) the most commonly used versions across all works. Table 2 shows dataset statistics.

Train, Validation, and Test Set Usage: We use the train, validation, and test sets as described in Section 3.4, option (b), where the information from the validation set can be used for testing, to avoid time gaps between training and testing. In addition, we make sure that the test set is never used for model selection and the datasets are split based on ordered timestamps, whereas one timestamp should not belong to two different sets.

[3] The supplementary material also contains a checklist for benchmark experiments in this field.

Table 1. Methods and their experimental settings: Filter settings (Section 3.1), settings for single- and multi-step prediction (Section 3.2), dataset versions ((a), (b), (c)) used in papers (Section 3.3), and validation set usage (Section 3.4). We report dataset versions by the number of quadruples in the training set. An entry ✓means that the model reported results on the respective setting, and an entry - that it does not. An entry *args* means, that the method provides the option to set this in the args of the code, but does not report the results in the paper. An entry *?* means that we cannot answer this question, as the code is not publicly available.

Name	RE-GCN	RE-Net	xERTE	CyGNet	TLogic	TANGO	Time Traveler	CEN	CluSTeR
Filter settings:									
raw	✓	✓	-	-	-	✓	-	-	✓
static	-	✓	-	✓	-	✓	-	-	-
time-aware	-	-	✓	-	✓	✓	✓	✓	✓
Prediction settings:									
single-step	args	partly[a]	✓	-	✓	✓	✓	✓[b]	?
multi-step	✓	✓	-	✓	args	-	-	-	?
Datasets:									
ICEWS14									
(a): 74845	✓	-	-	-	-	-	-	✓	✓
(b): 63685	-	-	✓	-	✓	-	✓	-	-
(c): 323895 w/o valid[c]	-	✓	-	✓	-	✓	-	-	-
ICEWS18									
(a): 373018	y	✓	✓	✓	✓	✓	✓	✓	✓
ICEWS05-15									
(a): 368868	✓	-	-	-	✓	-	-	-	✓
(b): 322958	-	-	✓	-	-	-	✓	-	-
(c): 369104	-	-	-	-	-	✓	-	-	-
GDELT									
(a): 1734399	✓	✓	✓	-	-	-	-	✓	
YAGO									
(a): 161540	✓	✓	-	✓	-	✓	✓	-	-
(b): 51205	-	-	✓	-	-	-	-	-	-
WIKI									
(a): 539286	✓	✓	-	✓	-	✓	✓	✓	-
Validation Set for Testing:									
Use Valid	✓	✓	✓	-	-	✓	✓	✓	?
Reference	[19]	[13]	[7]	[31]	[21]	[8]	[27]	[17]	[18]
Code Published	✓[d]	✓[e]	✓[f]	✓[g]	✓[h]	✓[i]	✓[j]	✓[k]	-

[a] RE-NET published results for the datasets ICEWS18 and GDELT ([13], Table 2, RE-Net w. GT). The published code does not provide the option to set this in the arguments.
[b] In addition to providing results for single-step setting, CEN has a so-called "online-setting". This means, that the model is re-fit after each test timestep before predicting the next timestep.
[c] This specific version of ICEWS14 comes without validation set. Instead, the test set is used for validation.
[d] `https://github.com/Lee-zix/RE-GCN`
[e] `https://github.com/INK-USC/RE-Net`
[f] `https://github.com/TemporalKGTeam/xERTE`
[g] `https://github.com/CunchaoZ/CyGNet`
[h] `https://github.com/liu-yushan/TLogic`
[i] `https://github.com/TemporalKGTeam/TANGO`
[j] `https://github.com/JHL-HUST/TITer/`
[k] `https://github.com/Lee-zix/CEN`

Table 2. Dataset Statistics for dataset version (a), as reported by [19].

