Spatial-temporal Knowledge Graph Network for Event Prediction

Anonymous Author(s)

ABSTRACT

Predicting multiple concurrent events has a remarkable effect on understanding social dynamics and acting in advance to reduce damage. With the interaction between different countries and regions more intense, trans-regional implication, which means the cause of the incident is not local but somewhere else, is an important reason for the occurrence of events. However, existing works neglect to model this spatial influence and only leverage the local information for event prediction. What's more, the natural world is continuous, and its dynamic evolving process is driven by emerging events, like an indigenous war, perhaps causing more social conflicts. Nonetheless, most studies ignore that the states of one entity at different timestamps are continuous and introduce some independent snapshots to model the temporal feature. To tackle the above two problems, we propose a **s**patial and **t**emporal **k**nowledge **g**raph neural **n**etwork (*STKGN*). Specifically, to construct the cross-regional connection, we propose a novel spatial-temporal event graph, where each region is denoted as a node and trans-regional influences are bidirectional edges. An event-driven memory network is designed to represent the state of an entity and is updated by emerging events, simulating a continuously evolving process. Then we use a broadcast network to spread the local information in the graph to obtain high-order reasons like the trans-regional implication. Extensive experiments on two real-world datasets demonstrate that STKGN achieves significant improvements over state-of-the-art methods. Further analysis shows the interpretability of the trans-regional implication.

CCS CONCEPTS

• **Information systems** → **Spatial-temporal systems**.

KEYWORDS

Multi-Event Prediction, Knowledge Graph, Dynamic Graph Embedding

ACM Reference Format:

Anonymous Author(s). 2022. Spatial-temporal Knowledge Graph Network for Event Prediction. In *Proceedings of 16st ACM International Conference on Web Search and Data Mining (WSDM 2023).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Social events such as organized crime, arrest, and protest parades show high occurrences in specific locations, times, and semantics. Predicting multiple co-occurring events of different types, also known as multi-event prediction, can reduce the potential social upheaval and damage, which meets the urgent need of the government. Early methods focus on mining specific patterns of the given event type, where each pattern is defined as a burst of context features. To build the mapping function from the feature to the occurrence of events, different algorithms (e.g. linear regression [3, 35] and multi-task learning [9, 10, 26, 41]) are adopted to model the underlying relations as indicators of ongoing or future events. However, they neglect to discover the hidden relational knowledge among entities.

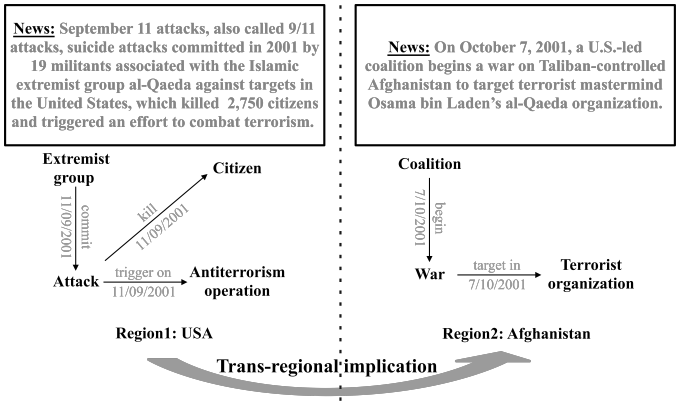

Figure 1: An example of trans-regional implications behind the occurrences of events.

Recently, Graph Neural Networks (GNN) have been widely researched to address non-euclidean data in many domains such as recommender systems [14], natural language processing [23] and protein interface prediction [8]. To explore the semantic correlation between entities, some works [6, 7, 30] utilize the temporal knowledge graphs to represent events extracted from formal reports or news articles by regarding subjects and objects as nodes and event types as edges, which encodes temporal text features into graphs and helps for forecasting. They follow the discrete-time dynamic graph neural network (DGNN) pattern, which means a list of graph snapshots taken at intervals in time is constructed to model the sequences of previous events. Technologically, a knowledge graph neural network, like CompGCN [34], is introduced to encode each snapshot, followed by an RNN-based module to learn the temporal patterns. Due to the advantages of GNN in better semantic embedding, these discrete-time dynamic graph based methods achieve the SOTA prediction performances.

Despite the previous successes, we argue that two vital problems have not been considered as follows: (1) **There are trans-regional implications behind the occurrences of events (Q1)**. The reason why a set of events happen in one district perhaps is not here but in other regions, especially for some first-time events. Take Figure 1 as an example. The 9/11 events in the USA triggered a war and caused a chain of conflict events in Afghanistan, revealing the geopolitical influence. However, existing methods are self-contained in each region, and the judgment of whether an event will occur is only based on what happened in the same region, which fails to consider the reasoning of spatial relation and creates a critical bottleneck to improving prediction performance. (2) **The real world is continuous and evolving (Q2)**, which means the state of an entity should continuously change, and its development should be driven by concerned events. Specifically, the state of an entity in the current timestamp is supposed to be inferred based on its state in the previous and emerging events. However, past studies leverage discrete-time dynamic graphs to learn

WSDM 2023, February 27-March 3, 2023, Singapore

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

entity embeddings in each timestamp, leading to a set of discrete embeddings of the same node in different timestamps. This kind of graph representation learning algorithm fails to model the evolving process of the natural world and is the sub-optimal way to simulate real-life systems, which has a similar consensus in Jodie [19], TGN [28] and TGAT [39].

In order to concurrently address all these technical challenges, this work presents a spatial-temporal knowledge graph network(STKGN) for event prediction. To solve **Q1**, we add the region information to temporal knowledge graphs as a kind of nodes, which are defined to be connected to each other bidirectionally and permanently. Consequently, the trans-regional influence is naturally held in this novel knowledge graph, which is the basis of mining the spatial-temporal pattern of event occurrence. To address **Q2**, we design an event-driven continuous-time dynamic graph neural network, including memory and broadcast networks. The memory network, whose update is only driven by events, aims to simulate the gradual development of entities followed by the broadcast network, by which the local changes propagate over the graph to explore the high-order influences on evolution. To summarize, the main contributions of this work are as follows:

- We construct a novel spatial-temporal knowledge graph for event prediction, which both holds the trans-regional influence and the time sequence pattern.
- We propose a continuous-time dynamic graph neural network to simulate and forecast the evolving process of entities.
- We conduct comprehensive experiments on two public datasets, which proves the effectiveness and interpretability of our proposed model.

2 PRELIMINARIES

In this section, we introduce the newly proposed spatial-temporal event graph used in this paper and then formulate the problem with some notations.

