SeDyT: A General Framework for Multi-Step Event Forecasting via Sequence Modeling on Dynamic Entity Embeddings

Hongkuan Zhou
James Orme-Rogers
{hongkuaz,ormeroge}@usc.edu
University of Southern California
Los Angeles, California, USA

Rajgopal Kannan
rajgopal.kannan.civ@mail.mil
US Army Research Lab
Los Angeles, California, USA

Viktor Prasanna
prasanna@usc.edu
University of Southern California
Los Angeles, California, USA

ABSTRACT

Temporal Knowledge Graphs store events in the form of subjects, relations, objects, and timestamps which are often represented by dynamic heterogeneous graphs. Event forecasting is a critical and challenging task in Temporal Knowledge Graph reasoning that predicts the subject or object of an event in the future. To obtain temporal embeddings multi-step away in the future, existing methods learn generative models that capture the joint distribution of the observed events. To reduce the high computation costs, these methods rely on unrealistic assumptions of independence and approximations in training and inference. In this work, we propose SeDyT, a discriminative framework that performs sequence modeling on the dynamic entity embeddings to solve the multi-step event forecasting problem. SeDyT consists of two components: a Temporal Graph Neural Network that generates dynamic entity embeddings in the past and a sequence model that predicts the entity embeddings in the future. Compared with the generative models, SeDyT does not rely on any heuristic-based probability model and has low computation complexity in both training and inference. SeDyT is compatible with most Temporal Graph Neural Networks and sequence models. We also design an efficient training method that trains the two components in one gradient descent propagation. We evaluate the performance of SeDyT on five popular datasets. By combining temporal Graph Neural Network models and sequence models, SeDyT achieves an average of 2.4% MRR improvement when not using the validation set and more than 10% MRR improvement when using the validation set.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Temporal reasoning**.

KEYWORDS

event prediction; graph neural networks

ACM Reference Format:
Hongkuan Zhou, James Orme-Rogers, Rajgopal Kannan, and Viktor Prasanna. 2021. SeDyT: A General Framework for Multi-Step Event Forecasting via Sequence Modeling on Dynamic Entity Embeddings. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3459637.3482177

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11.
https://doi.org/10.1145/3459637.3482177

1 INTRODUCTION

Knowledge Graphs (KGs) store real-world facts in the form of events while Temporal Knowledge Graphs (TKGs) store events with time information. Event prediction is an important task in TKG reasoning that discovers missing facts in TKGs. Event forecasting is a more challenging and more impactful task that predicts the events multi-step (timestamp) away in the future and can be used in many real-world problems like economic crisis prediction [2], crime prediction [8], and epidemic modeling [19]. In a TKG, events are represented by quadruplets (s, r, o, t) that indicate subjects, relations, objects, and timestamps. An event that is valid for a period of time is decomposed into multiple events with the same s, r, and o at different timestamps t. The multi-step event forecasting task is defined by predicting the events in the future ($t > T$) giving the event quadruplets happen in the past ($t \leq T$). As forecasting the future events with unseen subjects, objects, or relations is extremely hard, researchers have narrowed the space to identify the subjects, objects, or relations of future events given the rest of the elements in the ground truth quadruplets. In this work, we follow the most popular setup and focus on predicting the subject or object of the events in the future by answering the queries $(?, r, o, t)$ and $(s, r, ?, t)$, which is equivalent to a multi-class classification problem with the number of candidate classes equals to the total number of entities.

Recently, Graph Neural Networks (GNNs) have demonstrated strong expressive power and high versatility in graph representation learning. To learn temporal information as well as structural and contextual information, Temporal GNNs [6, 17] generate dynamic embeddings by adding time encodings to the node attributes and applying Recurrent Neural Networks (RNNs) to regulate the embeddings or the weights. In the event forecasting problem, the quadruplet representation of TKG seamlessly forms a dynamic heterogeneous graph where the entities (subjects and objects) become the nodes and relations become the heterogeneous edges. The timestamps are attached to their corresponding edges. However, it is hard to directly apply Temporal GNNs to forecast future events on the dynamic heterogeneous graph due to the lack of future graph structure information. Inference and reusing the future graph structure step-by-step amplify the prediction error in the temporal embeddings, which leads to low accuracy. To address this issue, previous works [9, 11] used generative models to fit the events into pre-defined heuristic-based probability systems. The future

entity embeddings multi-step away are generated by integrating the probability model over all the future timestamps. Compared with discriminative models, generative models typically have high computation costs and favor high-quality training data. The performance also profoundly depends on the design of the probability system. In this work, we propose SeDyT, a *discriminative framework* that applies sequence modeling on the extracted dynamic embeddings to perform multi-step forecasting. The auxiliary sequence model overcomes the challenge of multi-step forecasting using Temporal GNN, allowing SeDyT to be trained in a simple discriminative manner. The main contributions of this work are:

- We propose SeDyT, a highly flexible discriminative framework for event forecasting on TKGs which is compatible with most sequence models and Temporal GNNs.
- We develop an efficient mini-batch training method that updates the parameters in the sequence model and the Temporal GNN in SeDyT in one gradient descent propagation. The co-training of the two components allows SeDyT to accurately learn the probability of each entity in the queries without any approximations or assumptions.
- We evaluate the performance of SeDyT on five popular datasets under two different setups. Compared with the state-of-the-art baselines, SeDyT achieves the highest MRR on all five datasets.

2 BACKGROUND

There are many works [7, 22, 25, 30] on event prediction in TKGs with the task of predicting missing elements in the events. Recently, Re-NET [11] claimed to be the first to provide solution to the event forecasting task where only the ground truth in the training set is used at inference. Re-NET implemented a static GNN and several RNNs to capture the dynamic entity embeddings and mapped the embeddings to an auto-regressive probability model. Later, NLSM [9] proposed a variational inference technique to estimate the posterior distribution of an edge occurring between two specified nodes. In order to capture the joint distribution with low computation cost, these models have to stand on unpractical assumptions, such as the independence between two graph snapshots with a long time duration, two events happening at the same timestamp, and the statistical model parameters. To perform multi-step inference, these generative models need to integrate over the future steps which are also not computationally feasible. As a result, Re-NET samples in the probability space and NLSM approximates the posterior distribution using Evidence Lower Bound Objective (ELBO). By contrast, SeDyT does not need these assumptions and approximations in both training and inference. To guarantee accuracy, the probability of the target entity is directly computed by our discriminative model. Beside these generative models, CyGNet [33] tried to attach a copy module to predict the future from the history, which achieved state-of-the-art accuracy on some datasets.

2.1 Temporal Heterogeneous Graph Neural Network

GNN [13] and its variants [1, 27, 31, 32] generate node embeddings by iteratively gathering and aggregating neighbor information.

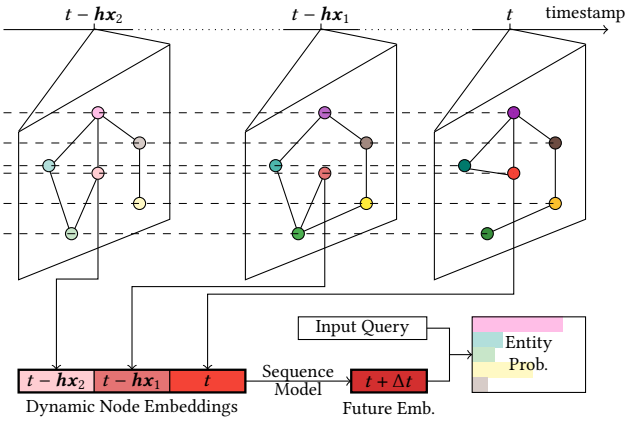

Figure 1: Illustration of SeDyT forecasting an event Δt-steps away in the future with selected dynamic history embeddings $t - hx$. The solid edges in each timestamp and the dashed lines connecting the entity at different timestamps represent the heterogeneous edges of the dynamic graph.

The forward propagation in the $(k)^{\text{th}}$ GNN layer with attention mechanism is defined as

$$\boldsymbol{h}_v^{(k)} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \frac{\exp(\boldsymbol{Q}_v^{(k-1)} \cdot \boldsymbol{K}_u^{(k-1)})}{\sum_{u' \in \mathcal{N}(v)} \exp(\boldsymbol{Q}_v^{(k-1)} \cdot \boldsymbol{K}_{u'}^{(k-1)})} \boldsymbol{V}_u^{(k-1)}\right) \quad (1)$$

where $\boldsymbol{Q}_v^{(k-1)} = \boldsymbol{W}_q^{(k-1)} \boldsymbol{h}_v^{(k-1)}$ is the query, $\boldsymbol{K}_v^{(k-1)} = \boldsymbol{W}_k^{(k-1)} \boldsymbol{h}_v^{(k-1)}$ is the key, and $\boldsymbol{V}_v^{(k-1)} = \boldsymbol{W}_v^{(k-1)} \boldsymbol{h}_v^{(k-1)}$ is the value. $\sigma(\cdot)$ denotes the activation function. $\mathcal{N}(v)$ denotes the set of one-hop neighbors of node v.