Dataset	#Nodes	#Rels	#Train	#Valid	#Test	Time Interval
ICEWS14	6869	230	74845	8514	7371	24 hours
ICEWS18	23033	256	373018	45995	49545	24 hours
ICEWS0515	10094	251	368868	46302	46159	24 hours
GDELT	7691	240	1734399	238765	305241	15 minutes
YAGO	10623	10	161540	19523	20026	1 year
WIKI	12554	24	539286	67538	63110	1 year

5 Experiments

In the following, we show the results for eight models and five datasets[4]. The supplementary material[5] contains additional information on specific experimental settings. Please find the source code with scripts for experiments and evaluation at `https://github.com/nec-research/TKG-Forecasting-Evaluation`.

We run the experiments on a system with one Nvidia TITAN RTX (24 GB) GPU, 512 GB Memory, and an Intel Xeon Silver 4208 CPU with 16 cores (32 threads).

To eliminate the four problems described in Section 5, we follow the evaluation protocol from Section 4: We report results on time-aware filter settings for single-step and multi-step settings, use the dataset versions (a), and report the results with the validation set usage option (b). We show aggregated results (mean MRR and Hits@k across all test samples) for the eight models for the datasets GDELT, YAGO, WIKI, ICEWS14, and ICEWS18 in Table 3. The upper part for each dataset contains results in multi-step setting, and the lower part in single-step setting, where models with results for single-step prediction should not be benchmarked against methods with results of multi-step prediction. We mark the best result for each dataset for each setting in **bold**. In addition, for the method CEN, we show results in online setting, where the model is updated continually during testing. For completeness and comparability to related work, the supplementary material reports results on raw and static filter settings. In addition, the supplementary material contains tables with information on the reproducibility of the results that have been reported by the original works [7, 27, 13, 19, 8, 21, 31, 17]. Figure 1 shows the MRR for three selected datasets (ICEWS18, WIKI, and GDELT) over test timestamps (snapshots) for different evaluation settings. In the following, we will discuss important insights.

Single-step and Multi-step setting: Table 3 shows the difference in scores for single- vs. multi-step setting: Overall, scores for single-step setting are higher

[4] Because of memory and runtime issues for multiple models due to its large amount of timestamps, and its similarity to the other ICEWS datasets, we excluded the dataset ICEWS05-15. By running the script available in our GitHub repository, interested readers can include this dataset.

[5] Please find the supplementary material at `https://github.com/nec-research/TKG-Forecasting-Evaluation/blob/main/paper_supplementary_material.pdf`.

Table 3. Experimental results for multi-step prediction, single-step prediction, and single-step prediction in online setting (with model updates) with datasets GDELT, YAGO, WIKI (top), and ICEWS14, ICEWS18 (bottom). Results for single-step prediction should not be compared to results for multi-step prediction. We report mean reciprocal rank (MRR), and Hits@k (H@k), with $k = 1, 3, 10$ in time-aware filter setting. The best results for each setting are marked in bold.

multi-step setting (time filter)												
	GDELT				YAGO				WIKI			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
RE-GCN	19.64	12.47	20.85	33.62	**75.40**	**71.75**	**77.67**	81.70	62.72	59.48	64.89	67.87
RE-Net	**19.71**	**12.48**	**20.90**	**33.93**	58.21	53.44	61.31	66.26	49.47	47.21	50.70	53.04
CyGNet	19.08	11.88	20.29	33.07	69.02	61.38	74.29	**83.42**	58.26	52.51	62.41	67.56
TLogic	17.68	11.26	18.90	30.29	66.93	63.14	70.63	71.58	**63.99**	**61.31**	**66.36**	**68.22**
single-step setting (time filter)												
	GDELT				YAGO				WIKI			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
RE-GCN	19.75	12.51	21.02	33.88	82.20	78.72	84.24	88.48	78.65	74.75	81.71	84.68
xERTE	18.89	12.73	21.09	31.96	87.31	84.20	90.28	**91.22**	74.52	70.30	78.58	80.13
TLogic	19.77	12.23	21.67	**35.62**	76.49	74.02	78.91	79.17	**82.29**	**78.62**	**86.04**	**87.01**
TANGO	19.22	12.19	20.42	32.81	62.39	59.04	64.69	67.75	50.08	48.30	51.41	52.76
Timetraveler	**20.23**	**14.14**	**22.18**	31.17	**87.72**	**84.55**	**90.87**	91.20	78.65	75.15	82.03	83.05
CEN	20.43	12.98	21.81	35.04	82.72	78.81	85.24	89.35	79.29	75.51	82.37	84.91
online setting (single-step with model update) (time filter)												
	GDELT				YAGO				WIKI			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
CEN	21.73	13.80	23.51	37.30	83.96	80.08	86.73	90.24	79.82	75.88	83.14	85.47