2.1 Spatial-temporal Event Graph

We propose a novel event graph $\mathcal{G}$ to hold the spatial and temporal information. Let $\mathcal{E}$, $\mathcal{L}$ and $\mathcal{R}$ denote the set of entities, locations, and event types, respectively. When an event $ev_{so}(t)$, which belongs to event type r and is associated with the subject s and object o, happens in location l at time t, we use three quadruples to record this history: (s, r, t, o), (s, r, t, l) and (o, r, t, l). To this end, the event type and temporal feature are represented by edges, while the spatial information is contained in nodes. To represent spatial influence, we add a new edge relation called *trans-regional influence* between each pair of locations, which is symmetric and exists in all timestamps. Figure 2 is an example to show the construction of the above spatial-temporal event graph using the news in Figure 1. Different from the discrete-time dynamic graph, which establishes a sequence of event graphs in ascending time order like $\{\mathcal{G}(t_1), \mathcal{G}(t_2), ...\}$, our proposed continuous-time graph continually changes the topology of the graph via the addition or deletion of node and edge. Specifically, the i−th node representation $\mathbf{e}_i(t)$ at timestamp t is inferred based on its previous embedding $\mathbf{e}_i(t-1)$ and the associated events at the current timestamp $ev_{ij}(t)$, that is $\mathbf{e}_i(t) = f(\mathbf{e}_i(t-1), ev_{ij}(t))$. In this way, the embedding of one node consists of a set of continuous vectors $\{\mathbf{e}(t_1), \mathbf{e}(t_2), ...\}$ and depicts a trajectory, which can describe the evolving process of its state.

2.2 Problem Setup

The goal of this study is to predict multiple co-occurring event types in the future. Formally, given historical events in past t timestamps,

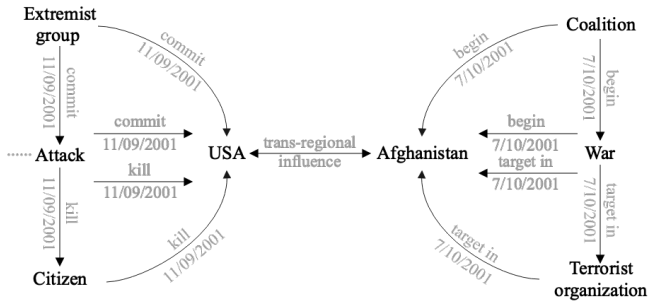

Figure 2: The spatial-temporal event graph.

an embedding $\mathbf{h}_l(t+1)$ is learned to represent the state of a specific location l at time $t+1$. Based on $\mathbf{h}_l(t+1)$, we model the probabilities of event types occurring here at time $t+1$ as

$$\mathbb{P}\left(\mathbf{y}(t+1) \mid \mathbf{h}_l(t+1)\right) = \sigma\left(\mathbf{W}_\gamma \mathbf{h}_l(t+1)\right) \tag{1}$$

where $\mathbf{y}_{t+1} \in \mathbb{R}^{|\mathcal{R}|}$ is a vector of event types. We compute the probabilities of different event types by feeding the latent embedding into a linear layer parameterized by $\mathbf{W}_\gamma \in \mathbb{R}^{|\mathcal{R}| \times d}$ followed by an element-wise sigmoid function for event prediction. In this paper, the number of the used historical timestamps is defined as a hyperparameter T, which will be analyzed in depth in Section 4.4.

3 STKGN

In this section, we first introduce the framework of the proposed model with the underlying motivation and then give the technical details of each module. Figure 3 presents the framework of STKGN.

3.1 Overview

Inspired by TGN [28], we utilize a continuous-time dynamic graph to model the real world. **The core idea is that an event interacting with a node will influence its evolving direction**. We design a memory network to represent the state of an entity and update it only when events happen. Specifically, a low-dimensional vector $\mathbf{m}_i(t)$ is assigned to denote what node i has seen so far at time t. When an emerging event $ev_q(t)$ associated with node i occurs, we first extract its semantics by a text convolution layer f_1 and then pass the helpful information into node i using a message gate network f_2. As one node usually is concerned in many events $ev_i^{all}(t)$, an aggregation function f_3 is proposed to fuse multiple messages from different events into an informative embedding, which is further fed into f_4 for memory update. When a new node first appears, its initial memory is set to be the zero vector. Thanks to this memory-based schema where memory embeddings change continuously, STKGN has the capability to represent the evolving process and gradual change of each node. **Note that the development of a node will affect both itself and the related entities especially for trans-regional influence.** Therefore, we hope to broadcast the local update throughout the graph to model the high-order reasoning chain. For example, the *9/11* events directly change the state of node *USA*, which indirectly affects the military environment of node *Afghanistan*. Technically, a GNN module f_5 is leveraged to pass the updated memory embeddings to its neighbor nodes $\mathcal{N}_i$ and learn the final representations $\mathbf{h}(t+1)$ for downstream tasks. The complete STKGN framework is presented in Algorithm 1.

For the sake of simplicity, we eliminate the time information of a vector, which is set to be t by default, a.k.a. $\mathbf{m}_i = \mathbf{m}_i(t)$, unless we specifically notate it.

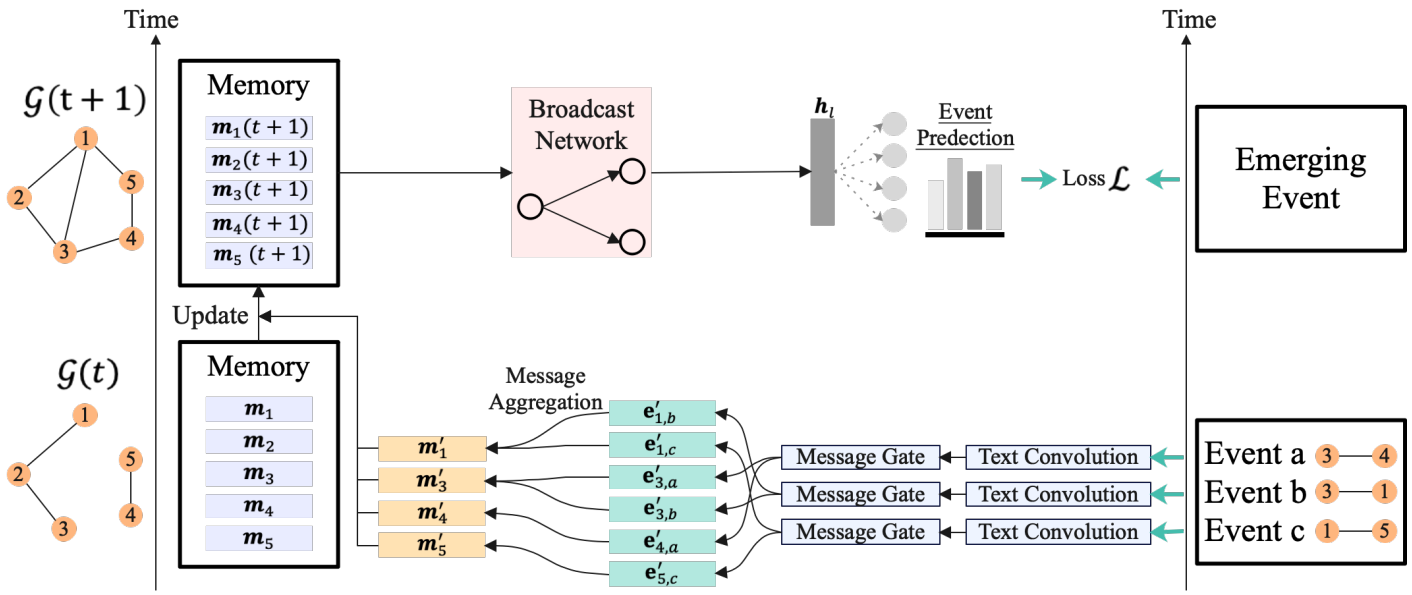

Figure 3: The framework of STKGN. Best viewed in color.