To capture temporal information as well as structural and contextual information, recent works [6, 21, 29] attached a functional time encoding that captures the time difference between the current time and the timestamp of the edge to traverse. Specifically, the neighbor messages $\boldsymbol{h}_u^{(k-1)}$ in Equation 1 are concatenated with an additional functional time encoding

$$\bar{\boldsymbol{h}}_u^{(k-1)} = \boldsymbol{h}_u^{(k-1)} || \cos(\boldsymbol{\omega}(t - t_{vu}) + \boldsymbol{\phi}) \quad (2)$$

where t_{vu} is the timestamp of the edge from node v to node u. $\boldsymbol{\omega}$ and $\boldsymbol{\phi}$ are two learnable vectors.

2.2 Sequence Model

Sequence models aim to predict the next or next several elements in a given sequence. Recurrent Neural Network (RNN) [20] and its variants [5, 10] capture the temporal dependency in the sequences by using the output in the previous cell as the input of the current cell. Transformers [26] have proven to be powerful models in language sequences that rely purely on attention mechanisms. Convolutional Neural Networks (CNNs) and Multi-Layer Perceptrons (MLPs) can also be used in sequence modeling if the sequence length is fixed.

3 APPROACH

Figure 1 shows a simplified diagram of our SeDyT framework. To answer the query $(?, r, o, t + \Delta t)$ or $(s, r, ?, t + \Delta t)$, SeDyT first computes the dynamic node embeddings of the input entity o or s at each of the history and current timestamps. Then, a sequence model generates the entity embeddings at the target timestamp $t+\Delta$. Finally, the classifier generates the probability of each candidate entity. We introduce these three processes in Section 3.1, Section 3.2, and Section 3.3, respectively. To efficiently train SeDyT in one gradient pass, we also design a novel training method, which is presented in Section 3.4.

3.1 Dynamic Node Embedding Generation

The dynamic node embeddings are generated by performing graph representation learning on the dynamic heterogeneous knowledge graph. In the knowledge graph, temporal heterogeneous edges represent events while static homogeneous nodes represent entities. For an event (s, r, o, t), we add two edges to the knowledge graph: one r-typed edge from node s to node o with timestamp t and one $\overline{r}$-typed edge from node o to node s, which enables the entity nodes to gather information from both their corresponding objects and subjects. We use learnable static embeddings to serve as the input node attributes, which could also be replaced with entity features (if present in the dataset).

We adopt the multi-head attention [27] mechanism that performs multiple forward propagation in Equation 1 with separate weight matrices $\boldsymbol{W}_q$, $\boldsymbol{W}_k$, and $\boldsymbol{W}_v$ in each attention head. Since the knowledge graph has heterogeneous edges, we use separate weight matrices for each edge type. Let $r\boldsymbol{h}_v^{(k)}$ be the aggregated message of node v from its neighbors $\mathcal{N}_r(v)$ with r-typed edges. We use the mean aggregated message of all connected edge types $\|\mathcal{N}(v)\|$ as the hidden feature of node v.

To capture the temporal information in the edges, we replace $\boldsymbol{h}$ with $\overline{\boldsymbol{h}}$ in Equation 2 except for the hidden features in the last layer. Thus, the generated embeddings contain the contextual information from node attributes as well as the structural and temporal information from temporal heterogeneous edges. Note that SeDyT is also compatible with traditional GNN+RNN methods [11, 17] to generate node embeddings.