multi-step setting (time filter)								
	ICEWS14				ICEWS18			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
RE-GCN	**37.82**	**27.86**	**42.14**	**57.50**	**29.03**	**19.52**	**32.66**	**47.50**
RE-Net	37.00	27.80	40.80	54.92	27.86	18.47	31.43	46.19
CyGNet	36.12	26.66	40.28	54.54	26.01	16.69	29.59	44.43
TLogic	35.48	26.54	39.59	53.11	24.01	15.59	27.23	41.20
single-step setting (time filter)								
	ICEWS14				ICEWS18			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
RE-GCN	42.11	31.36	47.33	**62.66**	**32.58**	**22.37**	**36.78**	**52.56**
xERTE	40.91	33.03	45.48	57.07	29.23	20.92	33.50	46.26
TLogic	**42.53**	**33.20**	**47.61**	60.29	29.59	20.42	33.60	48.05
TANGO	36.77	27.29	40.84	55.09	28.35	19.10	31.88	46.27
Timetraveler	40.83	31.90	45.43	57.59	29.13	21.29	32.54	43.92
CEN	41.80	31.85	46.59	60.87	31.50	21.69	35.40	50.69
online setting (single-step with model update) (time filter)								
	ICEWS14				ICEWS18			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
CEN	43.17	33.20	48.03	62.43	31.78	21.82	35.79	51.27