Algorithm 1 The STKGN framework

Input:
three-fold information at time t:
(1) the memory embedding $\mathbf{m}(t)$
(2) graph $\mathcal{G}(t)$
(3) emerging events $ev(t)$
hyperparameters:
(1) GNN layers L
Output:
final representation for prediction $\mathbf{h}$
Memory network (1-8)
for node i in $ev(t)$ **do**
  if node i is first encountered **then**
    $\mathbf{0} \rightarrow \mathbf{m}_i(t)$
  for event $ev_q(t)$ in $ev_i^{all}(t)$ **do**
    text convolution f_1: $ev_q(t) \rightarrow \mathbf{e}_q(t)$
    message gate f_2: $\mathbf{m}_i(t), \mathbf{e}_q(t) \rightarrow \mathbf{e}'_{i,q}(t)$
  message aggregation f_3: $\mathbf{e}'_{i,q}(t) \rightarrow \mathbf{m}'_i(t)$
  memory update f_4: $\mathbf{m}_i(t), \mathbf{m}'_i(t) \rightarrow \mathbf{m}_i(t+1)$
Broadcast network (9-12)
initial graph embedding: $\mathbf{h}_i^{(1)} = \mathbf{m}_i(t+1)$
for layer $l = 1, 2, ..., L$ **do**
  GNN layer f_5: $\mathbf{h}_i^{(l)}, \mathbf{h}_{j \in \mathcal{N}_i}^{(l)} \rightarrow \mathbf{h}_i^{(l+1)}$
pool multi-layer embeddings: $\mathbf{h}_i(t+1) = \sum_{l=1}^{L} \mathbf{h}_i^{(l)}$
update graph: $\mathcal{G}(t) \rightarrow \mathcal{G}(t+1)$

3.2 Memory network

Text convolution With an event represented by a quadruple, we combine its subject, event type, and object to export a sentence, like $(extremist\ group, commit, 11/09/2001, attack) \rightarrow extremist\ group\ commit\ attack$. Our task is to extract its semantics by learning a low-dimensional vector. To this end, a text convolution network is utilized. Specifically, given a document (a.k.a. sentence) consisting of n words as $\{w_1, w_2, ...w_n\}$, we initialize their features using the pretrained embeddings $\mathbf{w}$ from Google News[1] [25] and project the event to an embedding matrix: $\mathbf{E}_{ev} = [\mathbf{w}_1, \mathbf{w}_2, ...\mathbf{w}_n] \in \mathbb{R}^{n \times d_{pre}}$, where d_{pre} is the pretrained embedding dimension. Inspired by textCNN [4], we slide a convolution filter $\mathbf{cf} \in \mathbb{R}^{s \times d_{pre}}$ along rows to extract contextual features from a window of s words, followed by a max-pooling and activation function as

$$c = \sigma_1(\max(\mathbf{cf} * \mathbf{E}_{ev})) \tag{2}$$

where c is a scalar and σ_1 is the *Relu* activation function. Then d' filters are used to learn multiple features and infer an informative representation. What's more, we adopt two window sizes, which are 2 and 3, to capture different aspects of word relations, leading to the final representation of event ev_q as $\mathbf{e}_q = [c_1, c_2, ..., c_{2d'}] \in \mathbb{R}^{d \times 1}, d = 2d'$.

Message gate An emerging event will affect the subject's and object's evolving direction. For node i (a subject or obeject), we use a gate control network to decide what information of the event will flow into its memory unit, which is formulated as

$$\mathbf{e}'_{i,q} = (\mathbf{W}_1 \mathbf{e}_q + \mathbf{b}_1) \odot \sigma_2(\mathbf{W}_2[\mathbf{e}_q \parallel \mathbf{m}_i] + \mathbf{b}_2) \tag{3}$$

where the second term on the right side works as a soft on-off switch controlling the degree how much the memory unit accepts. $\mathbf{W}_1 \in \mathbb{R}^{d \times d}$, $\mathbf{b}_1 \in \mathbb{R}^{d \times 1}$, $\mathbf{W}_2 \in \mathbb{R}^{1 \times 2d}$ and $\mathbf{b}_2 \in \mathbb{R}^1$ are all trainable parameters. σ_2 is the *sigmoid* activation function. $\parallel$ is the concatenation operator. $\odot$ is the element-wise product operation.

Message aggregation Resorting to batch processing for efficiency reasons may lead to multiple events involving the same node i in one batch. Furthermore, critical events have a higher priority and greater impact on the node state. Therefore, we adopt an attention mechanism to fuse multiple messages from various events as

$$b_{i,q} = \mathrm{MLP}(\mathbf{e}'_{i,q} \parallel \mathbf{m}_i) \tag{4}$$

$$w_{i,q} = \frac{\exp b_{i,q}}{\sum_{ev_{q'} \in ev_i^{all}} \exp b_{i,q'}} \tag{5}$$

[1] https://code.google.com/archive/p/word2vec/

$$\mathbf{m}_i' = \sum_{ev_q \in ev_i^{all}} w_{i,q} \mathbf{e}_{i,q}' \tag{6}$$

where ev_i^{all} denotes all events associated with node i at the current timestamp, **MLP** is a two-layers feedforward neural network with the *LeakyReLU* as activation function. In this way, we attentively collect all messages which appear in current timestamp t and are associated with this node.