3.2 Sequence Modeling

SeDyT uses the future entity embeddings to forecast events. Let t be the current time. The future entity embedding $\boldsymbol{h}_v(t+\Delta t)$ of node v at timestamp $t + \Delta t$ is computed by performing fixed-length sequence modeling on the sequence of the current and past entity embeddings $[\boldsymbol{h}_v(t - \boldsymbol{hx})]$ where $\boldsymbol{hx}$ denotes the selected history timestamps $\left[\boldsymbol{hx}_{|\boldsymbol{hx}|}, \cdots, \boldsymbol{hx}_1, \boldsymbol{hx}_0 = 0\right]$. Sequence modeling is a well-studied problem, especially for fixed-length sequences. In this work, we focus on the following four sequence models.

- SATT: Sequence modeling based on self-attention [26] (encoder only). In each SATT layer, we compute the attention between each pair of elements in the entity embedding sequence.
- CONV: Sequence modeling based on the convolutional operation. We stack the sequences vertically to obtain 2-D tensors and apply a 3×3 2-D convolution kernel.
- MLP: Sequence modeling based on MLP. We concatenate the sequence horizontally and apply MLP to obtain the predicted future entity embedding.
- LSTM: Sequence modeling based on LSTM [10]. We use a multi-to-one LSTM to predict the future entity embedding.

3.3 Probability Generation

To predict the subject or object of a future event, we take the future embedding of the corresponding object or subject and the learnable static relation embedding as input. We use MLP with residual connections to map the input to the probability of the candidate embeddings.

3.4 Efficient Training

To train SeDyT in a discriminative way, we set the step Δt in the sequence model to be the time span of the validation and test set so that SeDyT can forecast all the events in the test set with the ground truth in the training set only. As a discriminative framework, SeDyT does not set a loss for the sequence model to predict the known history embeddings of the entity. Instead, we optimize the cross-entropy loss only between the generated probabilities and the ground truth entities for each event in the training set. However, directly performing gradient descent on this loss is infeasible because it must propagate on the graph structure which is different at the history timestamps $\boldsymbol{hx}$. To solve this problem, we create snapshots of window size T by duplicating the nodes and adding a special type of edge between the same entity across the snapshots to retain connectivity as shown in Figure 1. These self-connection edges are directed edges that only point from the past nodes to their future nodes, which avoids using the information from the future. This allows the Temporal GNN to compute forward and backward propagation at different timestamps simultaneously by computing the node embeddings of the same entity in different snapshots.

4 EXPERIMENTS

We evaluated the performance of SeDyT on five popular datasets: GDELT [15], ICEWS14 [25], ICEWS18 [4], WIKI [14], and YAGO [16]. We follow the same training, validation, and test split used in the event forecasting works [9, 11, 33]. On ICEWS14, we ignore the timestamps of test events and set $\Delta t = 1$ because the time duration in the test set is longer than the training set. We set 200 as the dimensions of the entity attributes and relation embeddings to be consistent with other works [11, 33]. In the Temporal GNN, we use a one-layer GNN described in Section 3.1 to generate 200-dimensional temporal entity embeddings. We combine the heterogeneous edge types with low occurrences and use shared weights to perform message passing on these edges. The window size T is set to be 4, 3, 3, 1, 1 on the five datasets, respectively. We show the results of SeDyT with four different sequence models mentioned in Section 3.2 with the history timestamps $\boldsymbol{hx} = [31, 23, 15, 7, 3, 1, 0]$. We adopt the copy module [33] which takes the temporal entity embeddings at the last timestamp $\boldsymbol{hx}_0 = 0$ as input and is disconnected in the gradient descent step. We use the ADAM optimizer [12] to train

Table 1: MRR, Hits@3, and Hit@10 results of event forecasting on the five datasets using the ground truth in the training set.