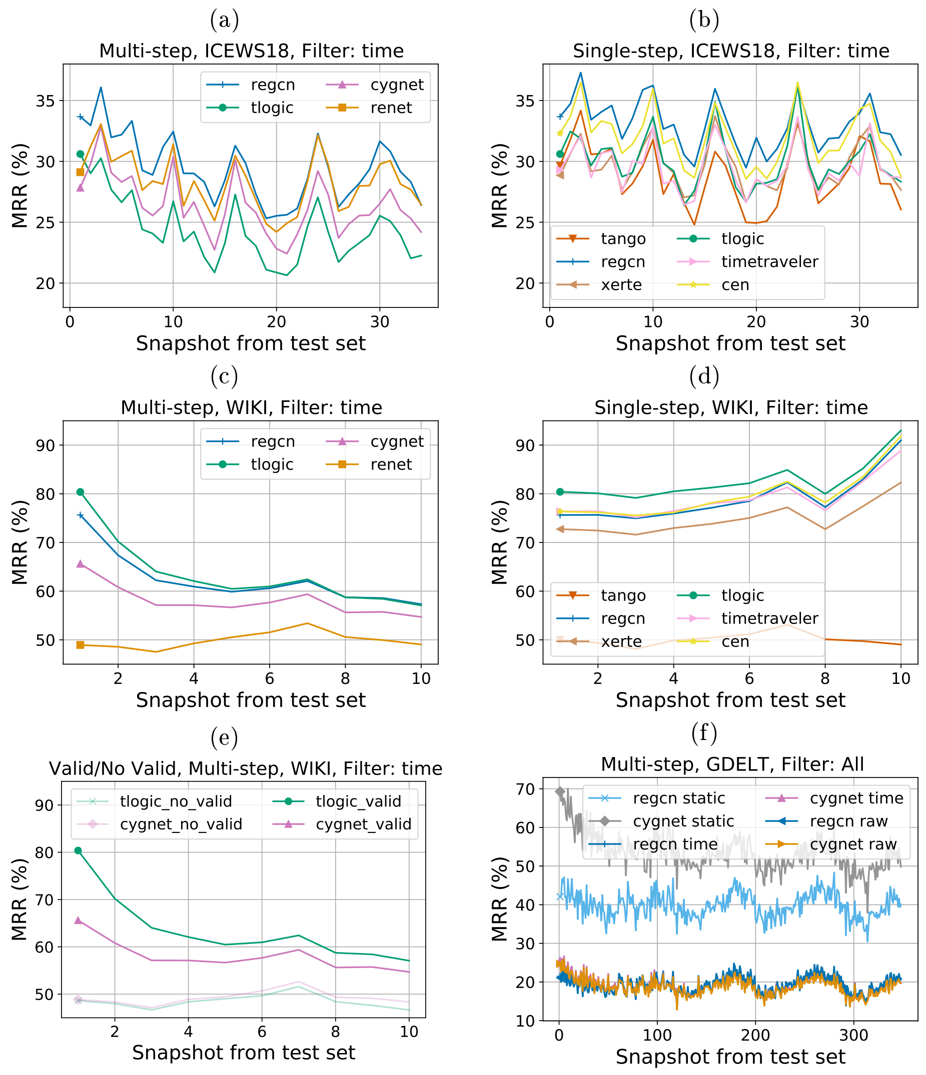

Fig. 1. MRR (in %) over snapshots from test set (one snapshot is one timestamp) per method. (a)-(d): Datasets ICEWS18 (a),(b) and WIKI (c),(d) for multi-step prediction (left) and single-step prediction (right); (e): Using vs. not using the validation set during testing for dataset WIKI; (f) Different filter settings for dataset GDELT.

than for multi-step setting. This is especially visible for the two models (TLogic and RE-GCN) that run in both settings, but also true for the other results. Figure 1(a) - (d) shows the MRR (in %) over snapshots in multi-step setting (left) and single-step setting (right).[6] The figure illustrates a contrasting trend

[6] The supplementary material shows results for ICEWS14, YAGO, and GDELT.

between multi-step prediction and single-step prediction with respect to MRR. Specifically, the MRR for multi-step prediction exhibits a decreasing pattern as the timestamps increase, whereas single-step prediction does not display a similar decreasing trend. This is especially visible for the WIKI dataset in a single-step setting, which displays an increasing tendency for the MRR with increasing timestamps for the four best-performing methods. The results reflect the statement from Section 3.2, that multi-step prediction is more challenging, and uncertainty accumulates with increasing number of forecasted timesteps, as the models can only leverage information from their own forecasts. Thus, benchmarking models for multi-step prediction against single-step prediction is only fair for the first timestamp.

Validation Set Usage: In Figure 1(e), we show the MRR (in %) over snapshots in multi-step setting for TLogic and CyGNet[7], when using the validation set for testing (Section 3.4, option (b)) vs. not using the validation set for testing (option (a)) for the dataset WIKI.[8] The figure displays a difference in MRR between the two settings for each model, especially in the first two snapshots with a difference in MRR of > 30 for TLogic. This difference is caused by the information gap between the last training timestamp and the first testing timestamp. For the case of WIKI, the number of timestamps in the validation set is $num_{valid} = 11$. The difference decreases with increasing timestamps, because, due to the multi-step setting, there is also a rising information gap when feeding the validation set. Thus, using the information from the validation set for testing and avoiding the information gap is crucial for fair comparison among models.

Filter Settings: Figure 1(f) shows the MRR (in %) over snapshots in multi-step setting, exemplary for CyGNet and RE-GCN for the dataset GDELT, computed with raw, static, and time-aware filter setting, as described in Section 3.1[9]. It reveals a large difference in MRR for static filter setting, vs. raw setting or time-aware filter setting, especially for CyGNet. This is also visible for aggregated results: Where CyGNet does not have the highest MRR scores on any dataset for time-aware filter settings (see Table 3), it has the highest MRR scores on all five datasets in static filter setting (see supplementary material). The static filter setting filters out all triples that have ever appeared from the corrupted triples, ignoring the time validity, and does not count a prediction of these triples as error. Thus, for a given query, if a model predicts entities that have appeared in this triple at an earlier timestep, this will not be considered erroneous, even if the predicted fact is not true in the timestep of question. The model will potentially be assigned a higher static filter score than if it would predict previously unseen facts. Thus, the static filter setting favors models that predict repeated facts.