Memory Update With a low-dimensional embedding to represent the upcoming change on this node, we update its memory in a continuous way to simulate the evolving process. Inspired by neural turing machines [12], we use two controllers of its memory write module, which are **add** and **erase** operations. Specifically, a new value that will be added to the memory states is dependent on both the input $\mathbf{m}_i'$ and the current states $\mathbf{m}_i$ as

$$\mathbf{add}_i = \sigma_1 \left(\mathbf{W}_3 \left[\mathbf{m}_i \parallel \mathbf{m}_i' \right] + \mathbf{b}_3 \right) \tag{7}$$

Meanwhile, a scalar $erase_i$ is leveraged to control how much the current states are involved in generating the new add-on value as

$$erase_i = \sigma_2 \left(\mathbf{W}_4 \left[\mathbf{m}_i \parallel \mathbf{m}_i' \right] + \mathbf{b}_4 \right) \tag{8}$$

Then the memory states are updated by:

$$\mathbf{m}_i(t+1) = (1 - erase_i) \cdot \mathbf{m}_i + erase_i \cdot \mathbf{add}_i \tag{9}$$

$\mathbf{W}_3 \in \mathbb{R}^{d \times 2d}$, $\mathbf{W}_4 \in \mathbb{R}^{1 \times 2d}$, $\mathbf{b}_3 \in \mathbb{R}^d$, and $\mathbf{b}_4 \in \mathbb{R}^1$ are all learnable parameters.

Essentially, Equation 9 adopts a linear combination between previous states $\mathbf{m}_i$ and the new add-on vector $\mathbf{add}_i$, which prevents the consecutive dynamic embeddings of a node from varying too much and drives the evolutionary change of the memory vector along a continuous trajectory in the latent space.

3.3 Broadcast network

As aforementioned in 3.1, related nodes have potential bidirectional influences, especially in two location nodes. The message about one node may indirectly affect the development of neighbor nodes. Therefore, we learn the relational embeddings to capture influence flow in the graph. Inspired by CompGCN [34], we perform a multi-layer graph convolution network as

$$\mathbf{h}_i^{(l+1)} = f\left(\sum_{(j,i) \exists (j,r,t,i) \in \mathcal{G}_t} \mathbf{W}_5^{(l)} \phi\left(\mathbf{h}_j^{(l)}, \mathbf{o}_r^{(l)} \right) \right) \tag{10}$$

where $\mathbf{h}_i^{(l)}$ and $\mathbf{o}_r^{(l)}$ denote representations in l-th layer for node i and event type r, respectively. For the initial layer, $\mathbf{h}_i^{(1)} = \mathbf{m}_i(t+1)$. $\mathbf{W}_5^{(l)} \in \mathbb{R}^{d \times d}$ is the weight parameter for aggregating operation in the l-layer. ϕ is the multiplication operation to combing node and relation embeddings. f is the $Tanh$ activation function. Then the relation embeddings are updated as follows

$$\mathbf{o}_r^{(l+1)} = \mathbf{W}_{rel} \mathbf{o}_r^{(l)} \tag{11}$$

where $\mathbf{W}_{rel} \in \mathbb{R}^{d \times d}$ is the transformation matrix, which projects all the relations to the same embedding space and allows them to be utilized in the next layer.

The final representation of the location node in time t is infered via sum pooling as

$$\mathbf{h}_l = \sum_{l=1}^{L} \mathbf{h}_l^{(l)} \tag{12}$$

3.4 Model Optimization

We regard the target task, a.k.a. event prediction, as a multi-label classification. Specifically, given a location $\mathbf{h}_l(t)$ and its ground truth label $\hat{\mathbf{y}}(t+1) = \{\hat{y}_i\}^{|\mathcal{R}|}$. Typically, y_i is a binary value, which denotes whether this kind of event happens. However, we unequally treat different event types and transfer the true label set to a distribution via the frequency. In consequence, $\hat{y}_i$ is a decimal in $[0, 1]$ and $\sum_{i=1}^{|\mathcal{R}|} \hat{y}_i = 1$. Meanwhile, Equation 1 gives the predicted probabilities $\mathbf{y}(t+1) = \{y_i\}^{|\mathcal{R}|}$. We normalize them via softmax operation. By this means, the categorical cross-entropy loss [22, 24] is designed as

$$\mathcal{L}_l = -\frac{1}{|\mathcal{R}|} \sum_{i \in |\mathcal{R}|} \hat{y}_i \log \left(\frac{\exp(y_i)}{\sum_{j \in |\mathcal{R}|} \exp(y_j)} \right) \tag{13}$$

Then we traverse all locations and summarize losses to optimize the model as

$$\mathcal{L} = \sum_{l \in \mathcal{L}} \mathcal{L}_l \tag{14}$$

We adopt the schema of next event prediction to regularize the training process. In each batch, all observed data within the given time windows $\{t, t+1, ..., t+T-1\}$ of a fixed length T are leveraged to predict what events will happen in time $t+T$. We slide this window along the whole dataset to create independent but temporally consistent training batches.

As for inference, we use a sigmoid function over the predicted score y_i and set a threshold, which is 0.5 by default, to decide the occurrence of an event.

4 EXPERIMENTS

We evaluate our proposed method to answer the following research questions:

- **RQ1**: Does our proposed STKGN outperform the state-of-the-art methods?
- **RQ2**: From the spatial pattern perspective, can STKGN provide potential explanations about trans-regional implications?
- **RQ3**: From the temporal pattern perspective, does the scheme of continuous-time dynamic graph proposed in STKGN perform better in time sensitivity and robustness compared with past temporal models in event prediction?
- **RQ4**: How do different components (i.e., memory network and broadcast network) and the critical hyperparameter (i.e. GNN layer numbers) affect STKGN?

4.1 Experimental Settings

Dataset Description. We use two common datasets in event prediction, which are the Integrated Conflict Early Warning System (ICEWS) and Global Database of Events,Language,and Tone (GDELT) [38]. Both of them are publicly available[2] with a similar data format, and we introduce one detailly for simplicity. ICEWS contains abundant political and crisis events in 260 countries or districts. These events are coded using 20 main categories (e.g. *make public statement* and *appeal*) and 296 (220 in GDELT) subcategories (e.g. *decline comment* and *appeal for diplomatic cooperation*). Each event is represented by its time (day, month, year), location (city, district, province), entity (subject, object) and event type, etc. In this paper, we select all events from three countries (a.k.a. $|\mathcal{L}| = 3$) in each dataset: Iraq, Afghanistan, and Iran in ICEWS while Iraq, Turkey, and Iran in GDELT. All data range from

[2]ICEWS: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/TYHZAK; GDELT: http://data.gdeltproject.org/events/index.html

Table 1: Statistics of two datasets.

		$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	#events
ICEWS	Iraq	6,581	224	32,107
	Afghanistan	8,002	239	36,285
	Iran	7,362	219	17,924
	all	17,529	245	86,316
GDELT	Iraq	4,903	189	21, 901
	Turkey	6,183	206	28, 850
	Iran	5,875	197	29,755
	all	15,943	215	80,506

Jan. 1, 2000 to Jan. 1, 2010 and the time granularity is one day. Data statistics are shown in Table 1.

Baselines. To demonstrate the effectiveness, we compare STKGN with three kinds of methods: designed specifically for event prediction without the usage of GNN (DNN, SIMDA), designed specifically for dynamic graph representation learning rather than event prediction (T-GCN, EvolveGCN, Jodie, TGN), and designed specifically for event prediction based on GNN (RENET, Glean, DGCN-rs).