Method	GDELT			ICEWS14			ICEWS18			WIKI			YAGO		
	MRR	H@3	H@10	MRR	H@3	H@10	MRR	H@3	H@10	MRR	H@3	H@10	MRR	H@3	H@10
R-GCN	23.31	24.94	34.36	26.31	30.43	45.34	23.19	25.34	36.48	37.57	39.66	41.90	41.30	44.44	52.68
RotatE	22.33	23.89	32.29	29.56	32.92	42.68	23.10	27.61	38.72	48.67	49.74	49.88	64.09	64.67	66.16
HyTE	6.37	6.72	18.63	11.48	13.04	22.51	7.31	7.50	14.95	43.02	45.12	49.49	23.16	45.74	51.94
EvolveRGCN	15.55	19.23	31.54	17.01	18.97	32.58	16.59	18.32	34.01	46.49	47.83	49.23	59.74	61.03	61.69
R-GCRN+MLP	37.29	41.08	51.88	36.77	40.15	52.33	35.12	38.26	50.49	47.71	48.14	49.66	53.89	56.06	61.19
RE-NET	40.42	43.40	53.70	45.71	49.06	59.12	42.93	45.47	**55.80**	51.97	52.07	53.91	65.16	65.63	68.08
CyGNet	50.22	53.26	57.42	48.41	52.23	**59.82**	45.82	**48.62**	55.14	45.23	50.81	52.12	64.42	65.02	67.59
SeDyT-SATT	54.96	54.79	58.30	**52.76**	**52.89**	57.42	45.91	45.89	49.62	52.73	52.82	53.65	66.17	66.41	68.39
SeDyT-CONV	54.86	54.68	58.14	52.72	52.86	57.43	45.91	45.86	49.54	**52.90**	**52.96**	**54.00**	**66.88**	**67.05**	**68.73**
SeDyT-MLP	**54.99**	**54.82**	**58.37**	52.61	52.72	57.37	46.01	45.98	49.74	52.84	52.95	53.89	66.17	66.41	68.39
SeDyT-LSTM	54.86	54.62	58.10	52.71	52.88	57.48	**46.04**	45.97	49.83	52.46	52.51	53.40	65.86	66.22	67.94

SeDyT till converge. We implement SeDyT using DGL [28] and PyTorch. All the results are averages of three runs. Our code is available at https://github.com/tedzhouhk/SeDyT.

We first compare the performance of SeDyT with the static methods [22, 24] and the temporal methods [7, 11, 18, 23, 33] under the setting of forecasting future events using the events in the training set only (i.e., Δt equals to the time duration of the validation and test set). We modify the code of CyGNet so that both the subject and the object are predicted using a single model. Table 1 shows the filtered [3] MRR, Hit@3, and Hit@10 (in percentile) of SeDyT and the aforementioned baselines. The results of baselines except for CyGNet are taken from RE-NET [11]. SeDyT achieves the highest MRR on all datasets with an average improvement of 2.40% over the state-of-the-art results. SeDyT also achieves the highest Hit@3 on all datasets and the highest Hit@10 on three datasets. The four variants of SeDyT achieves comparable performance, where SeDyT-CONV works best on the TKGs with the prolonged events (WIKI and YAGO). For the TKGs with point time events (GDELT, ICEWS14, and ICEWS18), SeDyT-MLP achieves best or close to best performance.

The setting of only using the ground truth in the training set at inference misspends the most up-to-date observed events that are in the validation set. To make use of the validation set at inference, we compare the performance of SeDyT with RE-NET [11] and NLSM [9] under the setting of forecasting future events using the events in the training and validation set (i.e., Δt equals to the time duration of the test set). The results of the baselines are taken from NLSM [9]. With the additional ground truth in the validation set, SeDyT forecasts the future events with shorter Δt, which greatly boosts the performance. SeDyT-CONV and SeDyT-MLP achieve the best performance on these two datasets with more than 10% improvement in MRR compared with NLSM.

Table 2: MRR, Hits@3, and Hit@10 results of event forecasting on WIKI and YAGO using the ground truth in the training and validation sets.

Method	WIKI			YAGO		
	MRR	H@3	H@10	MRR	H@3	H@10
RE-NET	53.57	54.10	55.72	66.80	67.23	69.77
NLSM	56.70	57.80	61.10	69.40	71.25	73.90
SeDyT-SATT	68.90	68.84	69.21	84.47	84.41	84.73
SeDyT-CONV	68.97	68.92	69.31	**84.48**	**84.39**	84.72
SeDyT-MLP	**68.98**	**68.94**	**69.34**	**84.48**	84.38	**84.87**
SeDyT-LSTM	68.95	68.91	69.28	84.45	84.38	84.76

5 CONCLUSION AND FUTURE WORK

In this work, we proposed SeDyT, a discriminative framework for event forecasting on TKGs that is compatible with most Temporal GNNs and sequence models. We developed a snapshot-based TKG representation and an efficient training algorithm that trains the two components in SeDyT in one gradient pass. We evaluated the performance of four SeDyT variants on five datasets and showed that SeDyT achieved the highest MRR on all datasets.