[7] The two models that run per default in multi-step setting, validation set option (a) from Section 3.4.

[8] The supplementary material shows results for YAGO, GDELT, ICEWS14, and ICEWS18.

[9] The supplementary material shows results for YAGO, WIKI, ICEWS14, and ICEWS18.

To summarize, we can see that no model shows the best results across all datasets. This evidence remarks the importance of fairly comparing models on different benchmarks. We stressed the clear differences in result scores for single-step and multi-step prediction. In addition, we pointed out that the usage of the validation set during testing does lead to substantially higher test scores. Further, we showed the significant influence of the filter setting used for score computation.

Comparing Results of Original Papers and This Work: It is not straightforward to compare the results from this study with the results reported in the original papers, when it comes to assessing the state-of-the-art method due to several reasons. Firstly, there exist variations in the evaluation settings and inconsistencies in the evaluations across different methods, as elaborated in Section . Secondly, the original papers lack complete comparisons between all methods, due to varying factors such as earlier or parallel publication times or results reported only on subsets of datasets.

To illustrate the impact of our proposed evaluation protocol on the ranking of compared methods, we show an example for CyGNet. The original paper reports higher MRRs for CyGNet compared to RE-Net on the datasets ICEWS14, ICEWS18, and GDELT, while lower MRRs on the datasets YAGO and WIKI. However, when employing our evaluation protocol, CyGNet achieves higher MRRs than RE-Net on YAGO and WIKI, but lower MRRs on all other datasets. A plausible explanation for this disparity is the utilization of different filter settings which, as highlighted in the preceding paragraph, notably influences the obtained scores.

6 Conclusion

Summary: In this work, we examined the evaluation of TKG Forecasting models. We uncovered and described inconsistencies that strongly influence the experimental results and thus lead to distorted comparisons among models. To address these problems, we formed a unified evaluation protocol from reasonable evaluation settings and re-evaluated state-of-the-art methods. We illustrated the importance of a consistent evaluation by showing the effect of different evaluation settings on the results. Our work aims at establishing a unified evaluation protocol, stimulating discussions on the evaluation, and raising the community's awareness of experimental issues, with the goal of advancing the research field of TKG Forecasting.

Limitation of this study: Due to computational infeasibility, we could not conduct multiple repeats for each experiment run[10]. Even with one repetition per run, we experienced significant computation times for many models, e.g., multiple days to weeks for the dataset GDELT; thus, multiple repetitions per model and dataset were not possible. Adding multiple repetitions to the eval-

[10] One experiment run: A one time training of a model with a given setting on a specific dataset.

uation would have further improved the robustness of our results, which are nonetheless obtained under a unified and reproducible protocol.

Future Work: In future work, we aim to extend the proposed evaluation protocol to: First, evaluate the full predicted graph for methods that can predict full graphs (e.g., RE-Net), instead of exclusively focusing on link prediction. This could be based on graph similarity or computing a percentage of correctly predicted triples. Second, evaluate the change of the predicted graph snapshots over time to analyze if the predictions evolve and if they are able to capture time information. This could be done by comparing the predictions at different time steps. Third, include more fine-grained evaluation to answer what properties the models learned and what they did not. This could, for example, be done using the framework KGxBoard [30], which breaks down the performance measure over individual data subsets.

Acknowledgements We warmly thank Federico Errica for his time and very valuable feedback.