- **DNN** Followed the setting in [7, 30], it is a deep neural network consisting of three dense layers.
- **SIMDA** [10] Based on the "first law of geography", it shares similar event subtype patterns across spatially closed tasks.
- **T-GCN** [40] It combines the GCN and GRU for traffic prediction, where the GCN is used to learn complex topological structures and the GRU is employed to learn dynamic traffic changes.
- **EvolveGCN** [27] It use an RNN to evolve the GCN parameters without resorting to node embeddings, which is more applicable to the frequent change of the node set.
- **Jodie** [19] It employs two recurrent neural networks to update the representations of users and items at every interaction and learns their continuous embedding trajectories.
- **TGN** [28] It proposes a generic inductive framework of temporal graph networks operating on continuous-time dynamic graphs.
- **RENET** [16] It employs a neighborhood aggregator to model the connection of facts at each timestamp and a recurrent event encoder to infer future events.
- **Glean** [7] It introduces CompGCN [34] to aggregate the entity and event type embeddings in each event snapshot, which are fed into a recurrent encoder to model temporal information for final prediction.
- **DGCN-rs** [30] It is a variant of Glean and replaces the RNN module in Glean with a dilated casual convolutional network.

Note that some baselines (e.g. Glean, DGCN-rs) are self-contained in one region and only leverage local events for prediction. To make the most of their potential and test their best performances in our datasets, we follow the input consistency principle and set the following two versions of such baselines:

- **baseline**$_{\text{local}}$ For each region, we adopt the same graph construction methods used in [7, 30] and collect local events to build a temporal event graph, which doesn't include nonlocal events. To this end, when forecasting future events in one country, this version only uses local histories.
- **baseline**$_{\text{global}}$ Followed the method proposed in this paper, we construct a spatial-temporal event graph via the data from several countries. When forcasting future events in one country, this version leverages both the local and nonlocal histories.

For those models that already take into account geographic information (e.g. SIMDA, STKGN), we use complete data and no additional annotations to the model.

Parameter Settings. We implement our STKGN model in Pytorch and Deep Graph Library (DGL)[3], which is a Python package for deep learning on graphs. For the same side information, we initialize node features via the same pretrained embeddings from Google News[4] followed by a transformation matrix to obtain the expected embedding size. We train all models by splitting the data by time to simulate the real situation. Therefore, we train all models on the first 80% of events, validate on the next 10%, and test on the last 10% of data. For GNN-based methods, the length of historical time windows and the propagation layers are set to 14 and 2 separately. For T-GCN, the number of hidden units is set to 100 for best performance. For Jodie, the subject and object are regarded as the user and item in an interaction. For TGN, the MLP and multi-head attention are adopted as the message function and embedding module separately. For DGCN-rs, dilated factor d is set to be $1, 2, 4$. The embedding size of all models is set to be 64 for a fair comparison. We adopt Adam [18] as the optimizer and the batch size is fixed at 1024 for all methods. We apply a grid search for hyper-parameters: the learning rate and the coefficient of L2 normalization are searched in $\{10^{-5}, 10^{-4}, \cdots, 10^{-1}\}$, and the dropout ratio is tuned in $\{0.0, 0.1, \cdots, 0.9\}$.

Evaluation Metrics.

- **Recall** evaluates whether the model can predict the event types that will occur in the future as fully as possible.
- **F1** is the weighted harmonic mean of precision and recall to comprehensively trade off recall and false predictions.

4.2 Performance Comparision (RQ1)

We report the empirical results in Table 2. The observations are as follows:

- STKGN consistently achieves the best performance on two datasets in terms of all measures. Specifically, it achieves significant improvements over the strongest baselines w.r.t. F1 by 6.31% and 3.22% in Afghanistan and Turkey, respectively. There are two main reasons: (1) STKGN incorporates the location nodes, which helps to model the trans-regional implications and reveal the spatial relations of event occurrences, and (2) the scheme of the continuous-time dynamic graph has SOTA ability to model the teporal pattern of events.
- Despite some particular cases, the global version of baselines outperforms its local version. The performance gain benefits from the introduction of additional nonlocal information. Moreover, the degrees of improvement from the local to global version vary over baselines, which is due to the different applicabilities of the model to this novel spatial-temporal knowledge graph.
- Among dynamic graph representation learning models (T-GCN, EvolveGCN, Jodie and TGN) and GNN-based event prediction models (RENET, Glean, DGCN-rs and STKGN), continuous-time DGNN based methods outperform discrete-time DGNN based methods, like T-GCN → TGN and Glean → STKGN. One obvious reason is the superiority of continuous-time DGNN over discrete-time DGNN.
- Jointly analyzing all models in the same country (e.g. Iran and Iraq) across two datasets, we find the performance in GDELT is poor than that in ICEWS. Moreover, the improvement from the local to the global version is also slightly more than that in ICEWS. A possible

[3]https://github.com/dmlc/dgl
[4]https://code.google.com/archive/p/word2vec/

Table 2: Performance comparision and the number is in the percentage formula (%). The best result in each country is highlighted in bold face and the second best are in underlined. ▲% denotes the relative improvement of STKGN over the scond best algorithm.

	ICEWS						GDELT					
	Iraq		Afghanistan		Iran		Iraq		Turkey		Iran	
	Recall	F1	Recall	F1	Recall	F1	Recall	F1	Recall	F1	Recall	F1
DNN	66.16	54.87	68.78	55.32	65.53	57.75	60.74	51.44	65.11	53.88	59.77	51.66
SIMDA	71.28	59.43	75.06	61.47	72.21	59.62	67.91	59.59	71.65	58.00	68.14	53.05
T-GCN$_{local}$	72.40	59.79	76.23	64.97	74.50	63.87	70.18	60.05	76.67	61.15	70.76	54.58
T-GCN$_{global}$	76.92	62.71	80.65	63.45	75.95	65.30	69.68	60.89	75.11	61.74	72.97	55.79
EvolveGCN$_{local}$	68.57	56.27	66.38	52.51	69.29	58.65	65.20	54.72	70.36	56.16	65.14	51.86
EvolveGCN$_{global}$	69.56	57.00	67.78	53.16	69.51	57.03	64.55	53.62	69.52	56.42	64.41	50.88
Jodie$_{local}$	77.63	62.67	74.72	63.68	75.69	63.22	71.25	61.37	78.61	62.69	72.19	55.40
Jodie$_{global}$	79.56	63.96	80.33	66.55	76.48	63.92	73.87	65.10	78.83	63.15	72.87	56.07
TGN$_{local}$	82.59	67.74	80.17	68.60	81.71	67.28	77.83	68.79	82.23	66.86	77.48	61.42
TGN$_{global}$	84.27	70.53	83.58	68.00	83.34	69.54	79.96	70.39	83.02	68.79	78.56	62.11
RENET$_{local}$	74.40	62.32	75.97	61.43	73.92	62.19	72.43	64.43	75.98	61.23	71.05	57.70
RENET$_{global}$	75.95	63.41	77.64	66.89	75.71	64.20	71.15	63.04	74.82	60.87	71.15	57.80
Glean$_{local}$	77.64	64.37	79.22	63.05	78.27	69.32	74.82	67.37	80.04	65.49	74.81	57.58
Glean$_{global}$	79.48	64.90	83.13	69.03	79.64	70.53	77.12	68.78	81.87	66.45	77.75	59.74
DGCN-rs$_{local}$	81.12	67.85	81.68	66.29	80.08	67.78	79.28	69.49	81.92	67.48	79.29	64.99
DGCN-rs$_{global}$	83.58	69.05	84.73	69.92	82.26	70.20	80.12	70.30	82.38	68.31	80.32	65.27
STKGN	**88.35**	**74.57**	**89.08**	**74.33**	**86.62**	**73.56**	**83.06**	**72.67**	**84.74**	**71.01**	**82.04**	**67.18**
▲%	4.84	5.73	5.13	6.31	3.94	4.79	3.67	3.23	2.07	3.22	2.14	2.92