In the future, we want to test the performance of SeDyT on large-scale datasets such as the GDELT dataset with multiple years instead of the one-month used in this work. As SeDyT is compatible with most Temporal GNNs, we would like to try more complex and powerful methods. In addition, under the current setting, SeDyT forecasts the future events with a fixed step Δt where the ground truth events in the last several timestamps are not used when forecasting the events at the beginning several timestamps in the future. For example, when forecasting an event at timestamp $t+1$, SeDyT with fixed step Δt ignores the ground truth events between $t-\Delta t+1$ to t. We plan to explore the sequence models that support adjustable sequence lengths like SATT and LSTM to support dynamic step prediction for SeDyT. Dynamic-step prediction also has the potential to improve the computation complexity when forecasting future events in a continuous-time interval.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation (NSF) under grants OAC-1911229 and CNS-2009057 and in part by the Army Research Lab (ARL) under ARL-USC collaborative grant DIRA-ECI:DEC21-CI-037.

REFERENCES

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. 2019. MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing *(Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). 21–29.

[2] Gijs Beets and Frans Willekens. 2009. The global economic crisis and international migration: An uncertain outlook. *Vienna Yearbook of Population Research* (2009), 19–37.

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf

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

[5] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014). https://doi.org/10.3115/v1/d14-1179

[6] da Xu, chuanwei ruan, evren korpeoglu, sushant kumar, and kannan achan. 2020. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJeW1yHYwH

[7] Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. HyTE: Hyperplane-based Temporally aware Knowledge Graph Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 2001–2011. https://doi.org/10.18653/v1/D18-1225

[8] Raimundo Dos Santos, Sumit Shah, Feng Chen, Arnold Boedihardjo, Chang-Tien Lu, and Naren Ramakrishnan. 2014. Forecasting Location-Based Events with Spatio-Temporal Storytelling. In *Proceedings of the 7th ACM SIGSPATIAL International Workshop on Location-Based Social Networks* (Dallas/Fort Worth, Texas) *(LBSN '14)*. Association for Computing Machinery, New York, NY, USA, 13–22. https://doi.org/10.1145/2755492.2755496

[9] Tony Gracious, Shubham Gupta, Arun Kanthali, Rui M. Castro, and Ambedkar Dukkipati. 2021. Neural Latent Space Model for Dynamic Networks and Temporal Knowledge Graphs. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*.

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[11] Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. Recurrent Event Network: Autoregressive Structure Inferenceover Temporal Knowledge Graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 6669–6683. https://doi.org/10.18653/v1/2020.emnlp-main.541

[12] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6980

[13] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[14] Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 1771–1776.

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

[16] Farzaneh Mahdisoltani, Joanna Asia Biega, and Fabian M. Suchanek. 2014. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *CIDR*.

[17] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B. Schardl, and Charles E. Leiserson. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

[18] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B. Schardl, and Charles E. Leiserson. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

[19] Fotios Petropoulos and Spyros Makridakis. 2020. Forecasting the novel coronavirus COVID-19. *PloS one* 15, 3 (2020), e0231236.

[20] Bastiaan Quast. 2016. rnn: a Recurrent Neural Network in R. *Working Papers* (2016). http://qua.st/rnn

[21] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. 2020. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637* (2020).

[22] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web*, Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam (Eds.). Springer International Publishing, Cham, 593–607.

[23] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. 2017. Structured Sequence Modeling with Graph Convolutional Recurrent Networks. In *ICONIP*.

[24] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197* (2019).

[25] Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. 2017. Know-Evolve: Deep Temporal Reasoning for Dynamic Knowledge Graphs. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 3462–3471. http://proceedings.mlr.press/v70/trivedi17a.html

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[27] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations* (2018).

[28] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019. Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks. *arXiv preprint arXiv:1909.01315* (2019).

[29] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2021. Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=KYPz4YsCPj

[30] Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020. TeRo: A Time-aware Knowledge Graph Embedding via Temporal Rotation. In *Proceedings of the 28th International Conference on Computational Linguistics*. 1583–1593.

[31] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation Learning on Graphs with Jumping Knowledge Networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer G. Dy and Andreas Krause (Eds.). PMLR, 5449–5458.

[32] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 793–803.

[33] Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. 2021. Learning from History: Modeling Temporal Knowledge Graphs with Sequential Copy-Generation Networks. In *AAAI*.