Ethical Statement

While TKG Forecasting has the potential to enable predictions for complex and dynamic systems, we argue that inconsistencies in experimental procedures and evaluation settings can lead to distorted comparisons among models, and ultimately, misinterpretation of results. Therefore, with our work, we want to highlight the importance of transparency and reproducibility in scientific research, as well as the importance of rigorous and reliable scientific practice. In this context we have identified inconsistencies in evaluation settings and provided a unified evaluation protocol. We ensure transparency by providing a URL to a GitHub repository containing our evaluation code. Within this repository, we use forked submodules to explicitly link to the original assets. Additionally, we report the training details, such as hyperparameters, in the supplementary material of our work.

While we have not focused on increasing the interpretability of individual models, we acknowledge the importance of explainability and interpretability in the field. Therefore, we note that among the compared models, xERTE [7] and TLogic [21] address some aspects of explainability and interpretability.

We did not evaluate the predictions of existing models on bias and fairness as it was out of scope for this work. However, we recognize that it is essential to increase fairness in the comparison of TKG Forecasting models. Therefore, we highlight inconsistencies and provide a unified evaluation protocol to improve comparability and fairness for existing models.

In terms of data collection and use, we used publicly available research datasets for our evaluation. We did not use the data for profiling individuals, and it does not contain offensive content. However, it is important to note that even publicly available data can be subject to privacy regulations, and we have taken measures to ensure that our data usage complies with applicable laws and regulations.

As this study focuses purely on evaluation of existing models, it does not induce direct risk. However, we recognize that TKG Forecasting models can have real-world consequences, especially when applied in domains such as finance and healthcare. Therefore, as the results in Section 5 show, we want to stress again that predictions can be unreliable and incomplete, and that these limitations have to be acknowledged when using them for decision making.

Bibliography

[1] Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Burges, C.J.C., Bottou, L., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States. pp. 2787–2795 (2013)

[2] Bordes, A., Weston, J., Collobert, R., Bengio, Y.: Learning structured embeddings of knowledge bases. In: Burgard, W., Roth, D. (eds.) Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011. AAAI Press (2011)

[3] Boschee, E., Lautenschlager, J., O'Brien, S., Shellman, S., Starz, J., Ward, M.: ICEWS Coded Event Data (2015)

[4] Brownlee, J.: Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python. Machine Learning Mastery (2018)

[5] Errica, F., Podda, M., Bacciu, D., Micheli, A.: A fair comparison of graph neural networks for graph classification. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020 (2020)

[6] García-Durán, A., Dumančić, S., Niepert, M.: Learning sequence encoders for temporal knowledge graph completion. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 4816–4821. Association for Computational Linguistics, Brussels, Belgium (Oct-Nov 2018)

[7] Han, Z., Chen, P., Ma, Y., Tresp, V.: Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021 (2021)

[8] Han, Z., Ding, Z., Ma, Y., Gu, Y., Tresp, V.: Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021. pp. 8352–8364. Association for Computational Linguistics (2021)

[9] Han, Z., Ma, Y., Wang, Y., Günnemann, S., Tresp, V.: Graph hawkes neural network for forecasting on temporal knowledge graphs. In: Das, D., Hajishirzi, H., McCallum, A., Singh, S. (eds.) Conference on Automated Knowledge Base Construction, AKBC 2020, Virtual, June 22-24, 2020 (2020)

[10] Han, Z., Ma, Y., Wang, Y., Günnemann, S., Tresp, V.: Graph hawkes neural network for forecasting on temporal knowledge graphs. In: Das,

D., Hajishirzi, H., McCallum, A., Singh, S. (eds.) Conference on Automated Knowledge Base Construction, AKBC 2020, Virtual, June 22-24, 2020 (2020)

[11] Han, Z., Zhang, G., Ma, Y., Tresp, V.: Time-dependent entity embedding is not all you need: A re-evaluation of temporal knowledge graph completion models under a unified framework. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021. pp. 8104–8118. Association for Computational Linguistics (2021)

[12] Jin, W., Qu, M., Jin, X., Ren, X.: Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. arXiv preprint arXiv:1904.05530 (2019), preprint version

[13] Jin, W., Qu, M., Jin, X., Ren, X.: Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In: Webber, B., Cohn, T., He, Y., Liu, Y. (eds.) Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020. pp. 6669–6683. Association for Computational Linguistics (2020)