reason is that GDELT includes many aspects of society, such as military, political, business, and social media, while ICEWS mainly consists of conflict-related events. Therefore, it is more difficult to recognize the event occurrences and evolution patterns in GDELT.

4.3 Case Study of Trans-regional Implication (RQ2)

To prove the interpretability of STKGN, we use the events of the Iraq War that broke out on Mar. 20, 2003, as an example to predict what type of events will happen in Iran. We visualize the predicted probability of the relationship *provide aid*. The ground truth is that Iran provided humanitarian aid on Mar. 21. Thus, we expect its predicted probability to be greater than 0.5, as aforementioned in Section 3.4. We conduct two tests based on whether or not to remove the influence between Iraq and Iran. The results in Figure 5 show that removing this edge leads to wrong predictions and a sharp drop in the predicted probability, from 0.61 to 0.07. On the contrary, the introduction of the relationship *trans-regional influence* successfully predicts the event type *provide aid* in Iran, which especially is a first-time event in the past month.

4.4 Time Sensitivity and Robustness Analysis (RQ3)

In this section, we explore the following two aspects: (1) the sensitivity to the number of historical timestamps T, and (2) the robustness when partial past data is missing, which is common in practical scenarios.

According to the method of modeling temporal signals, we classify past works into four types with some necessary explanations: (1) **Feature mapping** (DNN and SIMDA). The current or past events are regarded as features, which are fed into a classification model. (2) **RNN** including its variants like GRU (T-GCN, EvolveGCN, RENET, Glean). They follow the framework of discrete-time DGNN, where GNN is used to capture topological structures and RNN is specifically used to model sequential patterns. Therefore we call them RNN-based methods just from the temporal pattern perspective. (3) Dilated casual convolutional network, abbreviated as **DCC** (DGCN-rs). (4) Continuous-time DGNN, abbreviated as **CTDGNN** (Jodie, TGN, STKGN). Different from the discrete-time DGNN, they deeply integrate the temporal pattern into the GNN framework. For each kind of method, we choose one SOTA model according to the performances in Table 2. They are SIMDA, Glean, DGCN-rs and STKGN. Since there is the similar findings across model versions, cities and datasets, we just show the performance of the global version in Iraq using the ICEWS dataset for simplicity.

Sensitivity analysis. We search T in the range of {7, 14, 21, 28}. The results of Recall and F1 are in Figure 4(a) and (b). We observe that CTDGNN is more suitable for capturing long-term dependence among events, while the performances of the other three methods all drop slightly as the T becomes larger. The underlying reason is the adaptability of the memory network to social events. Social events hide potential long-term temporal dependence while the memory unit has a natural advantage for modeling long-term regularities, which has been demonstrated in recommender systems [15, 32]. Furthermore, in the baselines, DCC outperforms the feature mapping and RNN based mothods because the dilated convolution operation expands the receptive field, which has a similar finding in [30].

Robustness analysis. We randomly mask different ratios of training data, which are{10%, 20%, 30%, 40%, 50%}, and keep the test data unchanged. T is fixed to 14. Figure 4(c) shows that STKGN has clearly lower performance degradation compared to other methods when keeping less and less data, which is of great significance in practical scenarios. A possible reason is that the memory embedding representing the node state in CTDGNN is continuously evolving, and missing a limited part of past events will not drastically change its development direction. In addition, an interesting phenomenon is that RNN is not as robust as the feature mapping, which may be due to the fact that

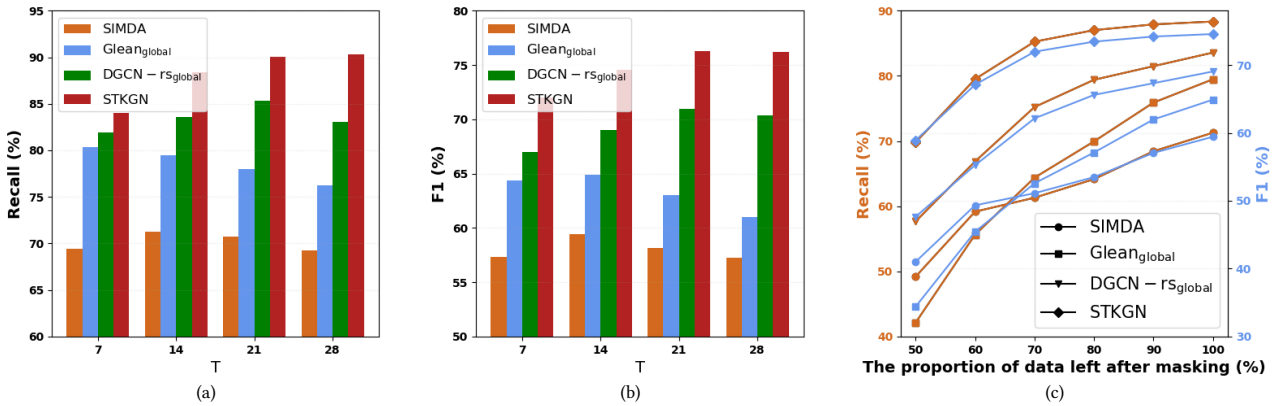

Figure 4: Time Sensitivity and Robustness Analysises. In (c), 100 % data left means no masking. Best viewed in color.