[14] Kotz, S., Balakrishnan, N., Johnson, N.L.: Continuous Multivariate Distributions. Volume 1: Models and Applications. Wiley, New York (2000)

[15] Leblay, J., Chekol, M.W.: Deriving validity time in knowledge graph. In: Champin, P., Gandon, F., Lalmas, M., Ipeirotis, P.G. (eds.) Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon, France, April 23-27, 2018. pp. 1771–1776. ACM (2018)

[16] Leetaru, K., Schrodt, P.A.: Gdelt: Global data on events, location, and tone, 1979–2012. In: ISA annual convention. pp. 1–49. Citeseer (2013)

[17] Li, Z., Guan, S., Jin, X., Peng, W., Lyu, Y., Zhu, Y., Bai, L., Li, W., Guo, J., Cheng, X.: Complex evolutional pattern learning for temporal knowledge graph reasoning. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 290–296. Association for Computational Linguistics, Dublin, Ireland (May 2022)

[18] Li, Z., Jin, X., Guan, S., Li, W., Guo, J., Wang, Y., Cheng, X.: Search from history and reason for future: Two-stage reasoning on temporal knowledge graphs. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021. pp. 4732–4743. Association for Computational Linguistics (2021)

[19] Li, Z., Jin, X., Li, W., Guan, S., Guo, J., Shen, H., Wang, Y., Cheng, X.: Temporal knowledge graph reasoning based on evolutional representation learning. In: Diaz, F., Shah, C., Suel, T., Castells, P., Jones, R., Sakai, T. (eds.) SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021. pp. 408–417. ACM (2021)

[20] Liao, T., Taori, R., Raji, I.D., Schmidt, L.: Are we learning yet? a meta review of evaluation failures across machine learning. In: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2) (2021)

[21] Liu, Y., Ma, Y., Hildebrandt, M., Joblin, M., Tresp, V.: Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022. pp. 4120–4127. AAAI Press (2022)

[22] Mahdisoltani, F., Biega, J.A., Suchanek, F.M.: Yago3: A knowledge base from multilingual wikipedias. In: CIDR (2015)

[23] Micheli, A.: Neural network for graphs: A contextual constructive approach. IEEE Trans. Neural Networks **20**(3), 498–511 (2009)

[24] Rossi, A., Barbosa, D., Firmani, D., Matinata, A., Merialdo, P.: Knowledge graph embedding for link prediction: A comparative analysis. ACM Trans. Knowl. Discov. Data **15**(2), 14:1–14:49 (2021)

[25] Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Networks **20**(1), 61–80 (2009)

[26] Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S.: Pitfalls of graph neural network evaluation. In: Relational Representation Learning Workshop (R2L 2018), NeurIPS, Montréal, Canada (2018)

[27] Sun, H., Zhong, J., Ma, Y., Han, Z., He, K.: Timetraveler: Reinforcement learning for temporal knowledge graph forecasting. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021. pp. 8306–8319. Association for Computational Linguistics (2021)

[28] Sun, Z., Vashishth, S., Sanyal, S., Talukdar, P.P., Yang, Y.: A re-evaluation of knowledge graph completion methods. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020. pp. 5516–5522. Association for Computational Linguistics (2020)

[29] Trivedi, R., Dai, H., Wang, Y., Song, L.: Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. Proceedings of Machine Learning Research, vol. 70, pp. 3462–3471. PMLR (2017)

[30] Widjaja, H., Gashteovski, K., Ben Rim, W., Liu, P., Malon, C., Ruffinelli, D., Lawrence, C., Neubig, G.: KGxBoard: Explainable and interactive leaderboard for evaluation of knowledge graph completion models. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. pp. 338–350. Association for Computational Linguistics, Abu Dhabi, UAE (Dec 2022)

[31] Zhu, C., Chen, M., Fan, C., Cheng, G., Zhang, Y.: Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021. pp. 4732–4740. AAAI Press (2021)