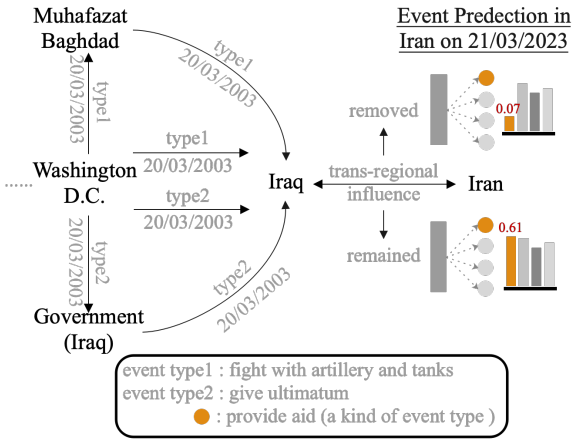

Figure 5: Case Study of Trans-regional Implication. Best viewed in color.

the feature mapping does not rely on time-series signals but only on the sufficiency of training data.

4.5 Study of STKGN (RQ4)

In this section, we conduct an ablation study to investigate the effect of different components in STKGN. In the following, we study the influence of an important hyperparameter, a.k.a. GNN layer numbers.

Impact of some important modules. To demonstrate the superiority of the memory network, broadcast network, and the add-erase operation to continuously update memory embedding, we compare the performance of STKGN with the following variants.

- **STKGN-w/o m** removes the memory network. In order to keep fairness and the same introduction of pre-training knowledge, we keep the text convolution layer to generate the initial embedding of the broadcast network.
- **STKGN-w/o b** removes the broadcast network., which means we directly use the memory embedding for prediction.
- **STKGN-mean** replaces the add-erase operation with average pooling.
- **STKGN-LSTM** replaces the add-erase operation with LSTM recommended in TGN to update memory.

Table 3: Impact of some important modules and the number is in the percentage formula (%).

	Iraq (ICEWS)		Iraq (GDELT)	
	Recall	F1	Recall	F1
STKGN-w/o m	73.06	60.67	68.53	59.69
STKGN-w/o b	83.13	69.75	80.07	70.71
STKGN-mean	85.83	71.47	81.26	71.15
STKGN-LSTM	82.58	68.70	77.15	68.90
STKGN-GRU	82.91	68.83	76.86	68.41

- **STKGN-GRU** replaces the add-erase operation with GRU recommended in TGN.

Table 3 shows the results in Iraq of two datasets, and it has similar conclusions in other countries, which are omitted. The major findings are as follows:

- The performance of STKGN-w/o m sharply degrades without the memory network, which proves the necessity of its capability to model node evolution.
- STKGN-w/o b underperforms STKGN. An obvious reason is that the lack of a propagation network prevents higher-order effects.
- Among three variants of the memory update module, STKGN-mean slightly underperforms STKGN while the other two achieve the worst performances. A possible reason is that average pooling maintains a linear combination and proves a continuous evolution of node states while the other two operations do not.

Impact of GNN layer numbers. The number of GNN layers affects the scope of the local update. We search L in the range of $\{1, 2, 3\}$ for all GNN-based baselines. Table 4 shows the results in Iraq of two datasets. From the observations above, we can find two uniform phenomenons: (1) $L = 2$ has the best performance, and (2) continuing to stack more GNN layers leads to overfitting. The reason is the inherent topology of this spatial-temporal event graph, where all important information of a location node is stored within its 2-hop range. For example, $Extremist\ group \xrightarrow{commit} USA \xleftarrow{trans-regional\ influence} Afghanistan$ has fully represented the impact on Afghanistan caused by related actors in another region. Therefore, $L = 2$ is the best, and too many layers can bring noisy signals.

Table 4: Impact of GNN layer numbers and the number is in the percentage formula (%).

	Iraq in ICEWS						Iraq in GDELT					
	L=1		L=2		L=3		L=1		L=2		L=3	
	Recall	F1	Recall	F1	Recall	F1	Recall	F1	Recall	F1	Recall	F1
T-GCN$_{global}$	76.07	62.25	76.92	62.71	74.78	60.92	68.04	59.37	69.68	60.89	65.37	58.23
EvolveGCN$_{global}$	68.76	56.15	69.56	57.00	66.77	54.82	63.31	52.19	64.55	53.62	60.11	49.46
Jodie$_{global}$	77.91	62.44	79.56	63.96	76.67	62.10	72.46	64.25	73.87	65.10	71.99	63.72
TGN$_{global}$	83.18	70.05	84.27	70.53	82.41	69.59	79.15	69.82	79.96	70.39	75.72	67.31
RENET$_{global}$	73.69	62.17	75.95	63.41	73.96	62.33	70.58	62.66	71.15	63.04	70.37	62.19
Glean$_{global}$	78.71	64.53	79.48	64.90	78.05	63.89	76.34	68.12	77.12	68.78	75.20	67.34
DGCN-rs$_{global}$	82.37	68.54	83.58	69.05	82.01	68.12	79.17	67.79	80.12	68.31	77.62	66.22
STKGN	87.07	73.95	88.35	74.57	86.69	73.12	82.31	71.99	83.06	72.67	80.86	70.63

5 RELATED WORK

Our study is closely related to a large body of literature on event prediction and dynamic graph representation.

Event prediction. There are two kinds of algorithms for event prediction. (1) Early studies take text features as input and use a multi-classification paradigm to predict event types. According to the feature mapping method, there are two solutions: linear regression [3, 35] and multi-task learning [9, 10, 26, 41]). Linear regression based methods first extract domain-specific features from the text and then uses a linear model to calculate the probability. [3] analyzes the text content of daily Twitter feeds to predict the stock market by two mood tracking tools, namely OpinionFinder and Google-Profile of Mood States (GPOMS). In order to further consider the geographical heterogeneity, the formulation of a multi-task learning framework for event forecasting is proposed. [41] builds a forecasting model for all locations simultaneously by extracting and utilizing appropriate shared information that effectively increases the sample size for each location, thus improving the forecasting performance. [10] shares similar event subtype patterns across spatially closed tasks and regard different timestamps as independent variables to extract temporal features. However, none of the above methods fully consider the deep connection between the actors of the events. (2) Recently, GNN-based methods show full potential in event prediction [7, 16, 30]. They follow the pattern of discrete-time dynamic graph neural network (DGNN) using a set of graph snapshots taken at intervals in time. Glean [7] employs CompGCN [34] to encode each snapshot and leverages RNN to model temporal signals. DGCN-rs [30] just replaces the RNN module in Glean with a dilated casual convolutional network to better capture long-term dependence. Although they perform better than feature mapping methods, they are still limited by discrete-time dynamic graph representations.

Dynamic graph representation. According to the approaches to model the system's time domain, there are two categories for dynamic graph representation. (1) discrete-time approaches [11, 21, 27, 29, 31, 40]. DySAT [36] generates a dynamic node representation by joint self-attention along two dimensions: structural neighborhoods and temporal dynamics. The former extracts feature from local node neighborhoods in each snapshot, while the latter captures graph evolution over multiple time steps. DyGGNN [31] leverages the gated graph neural networks (GGNNs) to learn graph's topology at each time step and LSTMs to propagate the temporal information among the time steps. Besides, EvolveGCN [27] proposes a novel dynamic evolution paradigm that evolves GCN parameters instead of nodes. However, most real-life systems of interactions such as traffic networks or biological interactomes are dynamic. Applying static graph deep learning models to dynamic graphs by ignoring the temporal evolution [20], this has been shown to be sub-optimal [39]. (2) continuous-time approaches [5, 19, 21, 28, 33, 37]. [5] proposes a continuous-time version of node2vec [13] for a more efficient dynamic link prediction. Jodie [19] aims to learn the continuous embedding trajectories of user and item in recommendation via updating them by two recurrent neural networks. DyRep [33] proposes a novel modeling framework for dynamic graphs that posits representation learning as a latent mediation process bridging two observed processes, which are the topological evolution and activities between nodes. TGAT [37] uses the self-attention mechanism as a building block and develops a novel functional time encoding technique based on the classical Bochner's theorem from harmonic analysis. TGN [2] designs a generic, efficient framework for deep learning on continuous-time dynamic graphs represented as sequences of timed events. However, the downstream task of the above models is link prediction or node classification [1, 17], not specifically for event prediction. To our best knowledge, STKGN is the first attempt to introduce the continuous-time dynamic graph for event prediction.

6 CONCLUSION

In this paper, we aim to mine the spatial and temporal patterns of event occurrences. We propose a novel spatial-temporal knowledge graph to model the trans-regional influence and a continuous-time dynamic graph neural network to simulate the evolving process of nodes. Comprehensive experiments are conducted to demonstrate the effectiveness and explainability of the above methods. For future work, we plan to investigate the effect of personalization between a pair of locations because the degree of correlation varies between countries. Another direction is to use multiple memory embeddings to represent entities' long-term and short-term states.

REFERENCES

[1] Claudio DT Barros, Matheus RF Mendonça, Alex B Vieira, and Artur Ziviani. 2021. A survey on embedding dynamic graphs. *ACM Computing Surveys (CSUR)* 55, 1 (2021), 1–37.

[2] Ye Bi, Liqiang Song, Mengqiu Yao, Zhenyu Wu, Jianming Wang, and Jing Xiao. 2020. A heterogeneous information network based cross domain insurance recommendation system for cold start users. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2211–2220.

[3] Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of computational science* 2, 1 (2011), 1–8.

[4] Yahui Chen. 2015. *Convolutional neural network for sentence classification.* Master's thesis. University of Waterloo.

[5] Sam De Winter, Tim Decuypere, Sandra Mitrović, Bart Baesens, and Jochen De Weerdt. 2018. Combining temporal aspects of dynamic networks with Node2Vec for a more efficient dynamic link prediction. In *2018 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)*. IEEE, 1234–1241.

[6] Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2019. Learning dynamic context graphs for predicting social events. In *Proceedings of the 25th ACM SIGKDD*

International Conference on Knowledge Discovery & Data Mining. 1007–1016.

[7] Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2020. Dynamic knowledge graph based multi-event forecasting. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1585–1595.

[8] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. 2017. Protein interface prediction using graph convolutional networks. *Advances in neural information processing systems* 30 (2017).

[9] Yuyang Gao and Liang Zhao. 2018. Incomplete label multi-task ordinal regression for spatial event scale forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[10] Yuyang Gao, Liang Zhao, Lingfei Wu, Yanfang Ye, Hui Xiong, and Chaowei Yang. 2019. Incomplete label multi-task deep learning for spatio-temporal event subtype forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 3638–3646.

[11] Palash Goyal, Sujit Rokka Chhetri, and Arquimedes Canedo. 2020. dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *Knowledge-Based Systems* 187 (2020), 104816.

[12] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401* (2014).

[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[14] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[15] Wendi Ji, Keqiang Wang, Xiaoling Wang, Tingwei Chen, and Alexandra Cristea. 2020. Sequential recommender via time-aware attentive memory network. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 565–574.

[16] Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2019. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. *arXiv preprint arXiv:1904.05530* (2019).

[17] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. 2020. Representation learning for dynamic graphs: A survey. *J. Mach. Learn. Res.* 21, 70 (2020), 1–73.

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

[19] Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1269–1278.

[20] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.

[21] Yao Ma, Ziyi Guo, Zhaocun Ren, Jiliang Tang, and Dawei Yin. 2020. Streaming graph neural networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 719–728.

[22] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. 2018. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*. 181–196.

[23] Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826* (2017).

[24] Aditya K Menon, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. 2019. Multilabel reductions: what is my loss optimising? *Advances in Neural Information Processing Systems* 32 (2019).

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).

[26] Yue Ning, Rongrong Tao, Chandan K Reddy, Huzefa Rangwala, James C Starz, and Naren Ramakrishnan. 2018. STAPLE: Spatio-temporal precursor learning for event forecasting. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 99–107.

[27] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. 2020. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5363–5370.

[28] 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).

[29] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. 2020. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th international conference on web search and data mining*. 519–527.

[30] Xin Song, Haiyang Wang, and Bin Zhou. 2021. DGCN-rs: A Dilated Graph Convolutional Networks Jointly Modelling Relation and Semantic for Multi-event Forecasting. In *International Conference on Neural Information Processing*. Springer, 666–676.

[31] Aynaz Taheri, Kevin Gimpel, and Tanya Berger-Wolf. 2019. Learning to represent the evolution of dynamic graphs with recurrent models. In *Companion proceedings of the 2019 world wide web conference*. 301–307.

[32] Qiaoyu Tan, Jianwei Zhang, Ninghao Liu, Xiao Huang, Hongxia Yang, Jingren Zhou, and Xia Hu. 2021. Dynamic memory based attention network for sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4384–4392.

[33] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. 2019. Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*.

[34] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2019. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082* (2019).

[35] Xiaofeng Wang, Matthew S Gerber, and Donald E Brown. 2012. Automatic crime prediction using events extracted from twitter posts. In *International conference on social computing, behavioral-cultural modeling, and prediction*. Springer, 231–238.

[36] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 185–194.

[37] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2021. Inductive representation learning in temporal networks via causal anonymous walks. *arXiv preprint arXiv:2101.05974* (2021).

[38] Michael D Ward, Andreas Beger, Josh Cutler, Matthew Dickenson, Cassy Dorff, and Ben Radford. 2013. Comparing GDELT and ICEWS event data. *Analysis* 21, 1 (2013), 267–297.

[39] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962* (2020).

[40] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2019. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 9 (2019), 3848–3858.

[41] Liang Zhao, Qian Sun, Jieping Ye, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. 2015. Multi-task learning for spatio-temporal event forecasting. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1503–1512.